Technical Reference
LIMITED WARRANTY

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

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

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

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

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

THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH MAY VARY FROM STATE TO STATE.
FEDERAL COMMUNICATIONS COMMISSION
RADIO FREQUENCY INTERFERENCE STATEMENT

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

First Edition (August 1981)

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

© Copyright International Business Machines Corporation 1981
**CONTENTS**

**SECTION 1. HARDWARE OVERVIEW** ............................................. 1-1  
System Block Diagram ............................................................. 1-4

**SECTION 2. HARDWARE** ............................................................ 2-1  
System Board ................................................................. 2-3  
System Board Data Flow ......................................................... 2-6  
I/O Channel ....................................................................... 2-8  
I/O Channel Diagram ............................................................. 2-9  
System Board I/O Channel Description ................................. 2-10  
System Board Component Diagram ........................................... 2-13  
Keyboard ........................................................................... 2-14  
Keyboard Interface Block Diagram .......................................... 2-15  
Keyboard Diagram ................................................................. 2-16  
Keyboard Scan Codes ............................................................... 2-17  
Keyboard Interface Connector Specifications .......................... 2-18  
Cassette User Interface .......................................................... 2-19  
  Cassette Jumpers ............................................................... 2-19  
  Circuit Block Diagrams ......................................................... 2-19  
Cassette Interface Connector Specifications .......................... 2-21  
Speaker Interface ................................................................. 2-22  
  Speaker Drive System Block Diagram .................................... 2-22  
I/O Address Map ................................................................. 2-23  
System Memory Map ............................................................... 2-25  
System Board And Memory Expansion Switch Settings .............. 2-28  
5 1/4” Diskette Drives Switch Settings .................................... 2-29  
Monitor Type Switch Settings .................................................. 2-29  
System Board Memory Switch Settings ..................................... 2-30  
32/64 KB Memory Expansion Option Switch Settings ............... 2-31  

**Power Supply** ................................................................. 2-33  
Power Supply Location ........................................................... 2-34  
Input Requirements ............................................................... 2-34  
DC Output ............................................................................. 2-34  
AC Output ............................................................................. 2-34  
Power Supply Connectors And Pin Assignments ....................... 2-35  
Important Operating Characteristics .................................. 2-36  
  Over Voltage/Current Protection ........................................... 2-36  
  Signal Requirements ............................................................ 2-36
IBM Monochrome Display and Parallel

Parallel Interface Description ......................................... 2-37
IBM Monochrome Display Adapter Block Diagram .................... 2-38
System Channel Interface .................................................. 2-39
  Lines Used ............................................................... 2-39
  Loads ................................................................. 2-39
  Special Timing ....................................................... 2-39
  Data Rates ............................................................ 2-39
  Interrupt and DMA Response Requirements ........................ 2-39
Modes Of Operation .......................................................... 2-40
Programming Considerations .............................................. 2-41
  Programming the 6845 CRT Controller ............................ 2-41
  Sequence of Events .................................................. 2-41
  Memory Requirements ................................................ 2-41
  DMA Channel .......................................................... 2-42
  Interrupt Levels ..................................................... 2-42
  I/O Address and Bit Map ............................................. 2-42
IBM Monochrome Display .................................................. 2-43
  Operating Characteristics ......................................... 2-43
IBM Monochrome Direct Drive Interface and
  Pin Assignment .......................................................... 2-44
Color/Graphics Monitor Adapter ......................................... 2-45
Color/Graphics Monitor Adapter Block Diagram ...................... 2-47
Major Components Definitions .......................................... 2-48
  Motorola 6845 CRT Controller .................................... 2-48
  Mode Set and Status Registers .................................. 2-48
  Display Buffer ....................................................... 2-48
  Character Generator ............................................... 2-48
  Timing Generator .................................................... 2-48
  Composite Color Generator ...................................... 2-48
Modes Of Operation .......................................................... 2-49
  Alphanumeric Mode .................................................. 2-49
  Color TV .............................................................. 2-49
  Color Monitor ......................................................... 2-50
IBM Monochrome Display Adapter Vs. Color/Graphics .............. 2-51
  Adapter Attribute Relationship .................................. 2-51
Color/Graphics Modes ..................................................... 2-51
  Graphics Storage Map ............................................... 2-52
  Description of Basic Operations .................................. 2-54
  Summary of Available Colors .................................... 2-55
INS 8250 Functional Pin Description .......................... 2-129
  Input Signals ................................................................ 2-129
  Output Signals ................................................................ 2-132
  Input/Output Signals ..................................................... 2-133
Programming Considerations ........................................... 2-133
  Asynchronous Communications Reset Functions ............ 2-133
INS 8250 Accessable Registers ........................................ 2-134
INS 8250 Line Control Register ....................................... 2-134
INS 8250 Programmable Baud Rate Generator .................... 2-135
Line Status Register .................................................... 2-137
Interrupt Identification Register ..................................... 2-139
Interrupt Enable Register .............................................. 2-141
Modem Control Register ................................................ 2-142
Modem Status Register .................................................. 2-143
Receiver Buffer Register ................................................ 2-144
Transmitter Holding Register ......................................... 2-145
Selecting The Interface Format ....................................... 2-146
Asynchronous Communications Adapter Connector
  Interface Specifications .................................................. 2-147

SECTION 3. ROM and SYSTEM USAGE .......................... 3-1
ROM BIOS ........................................................................ 3-2
  Use of BIOS .................................................................... 3-2
  Parameter Passing .......................................................... 3-2
  Interrupt Vector Listing .................................................. 3-3
  Vectors With Special Meaning ........................................ 3-5
    Interrupt 1CH - Timer Tick ........................................ 3-5
    Interrupt 1DH - Video Parameters ................................ 3-5
    Interrupt 1EH - Diskette Parameters ............................. 3-5
    Interrupt 1FH - Graphics Character Extensions .............. 3-6
  Other Read/Write Memory Usage .................................... 3-6
BIOS Programming Tip .................................................... 3-6
BIOS Memory Map .......................................................... 3-7

BIOS Cassette Logic .......................................................... 3-8
  Interrupt 15 ................................................................. 3-8
  Cassette Write ............................................................. 3-8
  Cassette Read ............................................................... 3-9
  Data Record Architecture ............................................... 3-10
  Error Recovery ............................................................. 3-10
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard Encoding and Usage</td>
<td>3-11</td>
</tr>
<tr>
<td>Encoding</td>
<td>3-11</td>
</tr>
<tr>
<td>Character Codes</td>
<td>3-11</td>
</tr>
<tr>
<td>Extended Codes</td>
<td>3-13</td>
</tr>
<tr>
<td>Extended Functions</td>
<td>3-13</td>
</tr>
<tr>
<td>Shift States</td>
<td>3-14</td>
</tr>
<tr>
<td>Shift Key Priorities</td>
<td>3-15</td>
</tr>
<tr>
<td>Special Handling</td>
<td>3-15</td>
</tr>
<tr>
<td>System Reset</td>
<td>3-15</td>
</tr>
<tr>
<td>Break</td>
<td>3-16</td>
</tr>
<tr>
<td>Pause</td>
<td>3-16</td>
</tr>
<tr>
<td>Print Screen</td>
<td>3-16</td>
</tr>
<tr>
<td>Keyboard Usage</td>
<td>3-17</td>
</tr>
<tr>
<td>BASIC Screen Editor Special Functions</td>
<td>3-19</td>
</tr>
<tr>
<td>DOS Special Functions</td>
<td>3-19</td>
</tr>
<tr>
<td>Low Memory Maps</td>
<td>3-21</td>
</tr>
<tr>
<td>0-7F Interrupt Vectors</td>
<td>3-21</td>
</tr>
<tr>
<td>BASIC and DOS Reserved Interrupts (80-3FF)</td>
<td>3-22</td>
</tr>
<tr>
<td>Reserved Memory Locations (400-5FF)</td>
<td>3-22</td>
</tr>
<tr>
<td>BASIC Workspace Variables</td>
<td>3-23</td>
</tr>
</tbody>
</table>

**APPENDICES**

<table>
<thead>
<tr>
<th>Appendix</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appendix A: ROM BIOS Listing</td>
<td>A-1</td>
</tr>
<tr>
<td>Appendix B: Assembly Instruction Set Reference</td>
<td>B-1</td>
</tr>
<tr>
<td>Appendix C: Of Characters Keystrokes and Color</td>
<td>C-1</td>
</tr>
<tr>
<td>Appendix D: Logic Diagrams</td>
<td>D-1</td>
</tr>
<tr>
<td>Appendix E: Unit Specifications</td>
<td>E-1</td>
</tr>
<tr>
<td>Glossary</td>
<td>G-1</td>
</tr>
<tr>
<td>Bibliography</td>
<td>Bib-1</td>
</tr>
<tr>
<td>Index</td>
<td>I-1</td>
</tr>
</tbody>
</table>
## FIGURE LISTING

1. System Block Diagram .................................. 1-4
2. System Board Data Flow ................................ 2-6, 7
3. I/O Channel Diagram ..................................... 2-9
4. System Board Component Diagram ....................... 2-13
5. Keyboard Interface Block Diagram ...................... 2-15
6. Keyboard Diagram ........................................ 2-16
7. Cassette Interface Read Hardware ...................... 2-19
8. Cassette Interface Write Hardware ...................... 2-20
9. Cassette Motor Control .................................. 2-20
10. Speaker Drive System Block Diagram ................... 2-22
11. System Memory Map .................................... 2-25
12. System Memory Map (Increments of 16KB) ............. 2-26
13. Power Supply and Connectors ........................... 2-35
14. IBM Monochrome Display Adapter Block Diagram ...... 2-38
15. Color/Graphics Monitor Adapter Block Diagram ....... 2-47
16. Parallel Printer Adapter Block Diagram ............... 2-66
17. Location of (Printer) DIP Switches .................... 2-72
18. Parallel Interface Timing ................................ 2-77
19. 5 1/4” Diskette Drive Adapter Block Diagram ........ 2-90
20. Game Control Adapter Block Diagram .................. 2-117
21. Joystick Schematic ..................................... 2-121
22. Asynchronous Communications Adapter Block Diagram ........................................... 2-124
23. Current Loop Interface .................................. 2-127
24. Selecting The Interface Format ......................... 2-146
25. BIOS Memory Map ....................................... 3-7
<table>
<thead>
<tr>
<th>TABLE LISTING</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Keyboard Scan Codes .................................. 2-17</td>
</tr>
<tr>
<td>2. 6845 Initialization Parameters ...................... 2-41</td>
</tr>
<tr>
<td>3. Monochrome Vs Color/Graphics Attributes .............. 2-51</td>
</tr>
<tr>
<td>4. Color/Graphics Modes .................................. 2-51</td>
</tr>
<tr>
<td>5. Summary of Available Colors .......................... 2-55</td>
</tr>
<tr>
<td>6. 6845 Register Description ........................... 2-56</td>
</tr>
<tr>
<td>7. Printer Specifications ................................ 2-71</td>
</tr>
<tr>
<td>8. Functions and Conditions of DIP Switch 1 ............. 2-72</td>
</tr>
<tr>
<td>9. Functions and Conditions of DIP Switch 2 ............. 2-73</td>
</tr>
<tr>
<td>10. Connector Pin Assignment and Description of Interface Signals .......................... 2-74</td>
</tr>
<tr>
<td>11. ASCII Coding Table ................................... 2-78</td>
</tr>
<tr>
<td>12. DC1/DC3 and Data Entry ............................... 2-82</td>
</tr>
<tr>
<td>13. Symbol Description ................................... 2-94</td>
</tr>
<tr>
<td>14. Status Register 0 ..................................... 2-100</td>
</tr>
<tr>
<td>15. Status Register 1 ..................................... 2-101</td>
</tr>
<tr>
<td>16. Status Register 2 ..................................... 2-102</td>
</tr>
<tr>
<td>17. Status Register 3 ..................................... 2-103</td>
</tr>
<tr>
<td>18. Mechanical and Electrical Specifications ............. 2-112</td>
</tr>
<tr>
<td>19. Memory Module Pin Configuration ...................... 2-114</td>
</tr>
<tr>
<td>20. DIP Module Start Address .............................. 2-115</td>
</tr>
<tr>
<td>21. I/O Decodes (3F8 - 3FF) .............................. 2-125</td>
</tr>
<tr>
<td>22. Asynchronous Communications Reset Functions ......... 2-133</td>
</tr>
<tr>
<td>23. BAUD Rate at 1.843 Mhz ............................... 2-137</td>
</tr>
<tr>
<td>24. Interrupt Control Functions (Asynchronous) .......... 2-140</td>
</tr>
<tr>
<td>25. Character Codes ....................................... 3-11</td>
</tr>
<tr>
<td>26. Keyboard Extended Functions ......................... 3-14</td>
</tr>
<tr>
<td>27. Keyboard - Commonly Used Functions ................... 3-17</td>
</tr>
<tr>
<td>28. Basic Screen Editor Special Functions ................. 3-19</td>
</tr>
<tr>
<td>29. DOS Special Functions ................................. 3-19</td>
</tr>
<tr>
<td>30. 0-7F Interrupt Vectors ............................... 3-21</td>
</tr>
<tr>
<td>31. Basic &amp; DOS Reserved Interrupts (80-3FF) ............. 3-22</td>
</tr>
<tr>
<td>32. Reserved Memory Locations (400-5FF) ................. 3-22</td>
</tr>
</tbody>
</table>
SECTION I.
HARDWARE OVERVIEW

The IBM Personal Computer has two major elements; a System Unit and a keyboard. In addition, a variety of options are offered including one or two 5-1/4” Diskette Drives with adapter which can be housed inside the System Unit, an IBM Monochrome Display, an 80 CPS Matrix Printer, two display adapters, storage increments to 256 KB, an Asynchronous Communications Adapter, Printer Adapter and a Game Control Adapter.

The System Unit is the heart of your IBM Personal Computer system. The System Unit houses the microprocessor, Read-Only Memory (ROM), Read/Write Memory, Power Supply, and System Expansion Slots for the attachment of up to five options. One or two 5-1/4” Diskette Drives can also be mounted in the system Unit providing 160KB of storage each.

The System Board is a large board which fits horizontally in the base of the System Unit and includes the microprocessor, 40KB ROM and 16KB memory. The memory can be expanded in 16KB increments to 64KB. The System Board also includes an enhanced version of the Microsoft BASIC-80 Interpreter without diskette functions. The BASIC Interpreter is included in the ROM. The System Board also permits the attachment of an audio cassette recorder for loading or saving programs and data.

The System Unit power system is a 63.5 watt, 4 level DC and 120 AC unit. It is a switching regulator design, allowing for light weight and high efficiency. The DC power capacity is designed to support an expanded system.

The 5-1/4” Diskette Drive Adapter fits into one of the five System Expansion Slots. This attachment supports two internal drives. The 5-1/4” Diskette Drive Adapter uses write precompensation and a phase lock loop for clock and data recovery.

The 5-1/4” Diskette Drive permits the IBM Personal Computer to read, write and store data on 5-1/4” diskettes. Each diskette stores approximately 160KB of data. Two of these drives may be installed internally in the System Unit.

The keyboard is attached to the System Unit with a light-weight, coiled cable. The keyboard features 83 keys, and offers commonly used data and word processing functions in a design combining the familiar typewriter and calculator pad layouts.
A base system requires one of two different display adapters, either a Color/Graphics medium resolution Monitor adapter or a high resolution monochrome alphanumeric adapter with a parallel printer adapter.

The Color/Graphics adapter operates at standard television frequencies (15.750Hz), allowing attachment to a variety of industry standard monitors, including home TVs with a user supplied RF modulator.

The Color/Graphic Monitor adapter supports a variety of modes selected by program control. The adapter supports color or black and white alphanumeric modes with line width of 40 or 80 characters and 25 lines. In the alphanumeric mode there are 256 characters.

This adapter provides both a standard composite video and direct drive outputs. In addition, a light pen feature input port is provided.

The IBM Monochrome Display is a high resolution green phosphor display offering the personal computer user quality usually found on larger computer systems. The display features an 11-1/2” screen with an anti-glare surface and a variety of highlighting choices. The screen displays 25 lines of 80 characters. It supports 256 different letters, numbers and special characters that are formed in a nine by 14 dot matrix.

The IBM Monochrome Display requires the Monochrome Display and Printer Adapter Option. This option installs in one of the System Unit’s five System Expansion Slots. The display is powered from the System Unit.

The 80 CPS Matrix Printer is a versatile, low cost, quality printer for the IBM Personal Computer. It prints in both directions at a nominal horizontal speed of 80 characters per second on continuous-feed, single or multi-part paper. The printer features four character sizes (40, 66, 80 or 132 characters per line), Power-on Self-test and simple paper loading and ribbon cartridge uppercase and lower case ASCII character set and 64 special graphic characters.

The 80 CPS Matrix Printer requires either the Monochrome Display and Printer Adapter or the Printer Adapter. These options install in one of the Systems Unit’s five System Expansion Slots. The Printer requires standard 120 volt, 60 Hz power through its own power cord. The printer requires the Printer Cable Option for attachment to the System Unit.
The 16KB Memory Expansion Kits allow you to increase the memory size of your IBM Personal Computer. The base system comes standard with 16KB of memory. Up to three 16KB Memory Expansion Kits may be installed to increase the memory size to 64KB. Memory can be further increased to 256KB with additional memory options once these three Expansion Kits are installed.

The Expansion Kits plug into the System Board and must be installed sequentially. They do not occupy any of the five System Expansion Slots.

The 32KB and 64KB Memory Expansion Options permit you to increase memory capacity beyond 64KB. Multiple 32KB and 64KB Memory Expansion Options may be installed as long as System Expansion Slots are available. A maximum of three 64KB memory options may be installed for a total of 256KB of memory.

The 32KB and 64KB Memory Expansion Options require a System Expansion Slot in the System Unit. The first 64KB on the System Board is required before 32KB and 64KB Memory Expansion Options can be installed.

The Asynchronous Communications Adapter provides a channel to data processing or input/output devices outside of your immediate system. These can be connected by telephone using a plug-in modem, or directly by cable when the device is nearby.

This option utilizes the RS232C asynchronous (start-stop) interface permitting attachment to a variety of devices including a large "host" computer or another IBM Personal Computer.

This option supports 50 to 9600 BPS transmission speeds. One 25 pin "D" shell, male type connector is provided to attach various peripheral devices. A "current loop" interface is located in the same connector, and a jumper block is provided to manually select either the voltage or the current loop interface.

The Asynchronous Communications Adapter requires a System Expansion Slot in the System Unit. An external modem is required for telephone line transmission.

The Game Control Adapter permits the attachment of user-supplied joysticks or paddles. Two joysticks and up to four paddles may be attached. IBM does not manufacture either the joysticks or the paddles. This option provides connectors for joysticks or paddles and requires a System Expansion Slot in the System Unit.

A block diagram of the system is on the following page (1-4).
Figure 1. SYSTEM BLOCK DIAGRAM
SECTION 2. HARDWARE

Contents:

System Board .................................................. 2-3
Power Supply ................................................... 2-33
IBM Monochrome Display And Printer Adapter .... 2-37
IBM Monochrome Display ................................. 2-43
Color/Graphics Monitor Adapter ................. 2-45
Printer Adapter and Printer ......................... 2-65
5-1/4” Diskette Drive Adapter ...................... 2-89
5-1/4” Diskette Drive .................................. 2-110
Memory Expansion Options 32KB and 64KB .... 2-113
Game Control Adapter ................................. 2-117
Asynchronous Communications Adapter .......... 2-123
SYSTEM BOARD

The System Board fits horizontally in the base of the System Unit and has dimensions of approximately 8-1/2 inches by 11 inches. The System Board is a multilayer single land-per-channel design, with ground and power internal planes provided. DC power and a signal from the power supply enter the board through two six pin connectors. Other connectors on the board are for attaching the keyboard, audio cassette, and the speaker. Five 62-pin card edge sockets are also mounted on the System Board. The system I/O channel is bussed across these five I/O slots.

There are 16 (13 used) Dual In-line Package (DIP) switches mounted on the card which can be read under program control. These switches are used to indicate to the system software what options are installed. They are used to indicate amounts of installed storage, both on the System Board and in the System Expansion slots, type of display adapter installed, and desired operation modes upon power-up; i.e., color or black and white and 80- or 40 character lines. Switches are also used to identify when the operating system is to be loaded from diskette, and how many diskette drives are attached.

The major elements of the System Board are divided into five major functional areas. They are, 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 functions are described in detail in this section, except for the I/O channel, which has its own section. Figure 2.0 “System Board Data Flow” page 2-6, illustrates these functional areas.

The heart of the System Board is the Intel 8088 microprocessor. This processor is an 8-bit bus version of the 16-bit 8086 processor by Intel. It is software compatible with the 8086 and, thus, supports 16-bit operations including multiply and divide. The processor supports 20 bits of addressing (1 megabyte of storage). The processor is implemented in maximum mode so a co-processor can be added as a feature. The processor is operated at 4.77 Mhz. This frequency is derived from a 14.31818 Mhz crystal which is divided by three for the processor clock and by four 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 microsec (m sec).
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 are provided to 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 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 ns if the processor ready line is not deactivated. Refresh DMA cycles take four clocks or 840 ns.

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

Of the eight prioritized levels of interrupt, six are bussed to the I/O slots for use by feature cards. Two levels are used on the System Board. Level 0, the highest priority, is attached to Channel 1 of the timer counter and provides a periodic interrupt. 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 is designed to support both ROM and Read/Write Memory. The System Board contains space for 48K x 8 of ROM or EPROM. Six module sockets are provided, each capable of accepting an 8K x 8 device. Five of the sockets are populated with 40 KB of ROM. This ROM contains the Cassette BASIC interpreter, cassette operating system, Power-on Self-test, I/O drivers, dot patterns for 128 characters in graphics mode, and a diskette bootstrap loader. The ROM is packaged in 24-pin modules and has an access time of 250 ns and a cycle time of 375 ns.

The System Board also contains from 16K x 9 to 64K x 9 of Read/Write Memory. A minimum system would have 16 KB of memory with module sockets for an additional 48 KB. In a cassette version of the system, approximately 4 KB is used by the system leaving approximately 12 KB of user's space for BASIC programs. Additional memory beyond the System Board's maximum of 64 KB, is obtained by adding memory cards in the System Expansion slots.
The memory is dynamic 16K x 1 chips with an access time of 250 ns and a cycle time of 410 ns. All R/W memory is parity checked.

The System Board contains circuits for attaching an audio cassette, the serial keyboard, and the speaker. The cassette adapter allows the attachment of any good quality audio cassette via either the microphone or auxiliary inputs. The board has a jumper for either input. This interface also provides a cassette motor control line for transport starting and stopping under program control. This interface reads and writes the audio cassette at a data rate of between 1,000 and 2,000 baud. The baud rate is variable and dependent on data content since a different bit-cell time is used for 0’s and 1’s. For diagnostic purposes, the tape interface can loop read to write to test the board’s circuits. The system software blocks cassette data, generates and checks data with a Cyclic Redundancy Check (CRC).

The processor also contains the adapter circuits for attaching the serial interface from the keyboard. This generates an interrupt to the processor when a complete scan code is received. This interface can request execution of a diagnostic in the keyboard.

Both the keyboard and cassette interfaces are provided via 5-pin DIN connectors, which are right angle mounts on the System Board and extend through the rear panel of the System Unit.

The system is provided with a 2-1/4-inch audio speaker mounted inside the System Unit. The System Board contains the control circuits and driver for the speaker. The speaker connects through a 2-wire interface which attaches to a 4-pin header on the System Board.

The speaker drive circuit is capable of approximately a 1/2 watt of power. The control circuits allow the speaker to be driven several different ways. First, a direct program control register bit may be toggled to generate a pulse train; second, the output of Channel 2 of the timer counter may be programmed to generate a waveform to the speaker. Third, the clock input to the timer/counter can be modulated with a program controlled I/O Register Bit. All three forms of control may be performed simultaneously.
Figure 2. SYSTEM BOARD DATA FLOW (SHEET 1 OF 2)
Figure 2. SYSTEM BOARD DATA FLOW (SHEET 2)
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 card +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 210 ns clocks or 1.05 m sec/byte. All DMA transfers require five clocks for a cycle time of 1.05 m sec/byte. Refresh cycles are present once every 72 clocks or approximately 15 m sec and require five clocks or approximately 7% of the bus bandwidth.

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

A channel check line exists for reporting error conditions to the processor. Activating this line results in a NMI to the 8088 processor. Memory Expansion Options use this line to report parity errors.

The I/O channel is repowered so there is sufficient drive to power all five System Expansion Slots, assuming two loads per slot. The IBM Option I/O adapters typically use only one load. A graphic illustration of the System I/O Channel and its descriptions are on the following pages.
I/O Channel Diagram

Figure 3. I/O CHANNEL DIAGRAM
System Board I/O Channel Description

The following is a description of the IBM Personal Computer System Board I/O Channel. All signal 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: This signal is a high speed clock with a 70 nsec. period (14.31818 MHz.) It has a 50% duty cycle.</td>
</tr>
<tr>
<td>CLK</td>
<td>O</td>
<td>Clock: This is the system clock. It is a divide-by-three of the oscillator and has a period of 210 nsec. (4.77 MHz.) The clock has a 33% duty cycle.</td>
</tr>
<tr>
<td>RESET DRV</td>
<td>O</td>
<td>Reset Driver: 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>A0-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) while A19 is the Most Significant Bit (MSB). These lines are generated by either the processor or the 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 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 in conjunction with AEN). Processor addresses are latched with the falling edge of ALE.</td>
</tr>
<tr>
<td>I/O CH CK</td>
<td>I</td>
<td>-I/O Channel Check: This line provides the CPU 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>
</tbody>
</table>
### I/O CH RDY
I

**I/O Channel Ready:** This line (normally high or "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 for any period in excess of 10 clock cycles (2.1 usec.) Machine cycles (I/O or memory) are extended by an integral number of CLK cycles (210 ns).

### IRQ2-IRQ7
I

**Interrupt Request 2 to 7:** These lines are used to signal the processor that an I/O device requires attention. They are prioritized with IRQ2 as the highest priority and IRQ7 as the lowest. An Interrupt Request is generated by raising an IRQ line (Low to High) and holding it high until it is acknowledged by the processor (Interrupt Service Routine).

### IOR
O

**-I/O Read Command:** This command line instructs an I/O device to drive its data onto the data bus. It may be driven by the processor or the DMA Controller. This signal is active LOW.

### IOW
O

**-I/O Write Command:** This command line instructs an I/O device to read the data on the data bus. It may be driven by the processor or the DMA controller. This signal is active LOW.

### MEMR

**-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.

### MEMW
O

**-Memory Write Command:** This command line instructs the memory to store the data present on the data bus. It may be driven by the processor or the DMA Controller. This signal is active LOW.
DRQ1-DRQ3 I DMA Request 1 to 3: These lines are asynchronous channel requests used by peripheral devices to gain DMA service. They are prioritized with DRQ1 having highest priority and DRQ3 the lowest. 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.

DACK0-DACK3 O -DMA Acknowledge 0 to 3: These lines are used to acknowledge DMA requests (DRQ1-DRQ3) and to refresh system dynamic memory (DACK0). They are active LOW.

AEN O Address Enable: This line is used to degate the processor and other devices from the I/O Channel to allow Direct Memory Access (DMA) transfers to take place. When this line is active (HIGH), the DMA Controller has control of the address bus, data bus, read command lines, (memory and I/O), and the write command lines, (memory and I/O).

T/C O Terminal Count: This line provides a pulse when the terminal count for any DMA channel is reached. This signal is active HIGH.

The following voltages are available on the System Board I/O Channel:

+5 Vdc ± 5%, Located on 2 connector pins.
-5 Vdc ± 10%, Located on 1 connector pin.
+12 Vdc ± 5%, Located on 1 connector pin.
-12 Vdc ± 10%, Located on 1 connector pin.
GND (Ground), Located on 3 connector pins.
Figure 4. SYSTEM BOARD COMPONENT DIAGRAM
Keyboard

The Keyboard is a device separate from the System Unit. It is attached via a serial interface cable approximately 6 feet in length which plugs into the rear of the System Unit. The attaching cable is coiled, like that of a telephone headset, and is a shielded four-wire wire connection. The interface contains power (+5 Vdc), ground and two bidirectional signal lines. The cable is permanently attached at the keyboard end and plugs into the System Unit via a DIN connector.

The keyboard uses a capacitive technology with a microcomputer (Intel 8048) performing the keyboard scan function. The keyboard is packaged in a low-profile enclosure with a tilt adjustment for 5 degree or 15 degree orientations.

The keyboard contains 83 keys laid out in three major groupings. The central portion of the keyboard contains a standard typewriter keyboard layout. On the left side, arranged as a 2x5 block, are 10 function keys. These keys are user-defined by software. On the right is a 16-key, key pad area. This area is, defined by the software but contains legends for the functions of numeric entry, cursor control calculator pad screen edit.

The keyboard interface is defined so system software has the maximum flexibility in defining keyboard operations such as shift states of keys, make/break keys, and typematic operation. This is accomplished by having the keyboard return scan codes rather than American National Standard Control Characters (ASCII) codes. In addition, all keys except control keys are typematic and generates both a make and a break-scan code. For example, key 1 produces scan code 01 on make, and code 81 on break. Break codes are formed by adding X '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 and when requested by the System Unit. This diagnostic CRC checks the microcomputer ROM, tests memory and checks for stuck keys. Additional functions are: keyboard scanning, key debounce, buffering of up to 20 key scan codes, maintaining bidirectional serial communications with the System Unit, and executing the hand shake protocol required by each scan code transfer. A keyboard diagram and table of scan codes are on the following pages. Figure (5) is a block diagram of the keyboard interface on the System Board.
Figure 5. KEYBOARD INTERFACE BLOCK DIAGRAM
NOTE

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

Figure 6. KEYBOARD DIAGRAM
### Table 1. Keyboard Scan Codes

<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 Interface Connector Specifications

<table>
<thead>
<tr>
<th>PIN</th>
<th>SIGNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+ Keyboard Clock</td>
</tr>
<tr>
<td>2</td>
<td>+ Keyboard Data</td>
</tr>
<tr>
<td>3</td>
<td>- Keyboard Reset (Not used by keyboard)</td>
</tr>
<tr>
<td>4</td>
<td>Ground</td>
</tr>
<tr>
<td>5</td>
<td>+5 Volts</td>
</tr>
</tbody>
</table>
Cassette User Interface

The cassette interface control is implemented in software. (See Firmware Section). An 8253 timer output is used to control the data to the cassette recorder. This output exits the System Board, at the rear, through pin 5 of a DIN connector. The cassette input data is read by an 8255A-5 Programmable Peripheral Interface (PPI) input port bit. This signal is received through pin 4 of the cassette connector. Software algorithms are used to generate and read cassette data. The cassette drive motor is controlled through pins 1 & 3 of the cassette connector. The motor on/off is controlled by an 8255A-PPI output port bit (Port ‘61H’, bit 3). The 8255A address and bit assignments are defined in the I/O Address Map page. On the following pages are read, write, and motor control block diagrams.

Cassette Jumpers

A 2x2 Berg Pin and Jumper are used on the cassette Data Out line. The jumper will allow the Data Out line to be used as a microphone input (75 mv.) when the jumper is placed across M and C pins. An auxiliary input is available when the jumper is placed across the A and C pins. The auxiliary input provides a .68 volt input to the recorder. Refer to System Board Component Diagram page (2-13) for cassette jumper location.

Circuit Block Diagrams

[Diagram of Circuit Block Diagrams]

Figure 7. CASSETTE INTERFACE READ HARDWARE
Figure 8. CASSETTE INTERFACE WRITE HARDWARE

Figure 9. CASSETTE MOTOR CONTROL
## Cassette Interface Connector Specifications

### Rear Panel

- **Cassette Interface Connector:**
  - 5 PIN DIN CONNECTOR

### Pin Signal Electrical Characteristics

<table>
<thead>
<tr>
<th>PIN</th>
<th>SIGNAL</th>
<th>ELECTRICAL CHARACTERISTICS*</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Motor Control</td>
<td>Common from Relay</td>
</tr>
<tr>
<td>2</td>
<td>Ground</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Motor Control</td>
<td>6 VDC; 1A (Relay N.O.)</td>
</tr>
<tr>
<td>4</td>
<td>Data In</td>
<td>500nA at ±13V - at 1,000 - 2,000 Baud</td>
</tr>
<tr>
<td>5</td>
<td>Data Out (Mic or Aux)</td>
<td>250 μA at [.68V or ** 75 mV]</td>
</tr>
</tbody>
</table>

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

**Data out can be chosen using a jumper located on planar. (AUX → .68V or MIC → 75 mV).

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

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

1. An 8255A-5 PPI output bit. The address and bit are defined in the I/O Address Map pages 2-23 and 2-24.

2. A timer Channel Clock out where the output is programmable within the functions of the 8253-5 timer with a 1.19 Mhz clock input. The timer gate is also controlled by an 8255A-5 PPI output port bit. Address and bit assignment are in the I/O Address Map pages 2-23 and 2-24.

---

![Speaker Drive System Block Diagram]

**Figure 10. SPEAKER DRIVE SYSTEM BLOCK DIAGRAM**

Channel 2 (Tone generation for Speaker)
- GATE 2 — Controlled by 8255A-5 PPI Bit
  (See I/O Map)
- CLK IN 2 — 1.19318 Mhz OSC
- CLK OUT 2 — Used to drive Speaker
  — Used to write data on the Audio Cassette

Speaker Connection - 4 Pin Berg Connector, Refer to System Board Diagram page 2-13 for speaker connection.

<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>
## I/O Address Map

<table>
<thead>
<tr>
<th>HEX RANGE</th>
<th>9 8</th>
<th>7 6 5 4</th>
<th>3 2 1 0</th>
<th>DEVICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>00–0F</td>
<td>0 0</td>
<td>0 0 0 Z</td>
<td>A3 A2 A1 A0</td>
<td>DMA CHIP 8237–2</td>
</tr>
<tr>
<td>20–21</td>
<td>0 0</td>
<td>0 0 1 Z</td>
<td>Z Z Z A0</td>
<td>INTERRUPT 8259A</td>
</tr>
<tr>
<td>40–43</td>
<td>0 0</td>
<td>0 1 0 Z</td>
<td>Z Z A1 A0</td>
<td>TIMER 8253–5</td>
</tr>
<tr>
<td>60–63</td>
<td>0 0</td>
<td>0 1 1 Z</td>
<td>Z Z A1 A0</td>
<td>PPI 8255A–5</td>
</tr>
<tr>
<td>80–83</td>
<td>0 0</td>
<td>1 0 0 Z</td>
<td>Z Z A1 A0</td>
<td>DMA PAGE REGS</td>
</tr>
<tr>
<td>0–0F</td>
<td>0 0</td>
<td>1 0 1</td>
<td></td>
<td>DMA CHIP 8237–2</td>
</tr>
<tr>
<td>AX</td>
<td>0 0</td>
<td>1 1 0</td>
<td></td>
<td>RESERVED</td>
</tr>
<tr>
<td>CX</td>
<td>0 0</td>
<td>1 1 1</td>
<td></td>
<td>RESERVED</td>
</tr>
<tr>
<td>EX</td>
<td>0 0</td>
<td>1 1 1</td>
<td></td>
<td>RESERVED</td>
</tr>
<tr>
<td>3F8–3FF</td>
<td>1 1</td>
<td>1 1 1 1</td>
<td>1 A2 A1 A0</td>
<td>TP RS–232–C CD</td>
</tr>
<tr>
<td>3F0–3F7</td>
<td>1 1</td>
<td>1 1 1 1</td>
<td>0 A2 A1 A0</td>
<td>5 1/4&quot; DRV ADAPTER</td>
</tr>
<tr>
<td>2F8–2FF</td>
<td>1 0</td>
<td>1 1 1 1</td>
<td>1 A2 A1 A0</td>
<td>RESERVED</td>
</tr>
<tr>
<td>378–37F</td>
<td>1 1</td>
<td>0 1 1 1</td>
<td>1 Z A1 A0</td>
<td>PARALLEL PRTR PRT</td>
</tr>
<tr>
<td>300–3DF</td>
<td>1 1</td>
<td>1 1 0 1</td>
<td>A3 A2 A1 A0</td>
<td>COLOR/GRAPHICS ADAPTER</td>
</tr>
<tr>
<td>278–27F</td>
<td>1 0</td>
<td>0 1 1 1</td>
<td>1 Z A1 A0</td>
<td>RESERVED</td>
</tr>
<tr>
<td>200–20F</td>
<td>1 0</td>
<td>0 0 0 0</td>
<td>A3 A2 A1 A0</td>
<td>GAME I/O ADAPTER</td>
</tr>
<tr>
<td>380–3BF</td>
<td>1 1</td>
<td>1 0 1 1</td>
<td>A3 A2 A1 A0</td>
<td>IBM MONOCHROME DISPLAY</td>
</tr>
</tbody>
</table>

Z = Don’t Care, i.e., Not in Decode

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

  Set mask: write X‘80’ to I/O Address X‘A0’ (enable NMI)
  
  Clear mask: write X‘00’ to I/O Address X‘A0’ (disable NMI)
# I/O Address Map

<table>
<thead>
<tr>
<th>PA0</th>
<th>+KBD SCAN CODE 0</th>
<th>X'0060'</th>
<th>I/O READ/WRITE MEMORY (SW2-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>I/O READ/WRITE MEMORY (SW2-2)</td>
</tr>
<tr>
<td>I</td>
<td>1</td>
<td></td>
<td>I/O READ/WRITE MEMORY (SW2-3)</td>
</tr>
<tr>
<td>N</td>
<td>2</td>
<td></td>
<td>I/O READ/WRITE MEMORY (SW2-4)</td>
</tr>
<tr>
<td>P</td>
<td>3</td>
<td></td>
<td>+CASSETTE DATA IN</td>
</tr>
<tr>
<td>U</td>
<td>4</td>
<td></td>
<td>+CASSETTE DATA IN (SW2-5)</td>
</tr>
<tr>
<td>T</td>
<td>5</td>
<td></td>
<td>+CASSETTE DATA IN (SW2-6)</td>
</tr>
<tr>
<td>U</td>
<td>6</td>
<td></td>
<td>+CASSETTE DATA IN (SW2-7)</td>
</tr>
<tr>
<td>T</td>
<td>7</td>
<td></td>
<td>+CASSETTE DATA IN (SW2-8)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PA0</th>
<th>+KBD SCAN CODE 1</th>
<th>X'0061'</th>
<th>+TIMER 2 GATE SPEAKER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>+SPEAKER DATA</td>
</tr>
<tr>
<td>O</td>
<td>1</td>
<td></td>
<td>+CASSETTE MOTOR OFF</td>
</tr>
<tr>
<td>U</td>
<td>2</td>
<td></td>
<td>16K BYTES</td>
</tr>
<tr>
<td>T</td>
<td>3</td>
<td></td>
<td>32K BYTES</td>
</tr>
<tr>
<td>U</td>
<td>5</td>
<td></td>
<td>48K BYTES</td>
</tr>
<tr>
<td>T</td>
<td>6</td>
<td></td>
<td>64K BYTES</td>
</tr>
<tr>
<td>U</td>
<td>7</td>
<td></td>
<td>(ENABLE KBD) OR + (CLR KBD &amp; ENABLE SENSE SW'S)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PA0</th>
<th>+KBD SCAN CODE 2</th>
<th>X'0062'</th>
<th>I/O READ/WRITE MEMORY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>+CASSETTE MOTOR OFF</td>
</tr>
<tr>
<td>I</td>
<td>1</td>
<td></td>
<td>+CASSETTE MOTOR OFF (SW2-1)</td>
</tr>
<tr>
<td>N</td>
<td>2</td>
<td></td>
<td>+CASSETTE MOTOR OFF (SW2-2)</td>
</tr>
<tr>
<td>P</td>
<td>3</td>
<td></td>
<td>+CASSETTE MOTOR OFF (SW2-3)</td>
</tr>
<tr>
<td>U</td>
<td>4</td>
<td></td>
<td>+CASSETTE MOTOR OFF (SW2-4)</td>
</tr>
<tr>
<td>T</td>
<td>5</td>
<td></td>
<td>+CASSETTE MOTOR OFF (SW2-5)</td>
</tr>
<tr>
<td>U</td>
<td>6</td>
<td></td>
<td>+CASSETTE MOTOR OFF (SW2-6)</td>
</tr>
<tr>
<td>T</td>
<td>7</td>
<td></td>
<td>+CASSETTE MOTOR OFF (SW2-7)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PA0</th>
<th>+KBD SCAN CODE 3</th>
<th>X'0063'</th>
<th>CMD/MODE REGISTER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>MODE REG VALUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X'99'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>* PA3 PA2 AMOUNT OF MEMORY</td>
</tr>
<tr>
<td>SW1-4</td>
<td>SW1-3 LOCATED ON SYS. BD.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>16K BYTES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>32K BYTES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>48K BYTES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>64K BYTES</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>* PA3</th>
<th>PA2</th>
<th>AMOUNT OF MEMORY LOCATED ON SYS. BD.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>16K BYTES</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>32K BYTES</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>48K BYTES</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>64K BYTES</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>** PA5</th>
<th>PA4</th>
<th>TYPE OF DISPLAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW1-6</td>
<td>SW1-5</td>
<td>COLOR CARD 40X25 (BW MODE)</td>
</tr>
<tr>
<td>0</td>
<td>RESERVED</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>COLOR CARD 80X25 (BW MODE)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>IBM MONOCHROME DISPLAY (80 X 25)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>** PA5</th>
<th>PA4</th>
<th>TYPE OF DISPLAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW1-6</td>
<td>SW1-5</td>
<td>COLOR CARD 40X25 (BW MODE)</td>
</tr>
<tr>
<td>0</td>
<td>RESERVED</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>COLOR CARD 80X25 (BW MODE)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>IBM MONOCHROME DISPLAY (80 X 25)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>*** PA7</th>
<th>PA6</th>
<th>NUMBER OF 5-1/4&quot; DRIVES IN SYSTEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW1-8</td>
<td>SW1-7</td>
<td>IBM MONOCHROME DISPLAY (80 X 25)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

NOTE: PA bit=0 implies switch "ON". PA bit=1 implies switch "OFF".
System Memory Map

<table>
<thead>
<tr>
<th>X'00000'</th>
<th>16 TO 64KB ON SYSTEM BOARD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>I/O CHANNEL ADDED MEM MAX 192KB</td>
</tr>
<tr>
<td></td>
<td>384K MEMORY FUTURE EXPANSION</td>
</tr>
<tr>
<td></td>
<td>128KB</td>
</tr>
<tr>
<td></td>
<td>EXPANSION MEMORY 216KB</td>
</tr>
<tr>
<td></td>
<td>40KB BASE SYSTEM ROM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>X'FFFFF'</th>
<th>265KB R/W MEMORY PRESENT SYSTEM MAX MEMORY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3/4 MEG MEMORY ADDRESS SPACE</td>
</tr>
<tr>
<td></td>
<td>FUTURE EXPANSION</td>
</tr>
<tr>
<td></td>
<td>128KB RESERVED GRAPHIC/DISPLAY BUFFER</td>
</tr>
<tr>
<td></td>
<td>256KB ROM ADDRESS SPACE</td>
</tr>
</tbody>
</table>

Figure 11. SYSTEM MEMORY MAP
System Memory Map (Increments of 16KB)

<table>
<thead>
<tr>
<th>START ADDRESS:</th>
<th>FUNCTION:</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECIMAL</td>
<td></td>
</tr>
<tr>
<td>HEX</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>16–64 KB READ/WRITE MEMORY</td>
</tr>
<tr>
<td>16K 04000</td>
<td>ON SYSTEM BOARD</td>
</tr>
<tr>
<td>32K 08000</td>
<td></td>
</tr>
<tr>
<td>48K 0C000</td>
<td></td>
</tr>
<tr>
<td>64K 10000</td>
<td>UP TO 192 KB</td>
</tr>
<tr>
<td>80K 14000</td>
<td>MEMORY IN I/O</td>
</tr>
<tr>
<td>96K 18000</td>
<td>CHANNEL</td>
</tr>
<tr>
<td>112K 1C000</td>
<td></td>
</tr>
<tr>
<td>128K 20000</td>
<td></td>
</tr>
<tr>
<td>144K 24000</td>
<td></td>
</tr>
<tr>
<td>160K 28000</td>
<td></td>
</tr>
<tr>
<td>178K 2C000</td>
<td></td>
</tr>
<tr>
<td>192K 30000</td>
<td></td>
</tr>
<tr>
<td>208K 34000</td>
<td></td>
</tr>
<tr>
<td>224K 38000</td>
<td></td>
</tr>
<tr>
<td>240K 3C000</td>
<td></td>
</tr>
<tr>
<td>256K 40000</td>
<td>384 KB FUTURE</td>
</tr>
<tr>
<td>272K 44000</td>
<td>R/W MEMORY EXPANSION</td>
</tr>
<tr>
<td>288K 48000</td>
<td>IN I/O CHANNEL</td>
</tr>
<tr>
<td>304K 4C000</td>
<td></td>
</tr>
<tr>
<td>320K 50000</td>
<td></td>
</tr>
<tr>
<td>336K 54000</td>
<td></td>
</tr>
<tr>
<td>352K 58000</td>
<td></td>
</tr>
<tr>
<td>368K 5C000</td>
<td></td>
</tr>
<tr>
<td>384K 60000</td>
<td></td>
</tr>
<tr>
<td>400K 64000</td>
<td></td>
</tr>
<tr>
<td>416K 68000</td>
<td></td>
</tr>
<tr>
<td>432K 6C000</td>
<td></td>
</tr>
<tr>
<td>448K 70000</td>
<td></td>
</tr>
<tr>
<td>464K 74000</td>
<td></td>
</tr>
<tr>
<td>480K 78000</td>
<td></td>
</tr>
<tr>
<td>496K 7C000</td>
<td></td>
</tr>
<tr>
<td>512K 80000</td>
<td></td>
</tr>
<tr>
<td>528K 84000</td>
<td></td>
</tr>
<tr>
<td>544K 88000</td>
<td></td>
</tr>
<tr>
<td>560K 8C000</td>
<td></td>
</tr>
<tr>
<td>576K 90000</td>
<td></td>
</tr>
<tr>
<td>592K 94000</td>
<td></td>
</tr>
<tr>
<td>608K 98000</td>
<td></td>
</tr>
<tr>
<td>624K 9C000</td>
<td></td>
</tr>
</tbody>
</table>

Figure 12. SYSTEM MEMORY MAP (INCREMENTS OF 16KB) (SHEET 1 OF 2)
## System Memory Map Cont.

<table>
<thead>
<tr>
<th>START ADDRESS: DECIMAL</th>
<th>FUNCTION:</th>
</tr>
</thead>
<tbody>
<tr>
<td>640K A0000</td>
<td>RESERVED</td>
</tr>
<tr>
<td>656K A4000</td>
<td>112 KB GRAPHICS/DISPLAY VIDEO BUFFER</td>
</tr>
<tr>
<td>672K A8000</td>
<td></td>
</tr>
<tr>
<td>688K AC000</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 BC000</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></td>
</tr>
<tr>
<td>816K CC000</td>
<td></td>
</tr>
<tr>
<td>832K D0000</td>
<td>192 KB MEMORY EXPANSION AREA</td>
</tr>
<tr>
<td>848K D4000</td>
<td></td>
</tr>
<tr>
<td>864K D8000</td>
<td></td>
</tr>
<tr>
<td>880K DC000</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 EC000</td>
<td></td>
</tr>
<tr>
<td>960K F0000</td>
<td>RESERVED</td>
</tr>
<tr>
<td>976K F4000</td>
<td>48 KB BASE SYSTEM ROM</td>
</tr>
<tr>
<td>992K F8000</td>
<td></td>
</tr>
<tr>
<td>1.008M FC000</td>
<td></td>
</tr>
</tbody>
</table>

Figure 12. SYSTEM MEMORY MAP (16KB) (SHEET 2)
System Board and Memory Expansion Switch Settings

On the following four pages are graphic illustrations of switch settings. These are necessary for the system to address components attached, and to specify the amount of memory installed both on the System Board and in the System Expansion Slots. Refer to the System Board Component Diagram (page 13) for DIP switch locations.

### SWITCH 1

<table>
<thead>
<tr>
<th>POSITION</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-7-8</td>
<td>NUMBER OF 5¼&quot; DISKETTE DRIVES INSTALLED; PAGE 2-29</td>
</tr>
<tr>
<td>2</td>
<td>UNUSED-MUST BE ON (RESERVED FOR CO-PROCESSOR)</td>
</tr>
<tr>
<td>3-4</td>
<td>AMOUNT OF MEMORY ON SYSTEM BOARD; PAGE 2-30</td>
</tr>
<tr>
<td>5-6</td>
<td>TYPE OF MONITOR YOU ARE USING; PAGE 2-29</td>
</tr>
</tbody>
</table>

### SWITCH 2

<table>
<thead>
<tr>
<th>POSITION</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-2-3-4</td>
<td>AMOUNT OF MEMORY OPTIONS INSTALLED; PAGE 2-30</td>
</tr>
<tr>
<td>5-6-7-8</td>
<td>ALWAYS IN THE OFF POSITION</td>
</tr>
</tbody>
</table>
5-1/4" Diskette Drives Switch Settings

Monitor Type Switch Settings

NOTE: SOME TELEVISIONS AND MONITORS OPERATED IN (80 x 25) MODE MAY HAVE CHARACTER LOSS.
System Board Memory Switch Settings

SWITCH 1

16KB

32KB

48KB

64KB

96 KB

128 KB

160 KB

192 KB

224 KB

256 KB

SWITCH 2

1 2 3 4 5 6 7 8

ON

1 2 3 4 5 6 7 8

ON

1 2 3 4 5 6 7 8

ON

1 2 3 4 5 6 7 8

ON

1 2 3 4 5 6 7 8

ON

1 2 3 4 5 6 7 8

ON

1 2 3 4 5 6 7 8

ON

1 2 3 4 5 6 7 8

ON

1 2 3 4 5 6 7 8

ON

1 2 3 4 5 6 7 8

ON
32/64KB Memory Expansion Option
Switch Settings

Note: Positions 6-7-8 must always be ON. The sequence shown below must be followed to allow the system to address the memory properly.
Power Supply

The system DC power supply is a 63.5 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 7 amps of +5 Vdc, ±5% 2 amps of +12Vdc, ±5% 300 ma of -5Vdc, ±10% and 250 ma of -12 Vdc, ±10%. All power levels are regulated with overvoltage and over current protection. The input is 120 Vac and fused. DC over-load or over-voltage conditions exist, the supply will automatically shut down until the condition is corrected. The supply is designed for continuous operation at 63.5 watts.

The System Board takes approximately 3 amps of +5 Vdc thus allowing approximately 4 amps of 5 Vdc for the adapters in the System Expansion Slots. The +12 Vdc power level is designed to power the two internal 5-1/4” Diskette Drives and the system’s dynamic memory. It is assumed that only one drive motor is active at a time. The −5 Vdc level is used for memory bias voltage and analog circuits in the diskette adapter phase lock loop. The +12 Vdc and −12 Vdc are used for powering the serial interface card EIA drivers and receivers for the Asynchronous Communications Adapter. All four power levels are bussed across the five System Expansion Slots and available for option adapter.

The IBM Monochrome Display is self-powered. However, the high resolution display receives its AC power from the System Unit power system. It is switched on and off with the power switch, which saves a wall outlet. The AC output for the display is a nonstandard connector, so only the AC high resolution Display can use this AC port.
The Power Supply is located at the right rear area of the System Unit. It supplies operating voltages to the System Board, IBM Monochrome Display, and provides two separate connections for power to the 5-1/4" Diskette Drives (if installed). The nominal power requirements and output voltages are listed on the following tables:

**Input Requirements**

**Voltage**

<table>
<thead>
<tr>
<th>VOLTAGE @ 60 Hz</th>
<th>NOMINAL</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vac</td>
<td>120</td>
<td>104</td>
<td>127</td>
</tr>
</tbody>
</table>

**Frequency**

60 Hz +/- .5 Hz

**Current**

2.5 AMPS MAX @ LOW LINE INPUT

**VOLTAGE OF 120 VAC 60 HZ**

**DC Output**

<table>
<thead>
<tr>
<th>VOLTAGE Vdc</th>
<th>CURRENT AMPS</th>
<th>REGULATION TOLERANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOMINAL</td>
<td>MIN</td>
<td>MAX</td>
</tr>
<tr>
<td>+ 5.0</td>
<td>2.3</td>
<td>7.0</td>
</tr>
<tr>
<td>- 5.0</td>
<td>0.0</td>
<td>0.3</td>
</tr>
<tr>
<td>+12.0</td>
<td>0.0</td>
<td>2.0</td>
</tr>
<tr>
<td>-12.0</td>
<td>0.0</td>
<td>0.25</td>
</tr>
</tbody>
</table>

**AC Output**

<table>
<thead>
<tr>
<th>VOLTAGE Vac</th>
<th>CURRENT AMPS</th>
<th>VOLTAGE LIMITS Vac</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOMINAL</td>
<td>MIN</td>
<td>MAX</td>
</tr>
</tbody>
</table>
| 120.0       | 0.0          | .75                | 101.0| 130.0
Power Supply Connectors and Pin Assignment

and location are shown below.

The power connector on the System Board is a 12 pin male connector.
Important Operating Characteristics

Over Voltage/Current Protection

**PRIMARY (INPUT)**

<table>
<thead>
<tr>
<th>VOLTAGE NOMINAL VAC</th>
<th>TYPE PROTECTION</th>
<th>RATING AMPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>120</td>
<td>60 Hz FUSE TYPE 2 SOC SD4</td>
<td>2 AMPS</td>
</tr>
</tbody>
</table>

Power On/Off Cycle: When the supply is turned off for a maximum of 5 seconds, and then turned on, the power good signal will be regenerated.

**Signal Requirements**

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

The Power Supply. Provides a power good signal out, to indicate the presence of the +/−5V and +/−12V outputs are above the sense level defined in the chart below, the power good signal is an up level (2.4V to 5.5V), TTL compatible and capable of sourcing 60 UA. When any of the four sensed output voltages is below its sense level voltage as defined in the chart below, the power good signal is down level (0V to 0.4V), TTL compatible and capable of sinking 500 UA. The power good signal (after all levels of the output voltage are good) has a turn on delay of 100 MS, but no greater than 500 MS.

The sense levels of the +/−5V and +/−12V outputs are:

<table>
<thead>
<tr>
<th>OUTPUT</th>
<th>MIN</th>
<th>SENSE VOLTAGE NOMINAL</th>
<th>MAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>+5V</td>
<td>+3.7</td>
<td>+4.0</td>
<td>+4.3</td>
</tr>
<tr>
<td>−5V</td>
<td>−3.7</td>
<td>−4.0</td>
<td>−4.3</td>
</tr>
<tr>
<td>+12V</td>
<td>+8.5</td>
<td>+9.6</td>
<td>+10.5</td>
</tr>
<tr>
<td>−12V</td>
<td>−8.5</td>
<td>−9.6</td>
<td>−10.5</td>
</tr>
</tbody>
</table>
IBM Monochrome Display and Parallel Printer Adapter

This adapter has dual functions. The first is to provide the interface to the IBM Monochrome Display. The second function is a parallel interface for the IBM 80 CPS Matrix Printer.

The monitor interface is designed around the Motorola 6845 CRT Controller module. There are 4K bytes of static memory on the card which are used for the display buffer. The memory is dual ported and may be accessed directly by the CPU. No parity is provided on the display buffer. A block diagram of the Monochrome Display function in on page 2-38.

The characteristics of the design are listed below:

- 80x25 Screen
- Direct Drive Output
- 9x14 Character Box
- 7x9 Character
- 18 Khz Monitor
- Character Attributes

The adapter supports 256 character codes. An 8K byte character generator contains the fonts for the character codes. The characters, values, keystrokes and screen characteristics are tabled in Appendix C. Of Characters, Keystrokes and Color.

Note: This Adapter when used with a display containing P39 Phosphor, will not support a light pen!

Parallel Interface Description

This topic is discussed in full on pages 2-65 through page 2-69.
Figure 14. IBM MONOCHROME DISPLAY ADAPTER BLOCK DIAGRAM
System Channel Interface

Lines Used
This card uses the address and data bus, memory and I/O read/write signals, reset, I/O Ready, I/O Clock, and IRQ7.

Loads
Where possible, only one “LS” load is on the signals present at the I/O slot. Some of the address bus lines have two “LS” loads. No signal has more than two “LS” loads.

Special Timing
At least one wait state will be inserted on all memory and I/O accesses from the CPU. The duration of the wait-state will vary because the CPU/monitor access is synchronized with the character clock on this adapter.

To insure proper initialization of the attachment, the first instruction issued to the card must be to set the high resolution bit of the monitor output Port 1. (OUT PORT 3B8 = 01H). A CPU access to this adapter must never occur if the high resolution bit is not set.

System configurations which have two display adapter cards must insure that both adapters are properly initialized after a power on reset. Damage to either display may occur if not properly initialized.

Data Rates
For the IBM Monochrome Display Adapter, two bytes are fetched from the display buffer in 553 ns providing a data rate of 1.8M bytes/second.

Interrupt and DMA Response Requirements
- The display buffer can be written into, or read from using DMA.
- The parallel interface uses the +IRQ7 line. Interrupt becomes active when the acknowledge input is low, and interrupts are enabled via the control port.
Modes of Operation

The IBM Monochrome Display and Printer Adapter supports 256 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.

```
<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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

CHARACTER CODE  EVEN ADDRESS (M)

```
<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>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>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
```

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 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>
```

2-40
Programming Considerations

Programming the 6845 CRT Controller

The following table summarizes the 6845 Internal Data Registers and their functions and parameters. For the IBM Monochrome Display, the values in the table must be programmed into the 6845 to insure proper initialization of the device.

**Table 2. 6845 INITIALIZATION PARAMETERS**

<table>
<thead>
<tr>
<th>REGISTER #</th>
<th>REGISTER FILE</th>
<th>PROGRAM UNIT</th>
<th>80x25 MONOCHROME</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0</td>
<td>HORIZONTAL TOTAL</td>
<td>CHARACTERS</td>
<td>61H</td>
</tr>
<tr>
<td>R1</td>
<td>HORIZONTAL DISPLAYED</td>
<td>CHARACTERS</td>
<td>50H</td>
</tr>
<tr>
<td>R2</td>
<td>HSYNC POSITION</td>
<td>CHARACTERS</td>
<td>52H</td>
</tr>
<tr>
<td>R3</td>
<td>HSYNC WIDTH</td>
<td>CHARACTERS</td>
<td>FH</td>
</tr>
<tr>
<td>R4</td>
<td>VERTICAL TOTAL</td>
<td>CHAR ROWS</td>
<td>19H</td>
</tr>
<tr>
<td>R5</td>
<td>VTOTAL ADJUST</td>
<td>SCAN LINE</td>
<td>6H</td>
</tr>
<tr>
<td>R6</td>
<td>VERTICAL DISPLAYED</td>
<td>CHAR ROW</td>
<td>19H</td>
</tr>
<tr>
<td>R7</td>
<td>VSYNC POSITION</td>
<td>CHAR ROW</td>
<td>19H</td>
</tr>
<tr>
<td>R8</td>
<td>INTERLACE MODE</td>
<td>---</td>
<td>02</td>
</tr>
<tr>
<td>R9</td>
<td>MAX SCAN LINE ADDRESS</td>
<td>SCAN LINE</td>
<td>DH</td>
</tr>
<tr>
<td>R10</td>
<td>CURSOR START</td>
<td>SCAN LINE</td>
<td>BH</td>
</tr>
<tr>
<td>R11</td>
<td>CURSOR END</td>
<td>SCAN LINE</td>
<td>CH</td>
</tr>
<tr>
<td>R12</td>
<td>START ADDRESS (H)</td>
<td>---</td>
<td>00H</td>
</tr>
<tr>
<td>R13</td>
<td>START ADDRESS (L)</td>
<td>---</td>
<td>00H</td>
</tr>
<tr>
<td>R14</td>
<td>CURSOR (H)</td>
<td>---</td>
<td>00H</td>
</tr>
<tr>
<td>R15</td>
<td>CURSOR (L)</td>
<td>---</td>
<td>00H</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>

Sequence of Events

The first command issued to this attachment must be to output to PORT 3B8, hex 01, to set high resolution mode. If the high resolution mode is not set, an infinite CPU wait-state will occur!

Memory Requirements

The attachment has 4K bytes of memory which is used for the display buffer. The memory supports one screen of 25 rows of 80 characters, plus a character attribute for each display character. No parity is provided on the memory. No system Read/Write memory is required for the monochrome adapter portion. The display buffer starts at address 'B0000'.
DMA Channels
The display buffer will support a DMA operation, however CPU wait-states will be inserted during DMA.

Interrupt Levels
Interrupt Level 7 is used on the parallel interface. Interrupts can be enabled or disabled via the Printer Control Port. The interrupt is a high level active signal.

I/O Address and Bit Map
The table below breaks down the functions of the I/O Address decode for the card. The I/O address decode is from ‘3B0’ through ‘3BF’. The bit assignment for each I/O address follows:

<table>
<thead>
<tr>
<th>I/O Address Function</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 to the high resolution Monochrome Display.

- CRT Output Port 1 (I/O Address ‘3B8’)

<table>
<thead>
<tr>
<th>Bit #</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>
CRT Status Port (I/O Address ‘3BA’)

<table>
<thead>
<tr>
<th>Bit</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>B/W Video</td>
</tr>
</tbody>
</table>

IBM Monochrome Display

The high resolution IBM Monochrome Display unit attaches to the System Unit via two cables of approximately 3’ (914 mm) in length. One cable is a signal cable which contains 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 ON/OFF switch to also control the display unit. An additional benefit is a reduction in the requirements for wall outlets to power the system. The monitor contains an 12” (305 mm) 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 table top or desk. The unit has both brightness and contrast adjustment controls on the front available to the operator.

Operating Characteristics

Screen

High persistence green phosphor (P 39) with an etched surface to reduce glare. Unit displays an 80 character by 25 line screen with a 9 dot wide by 14 dot tall character box.

Video Signal

Maximum video bandwidth of 16.27 Mhz.

Vertical Drive

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

Horizontal Drive

Positive level TTL compatible frequency, 18.432 Khz.
IBM Monochrome Direct Drive Interface and Pin Assignment

REAR PANEL

9 PIN "D" SHELL CONNECTOR

PARALLEL PRINTERADAPTER

At Standard TTL Levels

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

NOTE: Signal voltages are 0 - .6 Vdc at down level
+5 Vdc at high level
The Color/Graphics Monitor Adapter is designed to attach a wide variety of TV frequency monitors and TV sets (user-supplied RF modulator required for TVs). It is capable of operating in black and white or color, and provides three video interfaces; a composite video port, a direct drive port, and 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 A/N and APA modes. In A/N mode, the display can be operated in a 40x25 mode for low resolution monitor and TVs or 80x25 mode for high resolution monitors. In both modes, characters are defined in an 8x8 box and are 5x7 with one line of descender for lowercase (both uppercase and lowercase characters are supported in all modes). In black and white mode, the character attributes of Reverse Video, Blinking and Highlighting are available. In color mode, there are 16 foreground colors and 8 background colors available per character. In addition, blinking on a per character basis is available.

The adapter card contains 16KB of storage; thus, for a 40x25 screen, 1000 bytes are used to store character information and 1000 bytes are used for attribute/color information. This means that up to 8 pages of screens can be stored in the adapter memory. Similarly, in an 80x25 mode, 4 pages of display screen may be stored in the adapter. The full 16KB storage on the display adapter is directly addressable by the processor allowing maximum software flexibility in managing the screen. In A/N color modes, it is also possible to select the screen border color. One of 16 colors may be selected.
In APA mode, there are two resolutions available; 320x200 and 640x200. In the 320x200, 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 3 colors come from one of the two software selectable palettes. One palette contains red/green/brown, the other contains cyan/magenta/white.

The 640x200 mode is only available in black and white since the full 16KB of storage is used to define the on or off state of the pel.

The adapter operates in noninterlace mode at either 7 or 14 megahertz (Mhz) video bandwidth depending on the mode of operation selected.

In A/N mode, characters are formed from a ROM character generator. The character generator contains dot patterns for 256 characters. The character set contains the following major grouping of characters. Sixteen special characters for game support, 15 characters for support of word processing editing functions, the standard 96 ASCII graphic set, 48 characters to support foreign languages, 48 characters for business block graphics allowing drawing of charts, boxes and tables using single and double lines, 16 of the most often used Greek characters, and 15 of the most often used scientific notation characters.

The Color/Graphics Monitor Adapter function is packaged on a single card which fits into one of the five System Expansions Slots on the System Board. The direct drive and composite video ports are right-angle mounted connectors at the rear of the adapter and extend through the rear panel of the System Unit.

The display adapter is implemented using a Motorola 6845 CRT controller device. This adapter is highly programmable with respect to raster and character parameters. Thus, many additional modes are possible with clever programming of the adapter. A block diagram of the Color/Graphics Adapter is on the following page.
Figure 15. COLOR/GRAPHICS MONITOR ADAPTER BLOCK DIAGRAM
Major Components Definitions

Motorola 6845 CRT Controller
This device provides the necessary interface to drive a raster scan CRT.

Mode Set And Status Registers
This is a general purpose programmable I/O register. It has I/O points which may be individually programmed. Its function in this attachment is to provide mode selection (page 2-49 and 2-50) and color selection in the medium resolution color graphics mode (page 2-51.)

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

Character Generator
This attachment utilizes a ROM character generator. It consists of 8K bytes of storage which cannot be read/written under software control. This is a general purpose ROM character generator with three different character fonts. Two character fonts are used on this card (a 7x7 double dot and 5x7 single dot), selected by a card jumper. No jumper gives a 7x7 double dot, with a jumper a single dot font is selected.

Timing Generator
This block generates the timing signals used by the 6845 CRT controller and by the dynamic memory. It also resolves the CPU/graphic controller contentions for accessing the Display Buffer.

Composite Color Generator
The logic in this block generates base band video color information.
Modes of Operation

There are two basic modes of operation, 'Alphanumeric' and 'Graphics'. Each of these modes provide further options in both color and black-and-white. The following text describes each mode of operation.

Alphanumeric Mode

Alphanumeric Display Architecture

Every display character position is defined by two bytes in the regen buffer (part of display adapter, not system memory). Both the color and the black and white display adapter use this 2 byte character/attribute format.

<table>
<thead>
<tr>
<th>DISPLAY CHAR 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>

Attribute Byte Definition

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

<table>
<thead>
<tr>
<th>ATTRIBUTE FUNCTION</th>
<th>FG</th>
<th>BACKGROUND</th>
<th>FOREGROUND</th>
</tr>
</thead>
<tbody>
<tr>
<td>NORMAL</td>
<td>B</td>
<td>0 0 0 0</td>
<td>1 1 1 1</td>
</tr>
<tr>
<td>REVERSE VIDEO</td>
<td>B</td>
<td>1 1 1 1</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>NON DISPLAY (BLK)</td>
<td>B</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>NON DISPLAY (WHITE)</td>
<td>B</td>
<td>1 1 1 1</td>
<td>1 1 1 1</td>
</tr>
</tbody>
</table>

I = HIGH LIGHT FOREGROUND (CHAR)
B = BLINK FOREGROUND (CHAR)

Color TV

- Display up to 25 rows of 40 characters each
- Maximum of 256 characters
- Requires 2000 bytes of Read/Write Memory (on the adapter)
- 8x8 character box
- 7x7 double dotted characters (one descender)
- Character attributes (one for each character)
### Attribute Byte Definitions

<table>
<thead>
<tr>
<th>Character Code</th>
<th>Attribute Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVEN ADDRESS (M)</td>
<td>ODD ADDRESS (M+1)</td>
</tr>
</tbody>
</table>

**R:** Red  
**G:** Green  
**B:** Blue  
**I:** Intensity

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

- **Foreground Color**
- **Background Color**
- **Blinking**

**Note:** The starting address of the display buffer must be an even location.

**Color Monitor (with Direct Drive input capability)**

- Display up to 25 rows of 80 characters each
- Requires 4000 bytes of Read/Write Memory (on the adapter)
- Maximum of 256 character set
- 8x8 character box
- 7x7 character with one descender
- Same format for attributes as for color TV

**Note:** The starting address of the display buffer must be an even location.
IBM Monochrome Display Adapter Vs. Color/Graphics Adapter Attribute Relationship

Table 3. Monochrome Vs Color/Graphics Attributes

<table>
<thead>
<tr>
<th></th>
<th>ON THE MONOCHROME DISPLAY ADAPTER</th>
<th>ON THE COLOR/GRAPHIC DISPLAY ADAPTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>B R G B</td>
<td>CHAR. COLOR</td>
<td>BKGD. COLOR</td>
</tr>
<tr>
<td>FG BGD. FOREGROUND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NORMAL</td>
<td>B 0 0 0 1</td>
<td>1 1 1 1</td>
</tr>
<tr>
<td>RVV</td>
<td>B 1 1 1 1</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>NON DISP (BLK)</td>
<td>B 0 0 0 1</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>NON DISP (WHT)</td>
<td>B 1 1 1 1</td>
<td>1 1 1 1</td>
</tr>
</tbody>
</table>

ALL OTHER CODES
DEFINE FOREGROUND
BACKGROUND COLOR
COMBINATIONS

R G B *

- CODE WRITTEN WITH AN UNDERLINE
  ATTRIBUTE FOR THE IBM MONOCHROME DISPLAY
- WHEN EXECUTED ON A COLOR/GRAPHICS ADAPTER
- WILL RESULT IN A BLUE CHARACTER
- WHERE THE UNDERLINE ATTRIBUTES
  ARE ENCOUNTERED.
- CODE WRITTEN ON A COLOR/GRAPHICS ADAPTER
- WITH BLUE CHARACTERS, WILL BE
- DISPLAYED AS WHITE CHARACTERS
  ON BLACK BACKGROUND WITH A
  WHITE UNDERLINE ON THE MONOCHROME DISPLAY

* AN ADDITIONAL
8 COLOR (ACTUAL)
DIFFERENT SHADES
OF THE ABOVE)
ARE SELECTED BY
SETTING THE
(I) BIT

Note: Not all Monitors Recognize the (I) Bit

Table 4. Color/Graphics Modes

<table>
<thead>
<tr>
<th></th>
<th>HORIZONTAL</th>
<th>VERTICAL</th>
<th>NO OF COLORS (INCL. BACKGROUND COLOR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOW RES</td>
<td>160</td>
<td>100</td>
<td>16 (INCLUDES BLACK AND WHITE)</td>
</tr>
<tr>
<td>MED RES</td>
<td>320</td>
<td>200</td>
<td>4 COLORS: 1 OF 16 FOR BACKGROUND PLUS GREEN, RED, YELLOW OR CYAN, MAGENTA, WHITE</td>
</tr>
<tr>
<td>HIGH RES</td>
<td>640</td>
<td>200</td>
<td>B &amp; W ONLY</td>
</tr>
</tbody>
</table>
1. Low resolution color graphics (TV or monitor). (Note: This mode is not supported in ROM).
   - Up to 100 rows of 160 pels each (2x2)
   - 1 of 16 colors each pel specified by I, R, G and B
   - Requires 8000 byte of Read/Write Memory (on the adapter)
   - Memory mapped graphics (requires special memory map and set up to be defined later)

2. Medium resolution color graphics (TV or monitor)
   - Up to 200 rows of 320 pels each (1x1)
   - 1 out of 4 preselected colors in each box
   - Requires 16000 bytes of Read/Write Memory (on the adapter)
   - Memory mapped graphics
     4 pels/byte

FORMAT: \( \begin{array}{cccccccc}
7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
C1 & C0 & C1 & C0 & C1 & C0 & C1 & C0 \\
\end{array} \)

First display pel

- Graphics storage is organized in two banks of 8000 bytes each.

Graphics Storage Map

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>even scans (0, 2, 4, \ldots, 198) (8000 bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>#0000</td>
<td></td>
</tr>
<tr>
<td>#1F3F</td>
<td></td>
</tr>
<tr>
<td>#2000</td>
<td>odd scans (1, 3, 5, \ldots, 199) (8000 bytes)</td>
</tr>
<tr>
<td>#3F3F</td>
<td></td>
</tr>
</tbody>
</table>

Address #0000 contains pel information for upper left corner of display area.
Color selection is determined by the following logic:

C1 and C0 will select 4 of 16 preselected colors.

This color selection (palette) is preloaded in an I/O port.

```
C1  C0  CODE SELECT COLOR FOR DISPLAY
    POSITION
0  0  DOT TAKES ON COLOR OF 1 OF 16
    PRESELECTED BACKGROUND COLORS.
0  1  SELECT 1ST COLOR OF PRESELECT COLOR
    SET "1" OR "2"
1  0  SELECT 2ND COLOR OF PRESELECT COLOR
    SET "1" OR "2"
1  1  SELECT 3RD COLOR OF PRESELECT COLOR
    SET "1" OR "2"
```

The two color sets are:

<table>
<thead>
<tr>
<th>SET ONE</th>
<th>SET TWO</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLOR 1 - CYAN</td>
<td>COLOR 1 - GREEN</td>
</tr>
<tr>
<td>COLOR 2 - MAGENTA</td>
<td>COLOR 2 - RED</td>
</tr>
<tr>
<td>COLOR 3 - WHITE</td>
<td>COLOR 3 - BROWN</td>
</tr>
</tbody>
</table>

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

3. Black and white high resolution graphics (monitor)
   • Up to 200 rows of 640 pels each (1x1)
   • Black and white only
   • Requires 16000 bytes of Read/Write Memory (on the adapter)
   • Addressing and mapping is the same as for 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.
   • 8 pels/byte

```
7 6 5 4 3 2 1 0
[------------------]
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
          |
```
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 position in the buffer.

The CPU and the display control unit have equal access to the display buffer during all the operating modes except high resolution alphanumeric. During this mode, the CPU should access the display buffer during the vertical retrace time (if not, then the display will be affected with random patterns as the CPU is using the display buffer). The characters are displayed from a prestored “character generator” which contains the dot patterns of all the displayable characters.

In the graphics mode the displayed dots and colors are also fetched from the display buffer (up to 16K bytes). In the Color/Graphics Mode Section, the bit configuration for each graphics mode is explained.
Table 5. Summary of Available Colors

<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>Light Gray</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Dark 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>White</td>
</tr>
</tbody>
</table>

Note: "I" provides extra luminance (brightness) to each shade available. Resulting in the light colors listed above, except where the "I" bit is not recognized by some monitors.

Programming Considerations

Programming the 6845 CRT Controller

The 6845 has 19 internal registers which are used to define and control a raster scanned CRT display. One of these registers, the Address Register, is actually used as a pointer to the other 18 registers. It is a write only register which is loaded from the CPU by executing an OUT instruction to I/O address 3D4. The five least significant bits of the I/O bus are loaded into the Address Register.

In order to load any of the other 18 registers, the Address Register is first loaded with the necessary pointer and then the CPU may output a value to I/O address 3D5 in order to load the information in the preselected register.

The following table defines the values which must be loaded in 6845 Registers in order to control the different modes of operation supported by the attachment.
Table 6. 6845 Register Description

<table>
<thead>
<tr>
<th>ADDR REG.</th>
<th>REG. #</th>
<th>REGISTER TYPE</th>
<th>UNITS</th>
<th>I/O</th>
<th>40x25 ALPHA</th>
<th>80x25 ALPHA</th>
<th>GRAPHIC MODES</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>R0</td>
<td>Horizontal Total</td>
<td>Char.</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>Char.</td>
<td>Write Only</td>
<td>28</td>
<td>50</td>
<td>28</td>
</tr>
<tr>
<td>2</td>
<td>R2</td>
<td>Horiz. Sync Position</td>
<td>Char.</td>
<td>Write Only</td>
<td>2D</td>
<td>5A</td>
<td>2D</td>
</tr>
<tr>
<td>3</td>
<td>R3</td>
<td>Horiz. Sync Width</td>
<td>Char.</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>Char. 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>Char. Row</td>
<td>Write Only</td>
<td>19</td>
<td>19</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>R7</td>
<td>Vert. Sync Position</td>
<td>Char. Row</td>
<td>Write Only</td>
<td>1C</td>
<td>1C</td>
<td>70</td>
</tr>
<tr>
<td>8</td>
<td>R8</td>
<td>Interlace 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>Max. Scan Line Addr.</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 Addr. (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 Addr. (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 Addr. (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 Addr. (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.
Programming the Mode Control and Status Register

The following I/O devices are defined on the Color/Graphics Adapter.

<table>
<thead>
<tr>
<th>HEX ADDR.</th>
<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>FUNCTION OF REGISTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'3D8'</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>DO REG (MODE CONTROL)</td>
</tr>
<tr>
<td>X'3D9'</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>DO REG (COLOR SELECT)</td>
</tr>
<tr>
<td>X'3DA'</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>DI REG (STATUS)</td>
</tr>
<tr>
<td>X'3DB'</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>CLEAR LIGHT PEN LATCH</td>
</tr>
<tr>
<td>X'3DC'</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>PRE SET LIGHT PEN LATCH</td>
</tr>
<tr>
<td>X'3D0'</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Z</td>
<td>Z</td>
<td>6845 REGISTERS</td>
</tr>
<tr>
<td>X'3D1'</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Z</td>
<td>Z</td>
<td>6845 REGISTERS</td>
</tr>
<tr>
<td>X'3D0'</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Z</td>
<td>Z</td>
<td>6845 REGISTERS</td>
</tr>
<tr>
<td>X'3D1'</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Z</td>
<td>Z</td>
<td>6845 REGISTERS</td>
</tr>
</tbody>
</table>

Z = don’t care condition

Color Select Register

This is a 6 bit output only, register, it can not be read, its address is X'3D9' and can be written using the 8088 I/O OUT command.

The following is a description of the Register functions.

<table>
<thead>
<tr>
<th>Bit 0</th>
<th>B (BLUE) Border Color Select ALPH(A)/BACKGROUND</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 1</td>
<td>G (GREEN) Border Color Select ALPH(A)/BACKGROUND</td>
</tr>
<tr>
<td>Bit 2</td>
<td>R (RED) Border Color Select ALPH(A)/BACKGROUND</td>
</tr>
<tr>
<td>Bit 3</td>
<td>I Intensifies Border Color Select ALPH(A)/BACKGROUND IN 320 x 200</td>
</tr>
<tr>
<td>Bit 4</td>
<td>Select Alt Back Color Set For Alpha Color Modes</td>
</tr>
<tr>
<td>Bit 5</td>
<td>320 x 200 Color Set Select</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. Select the screens border color in 40x25 alpha mode. In graphics mode (medium resolution) 320 x 200 color, the screen background color (C0-C1) is selected by these bit settings.

Bit 4. This bit when set will select on alternate, intensified, set of background colors in the alpha mode.

Bit 5 is only used in the medium resolution color mode (320 x 200). It is used to select the active set of screen colors for the display.
When bit 5 is set to a "1" colors are determined as follows.
The C1 C0 Set selected are:
    0 0  Background as defined by Bit 0-3 of Port ‘3D9’
    0 1  Cyan
    1 0  Magenta
    1 1  White

When bit 5 is set to a "0" Colors are determined as follows.
The C0 C1 Set selected are:
    0 0  Background as defined by Bit 0-3 of Port ‘3D9’
    0 1  Green
    1 0  Red
    1 0  Yellow

Mode Select Register
This is a 6 bit output only register, it cannot be read. Its address is X‘3D8’. It can be written using the 8088 I/O OUT command.
The following is a description of the registers functions.

Bit 0

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>80 x 25 mode</td>
</tr>
<tr>
<td>1</td>
<td>Graphic Select</td>
</tr>
<tr>
<td>2</td>
<td>B &amp; W Select</td>
</tr>
<tr>
<td>3</td>
<td>Enable Video Signal</td>
</tr>
<tr>
<td>4</td>
<td>High Res 640 x 200 B &amp; W Mode</td>
</tr>
<tr>
<td>5</td>
<td>Change BACKGROUND INTENSITY to Blink Bit</td>
</tr>
<tr>
<td>6</td>
<td>Not Used</td>
</tr>
<tr>
<td>7</td>
<td>Not Used</td>
</tr>
</tbody>
</table>

Bit 0  Selects between 40 x 25 and 80 x 25 alpha mode, a “1” sets it to 80 x 25 mode.

Bit 1  Selects between ALPHA mode and 320 x 200 graphics mode, a “1” select 320 x 200 graphics mode.

Bit 2  Selects color or B & W mode, a “1” selects B & W.

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

Bit 4  When on, this bit selects the 640 x 200 B & W graphics mode. One color of 8 can be selected on direct drive sets in this mode by using register 3D9.
Bit 5  When on, this bit will change the character background intensity to the blinking attribute function for ALPHA 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>Bits</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>z</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>z</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>z</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>40 x 25 ALPHA B &amp; W</th>
</tr>
</thead>
<tbody>
<tr>
<td>40 x 25 ALPHA COLOR</td>
</tr>
<tr>
<td>80 x 25 ALPHA B &amp; W</td>
</tr>
<tr>
<td>80 x 25 ALPHA COLOR</td>
</tr>
<tr>
<td>320 x 200 B &amp; W GRAPHICS</td>
</tr>
<tr>
<td>320 x 200 COLOR GRAPHICS</td>
</tr>
<tr>
<td>640 x 200 B &amp; W GRAPHICS</td>
</tr>
</tbody>
</table>

* THE LOW RESOLUTION 160 x 100 MODE REQUIRES SPECIAL PROGRAMMING AND IS SET UP AS ALPHA MODE 40 x 25

**z** = don’t care condition

Status Register

The status register is a 4 bit read only register. Its address is X‘3DA’. 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 SW Made
Bit 3  Alpha Dots
Bit 4  Not Used
Bit 5  Not Used
Bit 6  Not Used
Bit 7  Not Used

Bit 0  This input 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 input has set the light pen trigger. This trigger is reset on power on and may also be cleared by doing an I/O OUT command to address X'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 the switch is on.

Bit 3  The ALPHA video output signal is readable in this status bit. Its purpose is to verify that video information is being generated for RAS purposes.

Sequence of Events
1. Determine mode of operation
2. Reset Video Enable bit
3. Program 6845 to select mode
4. Program mode/color select registers

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 Buffers address starts at X'B8000'.

Interrupt Level (Vertical Retrace)
Level 2
I/O Address and Bit Map
Read/Write Memory Address Space

- 01000: System Read/Write Memory
- B8000: Display Buffer (16K Bytes)
- BBFFF: Display Buffer (16K Bytes)
- C8FFF: 128K RESERVED REGEN AREA
## Color/Graphics Monitor Adapter Direct Drive, and Composite Interface Pin Assignment

### REAR PANEL

#### AT STANDARD TTL LEVELS

<table>
<thead>
<tr>
<th>Ground</th>
<th>1</th>
</tr>
</thead>
<tbody>
<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>

#### COLOR DIRECT DRIVE 9 PIN "D" SHELL CONNECTOR

#### COMPOSITE PHONO JACK

**HOOK-UP TO MONITORS**

<table>
<thead>
<tr>
<th>Video Monitor</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color/Graphics Composite Jack</td>
<td>2</td>
</tr>
</tbody>
</table>

**Composite Video Signal of approximately 1.5 Volts Peak to Peak Amplitude**

| Chassis Ground | 2 |
Color/Graphics Monitor Adapter
Auxiliary Video Connectors

RF Modulator Interface
- Light Pen Input 1
  (key) Not Used 2
- Light Pen Switch 3
  Chassis Ground 4
+ 5 Volts 5
+ 12 Volts 6

Light Pen Interface
- Light Pen Input 1
  (key) Not Used 2
- Light Pen Switch 3
  Chassis Ground 4
+ 5 Volts 5
+ 12 Volts 6

Color/Graphics Adapter
NOTES
Parallel 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 which 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 instructions. 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.

This function is packaged on an adapter which fits into any of the five System Expansion slots on the System Board. The input/output signals are made available at the back of the adapter via a right angle PCB mounted 25 PIN “D” type connector. This connector protrudes through the rear panel of the System Unit where a cable and shield 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 fault isolation determination 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 following page.
Figure 16. PARALLEL PRINTER ADAPTER BLOCK DIAGRAM
Programming Considerations

The Printer Adapter responds to 5 I/O instructions - 2 output and 3 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 CPU to read back the contents of the two latches. The third allows the CPU 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>Parallel Printer Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output to address 3BCH</td>
<td>Output to address 378H</td>
</tr>
<tr>
<td>Bit 7</td>
<td>Bit 3</td>
</tr>
<tr>
<td>Pin 9</td>
<td>Bit 2</td>
</tr>
<tr>
<td>Bit 6</td>
<td>Bit 1</td>
</tr>
<tr>
<td>Pin 8</td>
<td>Bit 0</td>
</tr>
<tr>
<td>Bit 5</td>
<td>Pin 5</td>
</tr>
<tr>
<td>Pin 7</td>
<td>Pin 4</td>
</tr>
<tr>
<td>Bit 4</td>
<td>Pin 3</td>
</tr>
<tr>
<td>Pin 6</td>
<td>Pin 2</td>
</tr>
</tbody>
</table>

This 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>Parallel Printer Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output to address 3BEH</td>
<td>Output to address 37AH</td>
</tr>
<tr>
<td>Bit 4</td>
<td>Bit 3</td>
</tr>
<tr>
<td>IRQ Enable</td>
<td>Bit 2</td>
</tr>
<tr>
<td>Pin 17</td>
<td>Bit 1</td>
</tr>
<tr>
<td>Enable</td>
<td>Bit 0</td>
</tr>
</tbody>
</table>

This instruction causes this latch to capture the five least significant bits of 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 1, the card will interrupt the CPU on the condition that Pin 10 transitions high to low.

These pins are driven by open collector drivers pulled to +5V through 4.7K OHM resistors. They can each sink approximately 7 ma and maintain 0.8 volts down level.

Note: For pin references, see Parallel Interface Connector Specifications, page 2-69.
This command presents the CPU with data present on the pins associated with the out to x’ ‘3BC’. This should normally reflect the exact value that was last written to x ‘3BC’. If an external device should be driving data on these pins (in violation of usage ground rules) at the time of an input, this data will be ‘or’ ed with the latch contents.

This command presents real time status to the CPU 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, 16, 17 and IRQ bit to be read by the CPU. In the absence of external drive applied to these pins, data read by the CPU will exactly match data last written to x’ ‘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 ‘or’ ed with data applied to the pins by the x’ ‘3BE’ latch.

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

Note: For pin references see Parallel Printer Adapter Interface Connector Specifications page 2-69.
**Parallel Printer Adapter**

**Interface Connector Specifications**

**REAR PANEL**

**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>AMP Pin No.</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>
IBM 80 CPS Matrix Printer

The printer is a self powered, standalone table top unit. It attaches to the System Unit via a parallel signal cable which is 6 feet in length. The unit obtains its AC power from a standard wall outlet (120 Vac). The printer is an 80 Character Per Second (CPS) bidirectional wire matrix device. It has a 9 wire head, allowing it to print characters in a 9x9 dot matrix. It can print in compressed mode 132 characters per line and in standard font, 80 characters per line. A large font also prints in 66 characters per line mode. The printer can print double size characters and double dotted characters. The printer prints the standard ASCII 96 character uppercase and lowercase character sets. In addition, a set of 64 special block graphic characters are available.

The printer can also accept commands setting the feed control desired for the application. Setting of 1 to 66 lines per page can be programmed and the lines per inch may be set to 5, 8, or 10. This printer attaches to the System Unit via the Parallel Printer Adapter or the combination Monochrome Display Adapter and Parallel Printer Adapter. The cable is a 25 lead shielded cable with a 25 pin “D” type connector at the System Unit end, and a 36 pin connector on the printer end.

Note: You may lose data anytime you are running a program with the printer off and attached to the System Unit.
Table 7. Printer Specifications

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PRINT METHOD:</td>
<td>Serial impact dot matrix</td>
</tr>
<tr>
<td>2</td>
<td>PRINT SPEED:</td>
<td>80 CPS</td>
</tr>
<tr>
<td>3</td>
<td>PRINT DIRECTION:</td>
<td>Bidirectional with logical seeking</td>
</tr>
<tr>
<td>4</td>
<td>NUMBER OF PINS IN HEAD:</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>LINE SPACING:</td>
<td>4.23 mm (1/6&quot;) or programmable</td>
</tr>
<tr>
<td>6</td>
<td>PRINTING CHARACTERISTICS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Matrix</td>
<td>9 x 9</td>
</tr>
<tr>
<td></td>
<td>Character Set</td>
<td>Full 96-character ASCII with decoders, plus 9 international characters/symbols</td>
</tr>
<tr>
<td></td>
<td>Graphic Character</td>
<td>64 block characters</td>
</tr>
<tr>
<td>7</td>
<td>PRINTING SIZES</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Characters per inch per line</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Normal: 50 characters per inch per line</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Enlarged: 25 characters per inch per line</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Condensed: 12 characters per inch per line</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Condensed Enlarged: 8 characters per inch per line</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MEDIA HANDLING</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Paper Feed</td>
<td>Adjustable sprocket pin feed</td>
</tr>
<tr>
<td></td>
<td>Paper Width Range</td>
<td>101.6 mm (4&quot;) to 254 mm (10&quot;)</td>
</tr>
<tr>
<td></td>
<td>Copies</td>
<td>One original plus two carbon copies (total thickness not to exceed 0.3 mm (0.012&quot;)</td>
</tr>
<tr>
<td></td>
<td>Paper Path</td>
<td>Rear</td>
</tr>
<tr>
<td>9</td>
<td>INTERFACES</td>
<td>Parallel 8-bit Data &amp; Control Lines</td>
</tr>
<tr>
<td>10</td>
<td>INKED RIBBON</td>
<td>Black</td>
</tr>
<tr>
<td></td>
<td>Color: Cartridge</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>ENVIRONMENTAL CONDITIONS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Operating Temperature Range:</td>
<td>5 to 35°C (41 to 95°F)</td>
</tr>
<tr>
<td></td>
<td>Operating Humidity</td>
<td>10 to 80% non-condensing</td>
</tr>
<tr>
<td>12</td>
<td>POWER REQUIREMENT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Voltage: 120VAC, 60 Hz</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Current: 1 Amp maximum</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Power Consumption: 100 VA maximum</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>PHYSICAL CHARACTERISTICS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Height: 107 mm (4.2&quot;)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Width: 374 mm (14.7&quot;)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Depth: 305 mm (12.0&quot;)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Weight: 5.5 kg (12 lbs.)</td>
<td></td>
</tr>
</tbody>
</table>
Setting The DIP Switches

There are two DIP switches on the control circuit board. In order to suit 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 Table 8 (DIP Switch 1) and Table 9 (DIP Switch 2).

Figure 17. LOCATION OF PRINTER DIP SWITCHES

Table 8. Functions and Conditions of DIP Switch 1

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Function</th>
<th>ON</th>
<th>OFF</th>
<th>Factory-set Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Not applicable</td>
<td></td>
<td></td>
<td>ON</td>
</tr>
<tr>
<td>2</td>
<td>CR</td>
<td>Print only</td>
<td>Print only Print &amp; line feed</td>
<td>ON</td>
</tr>
<tr>
<td>3</td>
<td>Buffer full</td>
<td>Print only</td>
<td>Print only Print &amp; line feed</td>
<td>ON</td>
</tr>
<tr>
<td>4</td>
<td>Cancel code</td>
<td>Invalid</td>
<td>Valid</td>
<td>OFF</td>
</tr>
<tr>
<td>5</td>
<td>Delete code</td>
<td>Invalid</td>
<td>Valid</td>
<td>ON</td>
</tr>
<tr>
<td>6</td>
<td>Error Buzzer</td>
<td>Sounds</td>
<td>Does not sound</td>
<td>ON</td>
</tr>
<tr>
<td>7</td>
<td>Character generator (Graphic pattern select)</td>
<td>N.A.</td>
<td>Graphic patterns select</td>
<td>OFF</td>
</tr>
<tr>
<td>8</td>
<td>SLCT IN signal</td>
<td>Fixed</td>
<td>Not fixed</td>
<td>ON</td>
</tr>
</tbody>
</table>

Fixed internally

Not fixed internally
Table 9. Functions and Conditions of DIP Switch 2

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Function</th>
<th>ON</th>
<th>OFF</th>
<th>Factory-set Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Not applicable</td>
<td></td>
<td></td>
<td>ON</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td></td>
<td></td>
<td>ON</td>
</tr>
<tr>
<td>3</td>
<td>AUTO FEED</td>
<td>Fixed internally</td>
<td>Not fixed internally</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>XT signal</td>
<td>Fixed</td>
<td>Not fixed</td>
<td>OFF</td>
</tr>
<tr>
<td>4</td>
<td>Coding table select</td>
<td>N.A.</td>
<td>Standard</td>
<td>OFF</td>
</tr>
</tbody>
</table>

Parallel Interface Description

(1) Specifications
(a) Data transfer rate: 1000 CPS (max.)
(b) Synchronization: By externally supplied STROBE pulses.
(c) Handshaking: ACKNLG or BUSY signals.
(d) Logic level: Input data and all interface control signals are compatible with the TTL level.

(2) Connector
Plug: 57-30360 (AMPHENOL)

(3) Connector pin assignment and descriptions of signals.
Connector pin assignment and descriptions of respective interface signals are provided in Table (10) page 2-74.
Table 10. Connector Pin Assignment and Descriptions of Interface Signals

<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 0.5μs at receiving terminal. The signal level is normally “HIGH”; read-in of data is performed at the “LOW” 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 “HIGH” level when data is logical “1” and “LOW” when logical “0”.</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>Approx. 5μs pulse. “LOW” indicates that data has been received and that 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 “HIGH” signal indicates that the printer cannot receive data. The signal becomes “High” in the following cases: 1. During data entry 2. During printing operation 3. In OFF-LINE state 4. During printer error status.</td>
</tr>
<tr>
<td>Signal Pin No.</td>
<td>Return Pin No.</td>
<td>Signal</td>
<td>Direction</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>--------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>12</td>
<td>30</td>
<td>PE</td>
<td>Out</td>
<td>A “HIGH” 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 FEED XT</td>
<td>In</td>
<td>With this signal being at “LOW” level, the paper is automatically fed one line after printing. (The signal level can be fixed to “LOW” with DIP SW pin 2-3 provided on the control circuit board.)</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. Not used.</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>TWISTED-PAIR RETURN signal GND level.</td>
</tr>
<tr>
<td>31</td>
<td></td>
<td>INIT</td>
<td>In</td>
<td>When the level of this signal becomes “LOW” the printer controller is reset to its initial state and the print buffer is cleared. This signal is normally at “HIGH” level, and its pulse width must be more than 50µs at the receiving terminal.</td>
</tr>
</tbody>
</table>
Table 10. Connector Pin Assignment and Descriptions of Interface Signals (cont.)

<table>
<thead>
<tr>
<th>Signal Pin No.</th>
<th>Return Pin No.</th>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| 32             |                | ERROR  | Out       | The level of this signal becomes “LOW” when the printer is in—  
|                |                |        |           | 1. PAPER END state  
|                |                |        |           | 2. OFF-LINE state  
|                |                |        |           | 3. Error state     |
| 33             | —              | GND    | —         | Same as with Pin No.  
|                |                |        |           | 19 to 30.         |
| 34             | —              | NC     | —         | Not used.         |
| 35             |                |        |           | Pulled up to +5V through 4.7 KΩ resistance. |
| 36             | —              | SLCT IN| In        | Data entry to the printer is possible only when the level of this signal is “LOW”. (Internal fixing can be carried out with DIP SW 1-8. The condition at the time of shipment is set “LOW” for this signal.) |

**NOTES**

1: “Direction” refers to the direction of signal flow as viewed from the printer.

2: “Return” denotes “TWISTED PAIR RETURN” and is to be connected at signal ground level.

As to the wiring for 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 the 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”.)
Data transfer sequence

Fig. 17 shows the sequence for data transmission.

Figure 18. PARALLEL INTERFACE TIMING DIAGRAM
ASCII Coding Table

Table 11 shows all available codes when the Printer is set for operation with standard coding by setting the DIP switch pin 2-4 to the OFF position. This DIP switch pin is factory-set to the OFF position.

<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>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<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>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>0000</td>
<td>NUL</td>
<td>0001</td>
<td>DC1</td>
<td>0010</td>
<td>DC2</td>
<td>0011</td>
<td>DC3</td>
<td>0100</td>
<td>DC4</td>
<td>0101</td>
<td>BEL</td>
<td>0110</td>
<td>HT</td>
<td>0111</td>
<td>CAN</td>
</tr>
<tr>
<td>0100</td>
<td>LF</td>
<td>0101</td>
<td>FF</td>
<td>0110</td>
<td>CR</td>
<td>0111</td>
<td>SO</td>
<td>1000</td>
<td>VT</td>
<td>1001</td>
<td>FF</td>
<td>1010</td>
<td>LF</td>
<td>1011</td>
<td>VT</td>
</tr>
<tr>
<td>1100</td>
<td>ESC</td>
<td>1101</td>
<td>CR</td>
<td>1110</td>
<td>CR</td>
<td>1111</td>
<td>SI</td>
<td>2-78</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ASCII Control Codes

Control Codes

Various kinds of control codes are contained in Table 11. These control codes are recognized by the printer and perform specified functions upon receipt of these codes. The following are descriptions of respective control codes.

(1) CR (Carriage Return)
When the CR code is transmitted to the print buffer, all data stored in the print buffer is printed.
(When AUTO FEED XT (Pin No. 14) is at “LOW” level or DIP switch pin 2-2 is ON, the paper is advanced one line automatically after printing.)
Note: When 80 columns of print data (including spaces) are continuously received and the following data is valid and printable, the Printer automatically begins to print the data stored in the print buffer. In this case, if AUTO FEED XT is at “LOW” level or DIP switch pin 2-3 is ON, the paper is advanced one line after printing.

(2) LF (Line Feed)
When the LF code is input, all data in the print buffer is printed and the paper is advanced one line.
Note: If no data precedes the LF code, or if all preceding data is “SPACE”, only paper feeding is performed.
For example, if the data is transferred in the order of DATA→CR→LF, DATA will be printed by the CR code, and when the Printer receives the LF code, it only carries out one line feed.

(3) VT (Vertical Tab)
When the VT code is input, all data preceding this code is printed. And the paper is advanced to the line position set by “ESC B” (described later). If no vertical tab position is set by ESC B, the VT code behaves like the LF code. Therefore, the paper is advanced one line after printing.

(4) FF (Form Feed)
The FF code carries out the printing of all data stored in the print buffer and advances the paper to the next predetermined Top of Form position. The Top of Form is determined when the POWR switch is turned on or the INIT signal is applied. If the form length per page is not set by “ESC C+n”, it is regarded as 66 or 72 lines.
Note: The form length of 72 lines per page is applicable to only the version marked with identifier code “M72” on the rear side of the lower case of the Printer. This code always initializes the printing of the data stored in the print buffer.

(5) SO (Shift Out)
When the SO code is input, all data that follows it in the same line will be printed out in enlarged (double width) characters. This code is cancelled by the printing operation or the input of “DC 4” code and can be input at any column position on a line. Therefore, normal size and enlarged characters can be mixed on the same line.

1. [DATA] ABC SO DEF DC 4 GHI CR LF
   [PRINT] ABCDEFGHI
2. [DATA] ABCD SO EFGH CR LF IJLK SO MNOP CR LF
   [PRINT] ABCDEFGHIKLMNOP

(6) SI (Shift In)
When the SI code is input, all data that follows it will be printed out in condensed characters. This code is cancelled by the input of “DC 2” code. The SI code can be input at any column position on a line, but all characters/symbols on the line containing SI code are printed out in condensed characters. When printing condensed characters, the data capacity of the print buffer will become 132 columns per line.
When the SO code is received after the input of the SI code, condensed enlarged characters (double width of condensed characters) can be printed. This condition is cancelled by “DC 4” code, and the character size returns to “condensed”.

1. [DATA] SI ABCDEFGHIJKLMNOP
   [PRINT] ABCDEFGHIJKLMNOP
2. [DATA] ABC SI DEF SO GHIJKLMNOP CR LF
   [PRINT] ABCDEFGHIJKLMNOP

(7) DC 4 (Device Control 4)
The DC 4 code cancels the SO mode.

[DATA] SI ABCDEF SO GHI DC 4 JKLMCR LF
[PRINT] ABCDEFGHIJKLMNOP
(8) **DC 2 (Device Control 2)**
The DC 2 code cancels the SI mode.

<table>
<thead>
<tr>
<th>DATA</th>
<th>SI ABCDEFG</th>
<th>SO GHI</th>
<th>CR LF</th>
<th>DC 2</th>
<th>JKL MN</th>
<th>CR LF</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRINT</td>
<td>ABCDEFGHI</td>
<td>CR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>JKL MN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(9) **HT (Horizontal Tab)**
The HT code carries out the horizontal tabulation. If there is no tab position set, this code is ignored. The tab stop positions are set by “ESC D+n” (described later).

(10) **CAN (Cancel)**
Upon the input of the CAN code, all data previously stored in the print buffer is cancelled. Therefore, this code is regarded as the print buffer clear command. This code clears the print buffer, but control codes (Excluding the SO code) are still valid even if the CAN code is transferred. The validity or invalidity of the CAN code is selectable by the DIP switch pin 1-4 on the control circuit board.

(11) **DEL (Delete)**
This code functions the same as the CAN code. The validity or invalidity of the DEL code is selectable by the DIP switch pin 1-5 on the control circuit board.

(12) **DC 1 (Device Control 1)**
The DC 1 code places the Printer in the Selected state. With the Printer in the Selected state, if the DC 1 code is input during data transfer, all data stored before the DC 1 code is ignored.

(13) **DC 3 (Device Control 3)**
The DC 3 code places the Printer in the Deselected state. In other words, it disables the Printer to receive data. Once the Printer is put in the Deselected state by the DC 3 code, the Printer will not revert to the Selected state unless the DC 1 code is input again.

**Note:** When the DC 1 and DC 3 codes are used, DIP switch pin 1-8 should be in the “OFF” position.

1. [DATA] DC 1 AAAAA DC 3 BBB BB DC 1 CCCCC CR LF
   [PRINT] AAAAAACCCCC
2. [DATA] AAAAA DC 1 BBB BB DC 3 CCCCC DC 1 CR LF
   [PRINT] BBB BB
Relations among the ON LINE switch, SLCT IN signal, DC1/DC3 code and interface signals are shown in Table 12 below.

Table 12. DC1/DC3 And Data Entry

<table>
<thead>
<tr>
<th>ON LINE SWITCH</th>
<th>SLCT IN</th>
<th>DC 1/DC 3</th>
<th>ERROR</th>
<th>BUSY</th>
<th>ACKNLG</th>
<th>SLCT</th>
<th>DATA ENTRY</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF-LINE</td>
<td>HIGH/LOW</td>
<td>DC 1/DC 3</td>
<td>LOW</td>
<td>HIGH</td>
<td>Not Generated</td>
<td>LOW</td>
<td>Impossible</td>
</tr>
<tr>
<td></td>
<td>HIGH</td>
<td>DC 1</td>
<td>HIGH</td>
<td>LOW/ HIGH Generated</td>
<td>HIGH</td>
<td>Possible (Normal entry)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LOW</td>
<td>DC 1/DC 3</td>
<td>HIGH</td>
<td>LOW/ HIGH Generated</td>
<td>HIGH</td>
<td>Possible (See Note 1.)</td>
<td></td>
</tr>
</tbody>
</table>

NOTES

1. In Table 12, it is assumed that as soon as the Printer receives data, it sends back the ACKNLG signal, though this data is not stored in the print buffer. In this status, the Printer is waiting for the DC 1 code for normal entry.
2. The DC 1/DC 3 code is valid under the condition that the DIP switch pin 1-8 is OFF, namely, the level of SLCT IN at the pin No. 36 of the interface connector is “HIGH”. When SLCT IN is “LOW”, the DC 1/DC 3 code is not valid.

(14) NUL (Null)
The NUL code is regarded as the termination for tabulation setting sequence (described in detail later).

(15) BEL (Bell)
When the BEL code is input, the buzzer sounds for about 3 seconds.

(16) Escape (ESC) control
(a) Escape numerical control
   Input of an “ESC” code followed by an ASCII numeric code permits each of the following functions to be performed.
   1) ESC 0 (Escape 0)
      Receipt of an “ESC” followed by ASCII code “0” causes the line spacing to be set at 1/8 inch. Input of the ESC 2 code or INIT signal to the interface connector or turning the power off and on again causes the line spacing to return to 1/6 inch.
   2) ESC 1 (Escape 1)
      Receipt of an “ESC” followed by ASCII code “1” causes the line spacing to be set at 7/72 inch. Input of the ESC 2 code or INIT signal to the interface connector or turning the power off and on again causes the line spacing to return to 1/6 inch.
3) ESC 2 (Escape 2)
Receipt of an “ESC” followed by ASCII code “2” causes the line spacing to be set at 1/6 inch. When the POWER switch is turned on, the line spacing is set at initial 1/6 inch. The ESC 2 code is also a command to execute “ESC A+n” modes (described later).

4) ESC 8 (Escape 8)
The ESC 8 code makes it possible to transmit data even if there is no paper in the Printer. This code should be transmitted before the Printer runs out of paper. After transmitting this code, when the Printer runs out of paper, the PE signal of the interface connector turns to High level; the ERROR signal remains at High level.

5) ESC 9 (Escape 9)
This code cancels the ESC 8 condition. When the power is turned on, the Printer is initialized into ESC 9 status. Therefore, the Printer cannot receive data when there is no paper.

6) ESC SI
This code functions the same as “SI”.

7) ESC SO
This code functions the same as “SO”.

(b) ESC alphabetic control
Receipt of an “ESC” code followed by ASCII code “X”(alphabetic code) permits each of the following functions to be performed.

Note: “n” represents a 7-bit binary number, and the most significant bit is not treated as data. “+” is inserted for the purpose of legibility only, and should not be input in actual operation.

1) ESC A+n
This code specifies the amount of line spacing in the Line Feed 1≤<n>10≤85 (Decimal): “n” is a binary number. “n”=1 is equivalent to 1/72 inch paper advancement. Since the distance between any two dot wires of the print head is 1/72 inch, any line spacing in increments proportional to the distance between the dot wires can be established.
The ESC A code is the command only to store spacing data into the memory. In other words, even if spacing data was transferred into the memory, the Printer does not actually carry out the line spacing in accordance with the spacing data. To execute the line spacing in accordance with the stored data, the ESC 2 code should be followed. Namely, the ESC 2 code is considered as the execution command for the line spacing.

Note: <How to input “n”>
When “n” is actually transferred to the Printer as data, it is transferred in the form of a 7-bit binary number.
In case of “ESC A+24”, actual output to the Printer is performed as $<1B>H<41>H<18>H$ in hexadecimal code.

2) ESC B+n1+n2+nk+NUL
(1≤<n>10≤66, 1≤k≤64, nk≤nk+1)
This code specifies the vertical tab stop positions. The first 64 valid tab stops per page are recognized in the Printer; subsequent tab stops are ignored.
A tab stop set at a line exceeding the form length is ignored. Tab stop numbers must be received in incremental numerical order. To execute predetermined tab stop positions, the VT code should be input. Once vertical tab stops are established, the data will be valid until new tab stops are specified. If no tab stop is set, the VT code
behaves like the LF code. Therefore, the paper is advanced one line after printing.

Receipt of “ESC B” code causes the Printer to accept the following codes as tab stop line numbers until the NUL code is input. The lack of the NUL code will cause incorrect data printout.

The form length must be set by “ESC C+n” code prior to setting tab stops.

Input of “ESC B” code followed by only the NUL code cancels predetermined tab stops.

3) ESC C+n (1≤<n>10≤66)
This code specifies the form length per page. The form length is determined by the number of lines (=“n”). The amount of a line spacing at this point is a predetermined numerical value by “ESC A+n”. When the form length is not programmed, one page is assumed at 66 or 72 lines. Prior to setting the vertical tab position, the form length should be set.

4) ESC D+n1+n2+. . . . . +nk+NUL
(1≤<n>10≤127,k≤112)
This code specifies the horizontal tab stop positions. The first 112 tab stops per line are recognized in the Printer, and subsequent tab stops are ignored. Tab stop numbers must be received in incremental numerical order.

If a tab stop position of higher value than 80 is received in normal character printing mode, all horizontal tab functions after 80 columns are ignored.

To execute tab stop positions, the HT code should be input. The HT code is ignored when the horizontal tab position has not been programmed.
The NUL code should be input as the command for the termination of the tab set sequence, and the lack of this code will cause incorrect data printout.

1. In case of 5th, 10th and 21st columns.

\[
\text{DATA} \quad \text{ESC D} \quad <S> \quad H \quad <A> \quad H \quad <1S> \quad H \quad \text{NUL} \quad \text{ABC} \quad \text{HT} \quad \text{DEF} \quad \text{HT} \quad \text{GHI} \quad \text{HT} \quad \text{JKL} \\
\text{PRINT} \quad \text{ABC} \quad \text{DEF} \quad \text{GHI} \quad \text{JKL}
\]

2. In case of lack of stop position.

\[
\text{DATA} \quad \text{ESC D} \quad <S> \quad H \quad <A> \quad H \quad \text{NUL} \quad \text{ABC} \quad \text{HT} \quad \text{DEF} \quad \text{HT} \quad \text{GHI} \quad \text{HT} \quad \text{JKL} \\
\text{PRINT} \quad \text{ABC} \quad \text{DEF} \quad \text{GHIJKL}
\]

3. In case of character data transferring over next tab stop.

\[
\text{DATA} \quad \text{ESC D} \quad <S> \quad H \quad <A> \quad H \quad <1S> \quad H \quad \text{NUL} \quad \text{ABCDEF} \quad \text{HT} \quad \text{GHI} \quad \text{HT} \quad \text{JKL} \quad \text{CR} \quad \text{LF} \\
\text{PRINT} \quad \text{ABCDEF} \quad \text{GHI} \quad \text{JKL}
\]

4. In case of transferring two HT codes at a time.

\[
\text{DATA} \quad \text{ESC D} \quad <S> \quad H \quad <A> \quad H \quad <1S> \quad H \quad \text{NUL} \quad \text{ABCD} \quad \text{HT} \quad \text{SPACE} \quad \text{HT} \quad \text{EFGH} \quad \text{CR} \quad \text{LF} \\
\text{PRINT} \quad \text{ABCD} \quad \text{EFGH}
\]

5) **ESC E**

The ESC E code causes the Printer to print emphasized characters. Emphasized printing gives the character a stronger impression on the paper.

This code can be input in any column position on a line.

The speed of the head carriage reduces to 40 CPS while printing emphasized characters.

\[
1. \quad \text{DATA} \quad \text{ESC E} \quad \text{ABCDEFGHI} \quad \text{CR} \quad \text{LF} \\
\text{PRINT} \quad \text{ABCDEFGHI}
\]

\[
2. \quad \text{DATA} \quad \text{SO} \quad \text{ESC E} \quad \text{ABCDEFGHI} \quad \text{CR} \quad \text{LF} \\
\text{PRINT} \quad \text{ABCDEFGHI}
\]

6) **ESC F**

The ESC F code cancels the emphasized printing mode.

7) **ESC G**

The ESC G code causes the Printer to perform the double printing. Double printing is carried out in the following manner:

a) A character is printed.

b) The paper is advanced by 1/216 inch.

c) The print head prints the same character again.

In this way, the character becomes bold.
8) ESC H
The ESC H code cancels the double printing mode.
5 1/4-Inch Diskette Drive Adapter

The System Unit has space and power for one or two 5-1/4” Diskette Drives. The drives are soft sectored, single sided, with 40 tracks. They are Modified Frequency Modulation (MFM) coded in 512 byte sectors, giving a formatted capacity of 163,840 bytes per drive. They have a track to track access time of 8 ms and a motor start time of 500 ms.

The 5-1/4” Diskette Drive Adapter fits in one of the System Board’s five System Expansion Slots. It attaches to the two drives via an internal daisy chained flat cable which connects to one end of the drive adapter. The adapter has a second connector on the other end which extends through the rear panel of the System Unit. This connector contains the signals for two additional external drives, thus the 5-1/4” Diskette Drive Adapter is capable of attaching four 5-1/4” drives, two internal, and two external.

The adapter is designed for double density MFM coded drives and uses write precompensation with an analog phase locked loop for clock and data recovery. The adapter is a general purpose device using the NEC \( \mu \)PD765 compatible controller. Thus the drive parameters are programmable. In addition, the attachment supports the drive’s write protect feature.

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 is also used to indicate operation complete and status condition requiring processor attention.

In general, the 5-1/4” Diskette Drive Adapter presents a high-level command interface to software I/O drivers. A block diagram of the 5-1/4” Diskette Drive Adapter is on the following page.
Figure 19. 5¼” 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 a NEC μPD765 or equivalent Floppy Disk Controller (FDC).

In the following description, drives numbers 0-3 are equivalent to drives A-D respectively.

Digital Output Register (DOR)

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 1</th>
<th>Drive</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>B</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>C</td>
</tr>
<tr>
<td>1</td>
<td>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 (FDC)

The following is a brief summary of the registers and commands implemented by the FDC.

The FDC contains two registers which 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 FDD status information. Data bytes are read out of, or written into, the Data...
Register in order to program or obtain the 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 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 FDD</td>
<td>FDD A Busy</td>
<td>DAB</td>
<td>FDD number 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/</td>
<td>DIO</td>
<td>Indicates direction of data transfer between FDC and Processor. If DIO = “1”, then transfer is from FDC Data Register to the Processor. If DIO = “0”, then transfer is from the Processor to FDC Data Register. Indicates Data Register is ready to send or receive data to or from the Processor. Both bits DIO and RQM should be used to perform the handshaking functions of “ready” and “direction” to the processor.</td>
</tr>
<tr>
<td>DB7</td>
<td>Request for Master</td>
<td>RQM</td>
<td>-</td>
</tr>
</tbody>
</table>
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 are made available to the processor.
Programming Considerations

Table 13. Symbol Descriptions

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 which 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 which users are going to read out or write into 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, and 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>
<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 Head.</td>
</tr>
</tbody>
</table>
### Table 13. Symbol Descriptions (continued)

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<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 which 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 register (DOR) p 2-91.</td>
</tr>
<tr>
<td>US1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Command Summary

0 indicates ‘logical 0’ for that bit, 1 means ‘logical 1’, X means ‘don’t care’.

### READ DATA

<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>SK</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</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>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>H</td>
<td>R</td>
<td>N</td>
<td>EOT</td>
<td>GPL</td>
<td>DTL</td>
<td></td>
<td>Data-transfer between the FDD and main-system</td>
</tr>
<tr>
<td>Execution</td>
<td>R</td>
<td>ST 0</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 1</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>ST 2</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>R</td>
<td>C</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>R</td>
<td>H</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>R</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>R</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Status information after Command execution</td>
</tr>
</tbody>
</table>

### READ DELETED DATA

<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>SK</td>
<td>0</td>
<td>0</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>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>H</td>
<td>R</td>
<td>N</td>
<td>EOT</td>
<td>GPL</td>
<td>DTL</td>
<td></td>
<td>Data-transfer between the FDD and main-system</td>
</tr>
<tr>
<td>Execution</td>
<td>R</td>
<td>ST 0</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 1</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>ST 2</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>R</td>
<td>C</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>R</td>
<td>H</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>R</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>R</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Status information after Command execution</td>
</tr>
</tbody>
</table>

### WRITE DATA

<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>H</td>
<td>R</td>
<td>N</td>
<td>EOT</td>
<td>GPL</td>
<td>DTL</td>
<td></td>
<td>Data-transfer between the main-system and FDD</td>
</tr>
<tr>
<td>Execution</td>
<td>R</td>
<td>ST 0</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 1</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>ST 2</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>R</td>
<td>C</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>R</td>
<td>H</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>R</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>R</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Status information after command execution</td>
</tr>
</tbody>
</table>
## Command Summary (continued)

<table>
<thead>
<tr>
<th>PHASE</th>
<th>R/W</th>
<th>D7 D6 D5 D4 D3 D2 D1 D0</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Command</strong></td>
<td>MT</td>
<td>MT MF 0 0 1 0 0 1</td>
<td>WRITE DELETED DATA</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td>Sector ID information prior to command</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td>execution</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td>Execution</td>
<td></td>
<td></td>
<td>Data-transfer between</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FDD and main-system</td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST 0</td>
<td>Status ID information</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 1</td>
<td>after common execution</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 2</td>
<td>Sector ID information</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>C</td>
<td>after command execution</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td><strong>Command</strong></td>
<td>0</td>
<td>MF SK 0 0 0 1 0 0</td>
<td>READ A TRACK</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td>Sector ID information prior to command</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td>execution</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td></td>
</tr>
<tr>
<td>Execution</td>
<td></td>
<td></td>
<td>Data-transfer between</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>the FDD and main-system</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FDC reads all of cylinders contents from</td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST 0</td>
<td>index hole to EOT.</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 1</td>
<td>Status information after command execution</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 2</td>
<td>Sector ID information</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>C</td>
<td>after command execution</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td><strong>Command</strong></td>
<td>0</td>
<td>MF 0 0 1 0 1 0</td>
<td>READ ID</td>
</tr>
<tr>
<td>Execution</td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td>The first correct ID information on the</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>XX X X X X HD US1 US0</td>
<td>cylinder is stored in data register.</td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST 0</td>
<td>Status information after command execution</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>N</td>
<td></td>
</tr>
</tbody>
</table>
Command Summary (continued)

<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>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>X</td>
<td>HD</td>
<td>US1</td>
<td>US0</td>
<td>Bytes/Sector, Sector/Track, Gap 3, filler byte, FDD 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 0</td>
<td></td>
<td></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>Command</td>
<td>W</td>
<td>MT</td>
<td>MF</td>
<td>SK</td>
<td>1</td>
<td>0</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>Execution</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Data compared between the FDD and main-system</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>
<td>Status information after command execution</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>Sector ID information</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>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>
<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>Execution</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Data compared between the FDD and main-system</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>
<td>Status information after command execution</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>Sector ID information after command execution</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>
</tbody>
</table>
## Command Summary (continued)

<table>
<thead>
<tr>
<th>PHASE</th>
<th>R/W</th>
<th>DATA BUS</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>W</td>
<td>MT MF SK X X X HD US1 US0</td>
<td>SCAN HIGH OR EQUAL</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>C</td>
<td>Command Codes</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>H</td>
<td>Sector ID information prior to command execution</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>N</td>
<td>Data compared between the FDD and main-system</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>EOT</td>
<td>Status information after command execution</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>GPL</td>
<td>Sector ID information after command execution</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>STP</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST 0</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST 1</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST 2</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>0 0 0 0 0 1 1 1</td>
<td>RECALIBRATE</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>X X X X X 0 US1 US0</td>
<td>Head retracted to track 0</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>0 0 0 0 1 0 0 0</td>
<td>Command Codes</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>ST 0</td>
<td>Status information at the end of seek operation about the FDC</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>PCN</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>0 0 0 0 0 1 1 1</td>
<td>SPECIFY</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>SRT</td>
<td>Command Codes</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>HLT</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>ND</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>0 0 0 0 0 1 0 0</td>
<td>SENSE DRIVE STATUS</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>X X X X X HD US1 US0</td>
<td>Status information about FDD</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>ST 3</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>0 0 0 0 1 1 1 1</td>
<td>Command Codes</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>X X X X X HD US1 US0</td>
<td>Head is positioned over proper cylinder on diskette</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>NCN</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>INVALID</td>
<td>Invalid command codes</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>Invalid Codes</td>
<td>(NoOp — FDC goes into standby state)</td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST 0</td>
<td>ST 0 = 80</td>
</tr>
</tbody>
</table>
## Command Status Registers

### Table 14. Status Register 0

<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 which 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   |                                                                             |

2-100
<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-systems 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 Data Field) of Status Register 2 is set.</td>
</tr>
</tbody>
</table>
Table 16. Status Register 2

<table>
<thead>
<tr>
<th>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 which 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 with 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 which 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 with the ND bit, and when the contents of C on the medium are different from that stored in the ID Register, and the content 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>
Table 17. Status Register 3

<table>
<thead>
<tr>
<th>BIT</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>Fault</td>
</tr>
<tr>
<td>D6</td>
<td>Write Protected</td>
</tr>
<tr>
<td>D5</td>
<td>Ready</td>
</tr>
<tr>
<td>D4</td>
<td>Track 0</td>
</tr>
<tr>
<td>D3</td>
<td>Two Side</td>
</tr>
<tr>
<td>D2</td>
<td>Head Address</td>
</tr>
<tr>
<td>D1</td>
<td>Unit Select 1</td>
</tr>
<tr>
<td>D0</td>
<td>Unit Select 0</td>
</tr>
</tbody>
</table>

**BIT NO.** | **NAME** | **SYMBOL** | **DESCRIPTION** |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>Fault</td>
<td>FT</td>
<td>This bit is used to indicate 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 used to indicate 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 used to indicate 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 used to indicate 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 used to indicate 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 used to indicate the status of the Head Address signal from the FDD.</td>
</tr>
<tr>
<td>D1</td>
<td>Unit Select 1</td>
<td>US 1</td>
<td>This bit is used to indicate the status of the Unit Select 1 signal to the FDD.</td>
</tr>
<tr>
<td>D0</td>
<td>Unit Select 0</td>
<td>US 0</td>
<td>This bit is used to indicate the status of the Unit Select 0 signal to the FDD.</td>
</tr>
</tbody>
</table>

**Programming Summary**

**DPC Registers (Ports)**

- **FDC Data Reg** I/O Address 3F5
- **FDC Main Status Reg** I/O Address 3F4
- **Digital Output Reg** I/O Address 3F2

<table>
<thead>
<tr>
<th>Bit</th>
<th>Drive</th>
<th>Select</th>
<th>Not FDC Reset</th>
<th>Enable INT &amp; DMA Requests</th>
<th>Drive A Motor Enable</th>
<th>Drive B Motor Enable</th>
<th>Drive C Motor Enable</th>
<th>Drive D Motor Enable</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00: DR #A</td>
<td>01: DR #B</td>
<td>10: DR #C</td>
<td>11: DR #D</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>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

All bits cleared with channel reset.
Interrupt 6

DMA 2

100 Disk Format
1 Head, 45 cylinders, 8 sectors/TRK, 512 bytes/sector, MFM.

FDC Constants
GPL RD/WR: 2A, HLT: 01, (8ms track-track)

Drive Constants
HD Load 35 ms
HD Settle 25 ms
Motor Start 500 ms

Comments
1. Head loads with drive select, wait HD Load time before RD/WR.
2. Following access, wait HD Settle time before RD/WR.
3. Drive motors should be off when not in use. Only A or B and C may run simultaneously. Wait Motor Start time before RD/WR.
4. Motor must be on for drive to be selected.
5. 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:
MPUL 5.5 Vdc
LPUL 2.7 Vdc
MPDL 0.5 Vdc
LPDL -0.5 Vdc

The following lines are used by this adapter.
+D0-7 (Bidirectional, Load: 1 74LS; Driver: 74LS 3-state)
These eight lines form a bus by which all commands, status, and data are transferred. Bit 0 is the low-order bit.
+A0-9 (Adapter Input, Load: 1 74LS)
These ten lines form an address bus by which a register is selected to receive or supply the byte transferred via lines D0-7. Bit 0 is the low-order bit.

+AEN (Adapter Input, Load: 1 74LS)
The content of lines A0-9 is ignored if this line is active.

-IOW (Adapter Input, Load: 1 74LS)
The content of lines D0-7 is stored in the register addressed by lines A0-9 or DACK2 at the trailing edge of this signal.

-IOR (Adapter Input, Load: 1 74LS)
The content of the register addressed by lines A0-9 or DACK2 is gated onto lines D0-7 when this line is active.

-DACK2 (Adapter Input, Load: 2 74LS)
This line active degates output DRQ2, selects the FDC data register as the source/destination of bus D0-7, and indirectly gates T/C to IRQ6.

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

+RESET (Adapter Input, Load: 1 74LS)
An up level aborts any operation in process and clears the Digital Output Register (DOR).

+DRQ2 (Adapter Output, Driver: 74LS 3-state)
This line is made active when the attachment is ready to transfer a byte of data to or from main storage. The line is made inactive by DACK2 becoming active or an I/O read of the FDC data register.

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

All signals are TTL compatible:

MPUL 5.5 Vdc
LPUL 2.4 Vdc
MPDL 0.4 Vdc
LPDL -0.5 Vdc

All adapter outputs are driven by open-collector gates. The drive(s) must provide termination networks to Vcc (except Motor Enable 1 which has a two kohm resistor to Vcc).

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

Adapter Outputs

-Drive Select A&B  (Driver: 7438)
  These two lines are used by drives A&B to degate all drivers to the adapter and receivers from the attachment (except Motor Enable) when the line associated with a drive is not active.

-Motor Enable A&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 not active.

-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 not-active.

-Write Data  (Driver: 7438)
  For each not-active to active transition of this line while Write Enable is active, the selected drive causes a flux change to be stored on the disk.

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

2-106
## Adapter Inputs

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Index</td>
<td>The selected drive supplies one pulse per disk revolution on this line.</td>
</tr>
<tr>
<td>-Write Protect</td>
<td>The selected drive makes this line active if a write protected diskette is mounted in the drive.</td>
</tr>
<tr>
<td>-Track 0</td>
<td>The selected drive makes this line active if the read/write head is over track 0.</td>
</tr>
<tr>
<td>-Read Data</td>
<td>The selected drive supplies a pulse on this line for each flux change encountered on the disk.</td>
</tr>
</tbody>
</table>
# 5-1/4" Diskette Drive Adapter
## Internal Interface Specifications

![Diagram of 34-pin connector](image)

**34 PIN KEYED EDGE CONNECTOR**

**NOTE:** LANDS 1–33 ARE ON THE BACKSIDE OF THE BOARD, LANDS 2–34 ARE ON THE FRONT, OR COMPONENT SIDE.

### AT STANDARD TTL LEVELS

<table>
<thead>
<tr>
<th>Ground-Odd Numbers</th>
<th>Land No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unused</td>
<td>1-33</td>
</tr>
<tr>
<td>Index</td>
<td>2,4,6</td>
</tr>
<tr>
<td>Motor Enable A</td>
<td>8</td>
</tr>
<tr>
<td>Drive Select B</td>
<td>10</td>
</tr>
<tr>
<td>Drive Select A</td>
<td>12</td>
</tr>
<tr>
<td>Motor Enable B</td>
<td>14</td>
</tr>
<tr>
<td>Direction (Stepper Motor)</td>
<td>16</td>
</tr>
<tr>
<td>Step Pulse</td>
<td>18</td>
</tr>
<tr>
<td>Write Data</td>
<td>20</td>
</tr>
<tr>
<td>Write Enable</td>
<td>22</td>
</tr>
<tr>
<td>Track 0</td>
<td>24</td>
</tr>
<tr>
<td>Write Protect</td>
<td>26</td>
</tr>
<tr>
<td>Read Data</td>
<td>28</td>
</tr>
<tr>
<td>Select Head 1</td>
<td>30</td>
</tr>
<tr>
<td>Unused</td>
<td>32</td>
</tr>
</tbody>
</table>

**IBM 5 1/4" Diskette Drives**

**5 1/4" Diskette Drive Adapter**

---

2-108
## 5-1/4" Diskette Drive Adapter

### External Interface Specifications

#### Rear Panel

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

#### Diagram:

![Diagram of 5-1/4" Diskette Drive Adapter](image)

---

*hardware*
5-1/4" Diskette Drive

The IBM 5-1/4" Diskette Drive is a single sided, double density, 40 track unit. The Diskette Drive has a formatted capacity of 163,840 bytes, and is capable of reading and recording digital data using Modified Frequency Modulation (MFM) methods. User access for diskette loading is provided by way of a slot located at the front of the unit.

The Diskette Drive is fully self-contained and requires no operator intervention during normal operation. The Drive consists of a spindle drive system, a head positioning system, and read/write/erase system.

When the front latch is opened, access is provided for the insertion of a diskette. The diskette is positioned in place by plastic guides, and the front latch. In/out location is ensured when the diskette is inserted until a back stop is encountered.

Closing the front latch activates the cone/clamp system resulting in centering of the diskette and clamping of the diskette to the drive hub. The drive hub is driven at a constant speed of 300 rpm by a servo controlled DC motor. In operation, the magnetic head is loaded into contact with the recording medium whenever the front latch is closed.

The magnetic head is positioned over the desired track by means of a 4-phase stepper motor/band assembly and its associated electronics. This positioner employs a one-step rotation to cause a 1-track linear movement. When a write-protected diskette is inserted into the Drive, the write-protect sensor disables the write electronics of the Drive and an appropriate signal is applied to the interface.

When performing a write operation, a 0.33 mm (0.013-in.) data track is recorded. This track is then tunnel erased to 0.30 mm (0.012 in.).

Data recovery electronics include a low-level read amplifier, differentiator, zero-crossing detector, and digitizing circuits. All data decoding is provided by the adapter card.

The Drive is also supplied with the following sensor systems:

1. A track 00 switch which senses when the Head/Carriage assembly is positioned at Track 00.
2. The index sensor, which consists of a LED light source and phototransistor, is positioned such that when an index hole is detected, a signal signal is generated.
The write-protect sensor disables the Diskette Drive electronics whenever a write-protect tab is applied to the diskette.

For Interface Information, refer to the Diskette Drive Adapter section.

Diskettes

The IBM 5-1/4" Diskette Drive uses a standard 133.4 mm (5.25 in.) diskette. For programming considerations, single sided, double density soft sectored diskettes are used. 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.
## Table 18. Mechanical and Electrical Specifications

<table>
<thead>
<tr>
<th>Media</th>
<th>Industry-compatible 5¼-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>85.85 mm (3.38 inches)</td>
</tr>
<tr>
<td>Width</td>
<td>149.10 mm (5.87 inches)</td>
</tr>
<tr>
<td>Depth</td>
<td>203.2 mm (8.0 inches)</td>
</tr>
<tr>
<td>Weight</td>
<td>2.04 Kg (4.5 lbs.)</td>
</tr>
<tr>
<td>Temperature</td>
<td></td>
</tr>
<tr>
<td>(Exclusive of Media)</td>
<td></td>
</tr>
<tr>
<td>Operating</td>
<td>10°C to 44°C (50°F to 112°F)</td>
</tr>
<tr>
<td>Non-operating</td>
<td>−40°C to 60°C (−40°F to 140°F)</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>8 msec track to track</td>
</tr>
<tr>
<td>Head Setting Time</td>
<td>25 msec (last track addressed)</td>
</tr>
<tr>
<td>Error Rate</td>
<td>1 per $10^9$ (recoverable)</td>
</tr>
<tr>
<td></td>
<td>1 per $10^{12}$ (non-recoverable)</td>
</tr>
<tr>
<td></td>
<td>1 per $10^6$ (seeks)</td>
</tr>
<tr>
<td>Head Life</td>
<td>20,000 hours (normal use)</td>
</tr>
<tr>
<td>Media Life</td>
<td>$3.0 \times 10^6$ 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>500 msec (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 dc ± 0.6v 900 ma AVE.</td>
</tr>
<tr>
<td></td>
<td>+5v dc ± 0.25 v, 600 ma AVE.</td>
</tr>
</tbody>
</table>
Memory Expansion Options

Two Memory Expansion Options offered for the IBM Personal Computer are the 32K x 9 and the 64K x 9 Memory Expansion Options. These options plug into any of the five System Expansion slots on the System Board. These options are used to extend system memory beyond 64KB. A maximum of 64KB of memory may be installed on the System Board as modules without using any System Expansion Slots or Expansion Options.

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

The expansion options are designed with 250 ns access 16K x 1 dynamic memory chips. On the 32KB card, 16-pin industry standard parts are used. On the 64KB card, stacked modules are used resulting in a 32K x 1 18-pin module. This allows the 32KB and 64KB to have approximately the same packaging densities.

Both expansion options are parity checked and 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 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 which are not performed on the expansion options but are done once on the System Board and made available in the I/O channel for all devices.

To allow the System to address 32KB and 64KB Memory Expansion Options, refer to the system configuration switch settings page 2-28.

Operating Characteristics

The System Board operates at a frequency of 4.77 Mhz, which results in a clock frequency of 210 ns.

 Normally, four clock cycles are required for a bus cycle so that an 840 nsec 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 both cards are:

Access - 250 ns
Cycle - 410 ns

**Memory Module Description**

Each option contains 18 dynamic memory modules. The 32KB Memory Expansion Option utilizes 16K x 1 bit modules and the 64KB Memory Expansion Options utilizes 32K x 1 bit modules.

Both memory modules require three voltage levels (+5Vdc, -5Vdc, +12Vdc) and 128 refresh cycles every 2 msec. Absolute maximum access times are:

From RAS: 250 ns
From CAS: 165 ns

**Table 19. Memory Module Pin Configuration**

<table>
<thead>
<tr>
<th>PIN NO.</th>
<th>16K X 1 BIT MODULE (Used on 32KB Card)</th>
<th>32K X 1 BIT MODULE (Used on 64KB Card)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>- 5V</td>
<td>- 5V</td>
</tr>
<tr>
<td>2</td>
<td>Data In **</td>
<td>Data In **</td>
</tr>
<tr>
<td>3</td>
<td>- Write</td>
<td>- Write</td>
</tr>
<tr>
<td>4</td>
<td>- RAS</td>
<td>- RAS 0</td>
</tr>
<tr>
<td>5</td>
<td>A0</td>
<td>- RAS 1</td>
</tr>
<tr>
<td>6</td>
<td>A2</td>
<td>A0</td>
</tr>
<tr>
<td>7</td>
<td>A1</td>
<td>A2</td>
</tr>
<tr>
<td>8</td>
<td>+ 12V</td>
<td>A1</td>
</tr>
<tr>
<td>9</td>
<td>+ 5V</td>
<td>+ 12V</td>
</tr>
<tr>
<td>10</td>
<td>A5</td>
<td>+ 5V</td>
</tr>
<tr>
<td>11</td>
<td>A4</td>
<td>A5</td>
</tr>
<tr>
<td>12</td>
<td>A3</td>
<td>A4</td>
</tr>
<tr>
<td>13</td>
<td>A6</td>
<td>A3</td>
</tr>
<tr>
<td>14</td>
<td>Data Out **</td>
<td>A6</td>
</tr>
<tr>
<td>15</td>
<td>- CAS</td>
<td>Data Out **</td>
</tr>
<tr>
<td>16</td>
<td>GND</td>
<td>- CAS 1</td>
</tr>
<tr>
<td>17</td>
<td>- *</td>
<td>- CAS 0</td>
</tr>
<tr>
<td>18</td>
<td>- *</td>
<td>GND</td>
</tr>
</tbody>
</table>

* 16K X 1 bit module has only 16 pins.
** Data In and Data Out are tied together (three state bus).
Switch - Configurable Start Address

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

Table 20. DIP Module Start Address

<table>
<thead>
<tr>
<th>NO.</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ON: A19=0 ; OFF: A19=1</td>
</tr>
<tr>
<td>2</td>
<td>ON: A18=0 ; OFF: A18=1</td>
</tr>
<tr>
<td>3</td>
<td>ON: A17=0 ; OFF: A17=1</td>
</tr>
<tr>
<td>4</td>
<td>ON: A16=0 ; OFF: A16=1</td>
</tr>
<tr>
<td>5</td>
<td>ON: A15=0 ; OFF: A15=1 *</td>
</tr>
<tr>
<td>6</td>
<td>Not Used</td>
</tr>
<tr>
<td>7</td>
<td>Not Used</td>
</tr>
<tr>
<td>8</td>
<td>Used Only In 64KB RAM Card *</td>
</tr>
</tbody>
</table>

* Switch No. 8 may be set on the 64KB Memory Expansion Option to use only half the memory on the card (i.e., 32KB). If Switch No. 8 is ON, all 64KB is accessible. If Switch No. 8 is OFF, address bit A15 (as set by Switch No. 5) is used to determine which 32KB are accessible and the 64KB option behaves exactly like a 32KB option.
Game Control Adapter

The Game Control Adapter allows the system to attach paddles and joysticks. Up to four paddles or two joysticks may be attached. In addition, four input for switches are provided. Paddle and joystick 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 joystick 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 card could be used as a general purpose I/O card with four analog (resistive) inputs plus four digital input points. This card fits into any of the five System Board I/O slots. The game control interface cable attaches to the rear of the card which protrudes through the rear panel of the System Unit.

Game Control Adapter Block Diagram

Figure 20. GAME CONTROL ADAPTER BLOCK DIAGRAM
Functional Description

Address Decode
The select on the Game Control Adapter is generated by two 74LS138's as an address decoder. AEN must be inactive while the address is 201 in order to generate the select. The select allows a write to fire the one-shots or a read to give the values of the trigger buttons and one-shot outputs.

Data Bus Buffer/Driver
The data bus is buffered by a 74LS244 buffer/driver. For an IN from address X'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 via an IN from address X'201'. A trigger button is on each joystick/paddle. These values are seen on data bits 7 through 4 (see Software Interface sub-section). These buttons default to an open state and are read as “1”. When a button is depressed, it is read as “0”. Software should be aware that these buttons are NOT debounced in hardware.

Joystick Positions
The joystick 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 X'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 via an IN from address X'201' and are seen on data bits 3 through 0.
I/O Channel Description

A9-A0: Address lines 9 through 0 are used to address the Game Control Adapter.

D7-D0: Data lines 7 through 0 are the data bus.

IOR, IOW: I/O Read and I/O Write are used when reading from or writing to an adapter (IN, OUT).

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

+5V: Power for the Game Control Adapter.

GND: Common ground.

A19-A10: Unused

MEMR, MEMW: Unused

DACK0-DACK3: Unused

IRQ7-IRQ2 Unused

DRQ3-DRQ1: Unused

ALE, T/C: Unused

CLK, OSC: Unused

I/O CHCK: Unused

I/O CH RDY: Unused

HRQ I/O CH: Unused

RESET DRV: Unused

-5v, +12v, -12v: Unused

Interface Description

The Game Control Adapter has 8 input lines, 4 of which are digital inputs and 4 of which are resistive inputs. The inputs are read with one IN from address x'201'.

The 4 digital inputs each have a 1K ohm pullup resistor to +5V. With no drive on these inputs, a ‘1’ is read. For a ‘0’ reading, the inputs must be pulled to ground.

The 4 resistive inputs, measured to +5V, 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 conversion by an OUT to address x‘201’. An IN from address x‘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.

Input from address x‘201’

<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>
</table>

Digital Inputs

Resistive Inputs

The typical input to the Game Control Adapter is a set of joysticks or game paddles.

The joysticks will typically have a set of two joysticks (A&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</td>
<td>B-#1</td>
<td>A-#2</td>
<td>A-#1</td>
<td>B-Y</td>
<td>B-X</td>
<td>A-Y</td>
<td>A-X</td>
</tr>
</tbody>
</table>


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

<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>
</tbody>
</table>


A schematic diagram for attaching a set of game controllers is on page 2-121.
15 PIN MALE 'D' SHELL

NOTE: POTENTIOMETER FOR X & Y COORDINATES HAS A RANGE OF 0 TO 100KΩ. BUTTON IS NORMALLY OPEN; CLOSED WHEN DEPRESSED.

Figure 21. JOYSTICK SCHEMATIC
Game Controller Adapter (Analog Input)
Connector Specifications

AT STANDARD TTL LEVELS

<table>
<thead>
<tr>
<th>Voltage</th>
<th>AMP Pin No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>+5 Volts</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 Volts</td>
<td>8</td>
</tr>
<tr>
<td>+5 Volts</td>
<td>9</td>
</tr>
<tr>
<td>Button 6</td>
<td>10</td>
</tr>
<tr>
<td>Position 2</td>
<td>11</td>
</tr>
<tr>
<td>Ground</td>
<td>12</td>
</tr>
<tr>
<td>Position 3</td>
<td>13</td>
</tr>
<tr>
<td>Button 7</td>
<td>14</td>
</tr>
<tr>
<td>+5 Volts</td>
<td>15</td>
</tr>
</tbody>
</table>

External Devices     Game Control Adapter
Asynchronous Communications Adapter

The Asynchronous Communications Adapter is a 4”H x 5”W card that plugs into a System Expansion Slot. All system control signals and voltage requirements are provided through a 2 x 31 position card edge tab. A jumper module is provided to select either RS-232-C or current loop operation.

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.

Figure (22) is a block diagram of the Asynchronous Communications Adapter.

The heart of the adapter is a INS8250 LSI chip or functional equivalent. The following is a summary of the 8250’s key features:

- Adds or Delete Standard Asynchronous Communication Bits (Start, Stop, and Parity) to or from Serial Data Stream.
- Full Double Buffering Eliminates Need for Precise Synchronization.
- Independently Controlled Transmit, Receive, Line Status, and Data Set Interrupts.
- Programmable Baud Rate Generator Allows Division of Any Input Clock by 1 to \((2^{16}-1)\) and Generates the Internal 16x Clock.
- 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.
- Fully Programmable Serial-Interface Characteristics
  - 5-, 6-, 7-, or 8-Bit Characters
  - Even, Odd, or No-Parity Bit Generation and Detection
  - 1-, 1 1/2-, or 2-Stop Bit Generation
  - Baud Rate Generation (DC to 9600 Baud)
• False Start Bit Detection.
• Complete Status Reporting Capabilities.
• Line Break Generation and Detection.
• Internal Diagnostic Capabilities.
  - Loopback Controls for Communications Link Fault Isolation.
  - Break, Parity, Overrun, Framing Error Simulation.
• Full Prioritized Interrupt System Controls.

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.

Asynchronous Communications Block Diagram

Figure 22. 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 (3F8 to 3FF) and writing data out to the card. Address bit A0, A1 and A2 select the different registers which 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 for Communications Adapter

Table 21. I/O Decodes (3F8 to 3FF)

<table>
<thead>
<tr>
<th>I/O DECODE</th>
<th>REGISTER SELECTED</th>
<th>DLAB STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>3F8</td>
<td>TX BUFFER</td>
<td>DLAB=0</td>
</tr>
<tr>
<td>3F8</td>
<td>RX BUFFER</td>
<td>DLAB=0</td>
</tr>
<tr>
<td>3F8</td>
<td>DIVISOR LATCH LSB</td>
<td>DLAB=1</td>
</tr>
<tr>
<td>3F9</td>
<td>DIVISOR LATCH MSB</td>
<td>DLAB=1</td>
</tr>
<tr>
<td>3F9</td>
<td>INTERRUPT ENABLE REGISTER</td>
<td>DLAB=0</td>
</tr>
<tr>
<td>3FA</td>
<td>INTERRUPT IDENTIFICATION REGISTERS</td>
<td></td>
</tr>
<tr>
<td>3FB</td>
<td>LINE CONTROL REGISTER</td>
<td></td>
</tr>
<tr>
<td>3FC</td>
<td>MODEM CONTROL REGISTER</td>
<td></td>
</tr>
<tr>
<td>3FD</td>
<td>LINE STATUS REGISTER</td>
<td></td>
</tr>
<tr>
<td>3FE</td>
<td>MODEM STATUS REGISTER</td>
<td></td>
</tr>
</tbody>
</table>

ADDRESS BITS

```
3F8 to 3FF  | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | DLAB | REGISTER               
-----------|----|----|----|----|----|----|----|----|----|----|------|------------------------
            | 1  | 1  | 1  | 1  | 1  | 1  | X  | X  | X  | 0  | 0    | Receive Buffer (read), Transmit Holding Reg. (write) |
            | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 0  | 0  | 0  | 1    | Interrupt Enable       |
            | 0  | 1  | 0  | X  | 0  | 1  | 1  | X  | 0  | 1  | 1    | Interrupt Identification |
            | 1  | 0  | 0  | X  | 1  | 0  | 1  | X  | 0  | 1  | 1    | Line Control            |
            | 1  | 0  | 0  | X  | 0  | 1  | 1  | X  | 0  | 0  | 1    | Modem Control           |
            | 1  | 1  | 0  | X  | 0  | 0  | 0  | X  | 0  | 1  | 1    | Line Status             |
            | 1  | 1  | 0  | X  | 1  | 1  | 0  | X  | 0  | 0  | 1    | Modem Status            |
            | 1  | 1  | 0  | X  | 0  | 1  | 1  | X  | 0  | 0  | 1    | None                    |
            | 0  | 0  | 0  | 1  | 0  | 0  | 1  | X  | 0  | 1  | 1    | Divisor Latch (LSB)     |
            | 0  | 0  | 0  | 1  | 0  | 1  | 0  | X  | 0  | 1  | 0    | Divisor Latch (MSB)     |
```

A2, A1 and A0 bits are "Don’t Cares" and are used to select the different register of the communications chip.
Interrupts

One interrupt line is provided to the system. This interrupt is IRQ4 and will be positive active. To allow the communications card to send interrupts to the system, Bit 3 of the Modem Control Register must be set = 0 (low). At this point, any interrupts allowed by the Interrupt Enable Register will cause an interrupt.

The data format will be as follows:

![Transmitter Output and Receiver Input Diagram]

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-232-C 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 Corporation that use this particular type of interface.

Pin 18 + receive current loop data (20Ma)
Pin 25 - receive current loop return (20Ma)
Pin 9 + transmit current loop return (20Ma)
Pin 11 - transmit current loop data (20Ma)
The voltage interface is a serial interface. It supports certain data and control signals as listed below.

- Pin 2: Transmit Data
- Pin 3: Receive Data
- Pin 4: Request to Send
- Pin 5: Clear to Send
- Pin 6: Data Set Ready
- Pin 7: Signal Ground
- Pin 8: Carrier Detect
- Pin 20: Data Terminal Ready
- Pin 22: Ring Indicate

The adapter converts these signals to/from TTL levels to EIA voltage levels. These signals are sampled or generated by the communication 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

+15V — — — — — — — — — — — — — — — — — — — — — — — —
On Function

+3V — — — — — — — — — — — — — — — — — — — — — — — —
0V

Invalid Levels

-3V — — — — — — — — — — — — — — — — — — — — — — — —
Off Function

-15V — — — — — — — — — — — — — — — — — — — — — — — —
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 minus three volts with respect to signal ground. The signal will be considered in the “spacing” condition when the voltage is more positive than plus three volts with respect to signal ground. The region between plus three volts and minus three volts is defined as the transition region, will be considered in invalid levels. The voltage which is more negative than -15V or more positive than +15V will be considered in invalid levels.

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

For interface control circuits, the function is “on” when the voltage is more positive than +3V with respect to signal ground and is “off” when the voltage is more negative than -3V with respect to signal ground.
INS8250 Functional Pin Description

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

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

Input Signals

Chip Select (SC0, CS1, CS2), Pins 12-14: When CS0 and CS1 are high and 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 enable communication between the INS8250 and the CPU.

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

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

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

Note: Only an active DOSTR or DOSTR input is required to transfer data to the INS8250 during a write operation. Therefore, tie either the DOSTR input permanently low or the 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 (SOC, CS1, CS2) signals.

Note: An active 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, the 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 into 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 byte)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Divisor Latch (most significant byte)</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 Table 1.)

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

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

Clear to Send (CTS), Pin 36: The CTS signal is a MODEM control function input whose condition can be tested by the CPU 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 CPU 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 state 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 has been detected by the MODEM or data set. The RLSD signal is a MODEM-Control function input whose condition can be tested by the CPU 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 CPU 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 a high state since the previous reading of the MODEM Status Register.

**Note:** Whenever the RI bit of the MODEM Status Register changes from a high to a low state, an interrupt is generated if the MODEM Status Interrupt is enabled.

VCC, Pin 40: +5 volt supply.

VSS, Pin 20: Ground (0-volt) 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 logic 1.

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

Baud Out (BAUDOUT), Pin 15: 16x 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 via the IER: Receiver Error Flag; Received Data Available; Transmitter Holding Register Empty; and 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 (Logic 1) state upon a Master Reset operation.
Input/Output Signals

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

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

Programming Considerations

Table 22. Asynchronous Communications Reset Functions

<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</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0–3 Forced and 4–7 Permanent)</td>
</tr>
<tr>
<td>Interrupt Identification</td>
<td>Master Reset</td>
<td>Bit 0 is High,</td>
</tr>
<tr>
<td>Register</td>
<td></td>
<td>Bits 1 and 2 Low</td>
</tr>
<tr>
<td>Line Control Register</td>
<td>Master Reset</td>
<td>Bits 3–7 are</td>
</tr>
<tr>
<td>MODEM Control Register</td>
<td>Master Reset</td>
<td>Permanently Low</td>
</tr>
<tr>
<td>Line Status Register</td>
<td>Master Reset</td>
<td>All Bits Low</td>
</tr>
<tr>
<td>MODEM Status Register</td>
<td>Master Reset</td>
<td>All Bits Low</td>
</tr>
<tr>
<td>SOUT</td>
<td>Master Reset</td>
<td>Except Bits 5 &amp; 6 are High</td>
</tr>
<tr>
<td>INTRPT (RCVR Errs)</td>
<td>Read LSR/MR</td>
<td>Bits 0–3 Low</td>
</tr>
<tr>
<td>INTRPT (RCVR Data Ready)</td>
<td>Read RBR/MR</td>
<td>Bits 4–7 — Input Signal</td>
</tr>
<tr>
<td>INTRPT (RCVR Data Ready)</td>
<td>Read IIR/Write</td>
<td>Low</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>
INS8250 Accessible Registers

The system programmer may access or control any of the INS8250 registers via the CPU. These registers are used to control INS8250 operations and to transmit and receive data.

INS8250 Line Control Register

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

Line Control Register (LCR)

3FB

<table>
<thead>
<tr>
<th>BIT</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>
</tr>
<tr>
<td>Word Length Select Bit 0 (WLS0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Word Length Select Bit 1 (WLS1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Stop Bits (STB)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parity Enable (PEN)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Even Parity Select (EPS)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stick Parity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set Break</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Divisor Latch Access Bit (DLAB)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bit 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>

Bit 2: This bit specifies the number of Stop bits in each transmitted or received serial character. If bit 2 is a logic 0, 1 Stop bit is generated or checked in the transmit or receive data, respectively. If bit 2 is logic 1 when a 5-bit word length is selected via bits 0 and 1, 1-1/2 Stop bits are generated or checked. If bit 2 is logic 1 when either a 6-, 7-, or 8-bit word length is selected, 2 Stop bits are generated or checked.
Bit 3: This bit is the Parity Enable bit. When bit 3 is a logic 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 logic 1 and bit 4 is a logic 0, an odd number of logic 1’s is transmitted or checked in the data word bits and Parity bit. When bit 3 is a logic 1 and bit 4 is a logic 1, an even number of bits is transmitted or checked.

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

Bit 6: This bit is the Set Break Control bit. When bit 6 is a logic 1, the serial output (SOUT) is forced to the Spacing (logic 0) state and remains there regardless of other transmitter activity. The set break is disabled by setting bit 6 to a logic 0. This feature enables the CPU 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 (logic 1) to access the Divisor Latches of the Baud Rate Generator during a Read or Write operation. It must be set low (logic 0) to access the Receiver Buffer, the Transmitter Holding Register, or the Interrupt Enable Register.

INS8250 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 $16x$ the Baud rate $[\text{divisor} \# = \frac{\text{frequency input}}{\text{baud rate} \times 16}]$. Two 8-bit latches store the divisor in a 16-bit binary format. These Divisor Latches must be loaded during initialization in order to ensure desired operation of the Baud Rate Generator. Upon loading either of the Divisor Latches, a 16-bit Baud counter is immediately loaded. This prevents long counts on initial load.
Table 23 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 23. BAUD RATE AT 1.843 Mhz

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

Line Status Register

This 8-bit register provides status information to the CPU concerning the data transfer. The contents of the Line Status Register are indicated and described below.

Line Status Register (LSR)

3FD
Bit 0: This bit is the receiver Data Ready (DR) indicator. Bit 0 is set to a logic 1 whenever a complete incoming character has been received and transferred into the Receiver Buffer Register. Bit 0 may be reset to a logic 0 either by the CPU reading the data in the Receiver Buffer Register or by writing a logic 0 into it from the CPU.

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 CPU before the next character was transferred into the Receiver Buffer Register, thereby destroying the previous character. The OE indicator is reset whenever the CPU 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 logic 1 upon detection of a parity error and is reset to a logic 0 whenever the CPU 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 logic 1 whenever the Stop bit following the last data bit or parity bit is detected as a zero bit (Spacing level).

Bit 4: This bit is the Break Interrupt (BI) indicator. Bit 4 is set to a logic 1 whenever the received data input is held in the Spacing (logic 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 CPU when the Transmit Holding Register Empty Interrupt enable is set high. The THRE bit is set to a logic 1 when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register. The bit is reset to logic 0 concurrently with the loading of the Transmitter Holding Register by the CPU.

Bit 6: This bit is the Transmitter Shift Register Empty (TSRE) indicator. Bit 6 is set to a logic 1 whenever the Transmitter Shift Register is idle. It is reset to logic 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 logic 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. The four levels of interrupt conditions are as follows: 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 that interrupt are stored in the Interrupt Identification Register (refer to Table 5). 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 CPU. The contents of the IIR are indicated and described below.

### Interrupt Identification Register (IIR)

<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>OIF INTERRUPT PENDING</td>
<td>INTERRUPT ID BIT (0)</td>
<td>INTERRUPT ID BIT (1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>= 0</td>
<td>= 0</td>
<td>= 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>= 0</td>
<td>= 0</td>
<td>= 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bit 0: This bit can be used in either a hardwired prioritized or polled environment to indicate whether an interrupt is pending. When bit 0 is a logic 0, 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 logic 1, no interrupt is pending and polling (if used) continued.

Bits 1 and 2: These two bits of the IIR are used to identify the highest priority interrupt pending as indicated in Table 5.

Bits 3 through 7: These five bits of the IIR are always logic 0.
### Table 24. Interrupt Control Functions

<table>
<thead>
<tr>
<th>Interrupt ID Register</th>
<th>Priority Level</th>
<th>Interrupt Type</th>
<th>Interrupt Source</th>
<th>Interrupt Reset Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 2 Bit 1 Bit 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0 1</td>
<td></td>
<td>None</td>
<td>Overrun Error or Parity Error or Framing Error or Break Interrupt</td>
<td></td>
</tr>
<tr>
<td>1 1 0</td>
<td>Highest</td>
<td>Receiver Line Status</td>
<td>Reading the Line Status Register</td>
<td></td>
</tr>
<tr>
<td>1 0 0</td>
<td>Second</td>
<td>Data Available</td>
<td>Reading the Receiver Data Available Buffer Register</td>
<td></td>
</tr>
<tr>
<td>0 1 0</td>
<td>Third</td>
<td>Transmitter Holding Register Empty</td>
<td>Reading the IIR Register (if source of interrupt) or Writing into the Transmitter Holding Register</td>
<td></td>
</tr>
<tr>
<td>0 0 0</td>
<td>Fourth</td>
<td>MODEM Status</td>
<td>Clear to Send or Data Set Ready or Ring Indicator or Received Line Signal Detect</td>
<td>Reading the MODEM Status Register</td>
</tr>
</tbody>
</table>
Interrupt Enable Register

This 8-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 logic 1, selected interrupts can be enabled. Disabling the interrupt system inhibits the Interrupt Identification Register and the active (high) INTRPT output from the chip. All other system functions operate in their normal manner, including the setting of the Line Status and MODEM Status Registers. The contents of the Interrupt Enable Register are indicated and described below.

Interrupt Enable Register (IER)

3F9 DLAB=0

<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>ENABLE DATA AVAILABLE INTERRUPT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1=</td>
<td>ENABLE TX HOLDING REG EMPTY INTERRUPT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1=</td>
<td>ENABLE RECEIVE LINE STATUS INTERRUPT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1=</td>
<td>ENABLE MODEM STATUS INTERRUPT</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>
</tbody>
</table>

Bit 0: This bit enables the Received Data Available Interrupt when set to logic 1.

Bit 1: This bit enables the Transmitter Holding Register Empty Interrupt when set to logic 1.

Bit 2: This bit enables the Receiver Line Status Interrupt when set to logic 1.

Bit 3: This bit enables the MODEM Status Interrupt when set to logic 1.

Bits 4 through 7: These four bits are always logic 0.
MODEM Control Register

This 8-bit register controls the interface with the MODEM or data set (or a peripheral device emulating a MODEM). The contents of the MODEM Control Register are indicated and described below.

MODEM Control Register (MCR)

3FC

<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>DATA TERMINAL READY (DTR)</td>
<td>REQUEST TO SEND (RTS)</td>
<td>OUT 1</td>
<td>OUT 2</td>
<td>LOOP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Bit 0: This bit controls the Data Terminal Ready (DTR) output. When bit 0 is set to a logic 1, the DTR output is forced to a logic 0. When bit 0 is reset to a logic 0, the DTR output is forced to a logic 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 (OUT 1) signal, which is an auxiliary user-designated output. Bit 2 affects the OUT 1 output in a manner identical to that described above for bit 0.

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

Bit 4: This bit provides a loopback feature for diagnostic testing of the INS8250. When bit 4 is set to logic 1, the following occur: the transmitter Serial Output (SOUT) is set to the Marking (logic 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,DSR, 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 logic 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 logic 0.

**Bits 5 through 7:** These bits are permanently set to logic 0.

**MODEM Status Register**

This 8-bit register provides the current state of the control lines from the MODEM (or peripheral device) to the CPU. In addition to this current-state information, four bits of the MODEM Status Register provide change information. These bits are set to a logic 1 whenever a control input from the MODEM changes state. They are reset to logic 0 whenever the CPU reads the MODEM Status Register.

The content of the MODEM Status Register are indicated and described below.

**MODEM Status Register (MSR)**

<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>

- **DELTA CLEAR TO SEND (DCTS)**
- **DELTA DATA SET READY (DDSR)**
- **TRAILING EDGE RING INDICATOR (TERI)**
- **DELTA RX LINE SIGNAL DETECT (DRLSD)**
- **CLEAR TO SEND (CTS)**
- **DATA SET READY (DSR)**
- **RING INDICATOR (RI)**
- **RECEIVE LINE SIGNAL DETECT (RLSD)**
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 CPU.

Bit 1: This bit is the Delta Data Set Ready (DDSR) indicator. Bit 1 indicates that the DSR input to the chip has changed state since the last time it was read by the CPU.

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 (logic 1) to an Off (logic 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 logic 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 1, this bit 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 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 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 1, this bit is equivalent to OUT 2 of the MCR.

Receiver Buffer Register

The Receiver Buffer Register contains the received character as defined below.

Receiver Buffer Register (RBR)

3F8  DLAB=0  READ ONLY

Bit 0 is the least significant bit and is the first bit serially received.
Transmitter Holding Register

The Transmitter Holding Register contains the character to be serially transmitted and is defined below:

Transmitter Holding Register (THR)

3F8  DLAB=0  WRITE ONLY

Bit 0 is the least significant bit and is the first bit serially transmitted.
Selecting The Interface Format

The Voltage or Current loop interface is selected by plugging the programmed shunt module, with the locator dot up or down. See the figure below for the two configurations.

![Diagram showing interface configurations]

Figure 23. SELECTING THE INTERFACE FORMAT
Asynchronous Communications Adapter Connector Interface Specifications

AT STANDARD TTL LEVELS

<table>
<thead>
<tr>
<th>Description</th>
<th>Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC</td>
<td>1</td>
</tr>
<tr>
<td>Transmit Data</td>
<td>2</td>
</tr>
<tr>
<td>Receive 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>Carrier detect</td>
<td>8</td>
</tr>
<tr>
<td>+Transmit current loop return (20 ma)</td>
<td>9</td>
</tr>
<tr>
<td>NC</td>
<td>10</td>
</tr>
<tr>
<td>+Transmit current loop data (20 ma)</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 (20 ma)</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 Indicate</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 (20 ma)</td>
<td>25</td>
</tr>
</tbody>
</table>

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.
SECTION 3. ROM and SYSTEM USAGE

Contents:

- ROM BIOS ............................................. 3-2
- BIOS Cassette Logic ................................. 3-8
- Keyboard Encoding and Usage ..................... 3-11
- Low Memory Maps ................................. 3-21
ROM BIOS

The ROM resident Basic I/O System (BIOS) provides the device level control of the major I/O devices in the System Unit. The BIOS routines allow the assembly language programmer to perform block (diskette and cassette) or character (Video, communications, keyboard and printer) level I/O operations without any concern for device address and operating characteristics. Additionally, system services such as time of day and memory size determination are provided. The goal is to provide an operational interface to the system and relieve the programmer from concern over hardware device characteristics. Finally the BIOS interface insulates the user from the hardware allowing new devices to be added to the System Unit, yet retaining the BIOS level interface to the device. In this manner, user programs become transparent to hardware modifications and enhancements. A complete listing of the BIOS is provided in Appendix “A”.

Use of BIOS

Access to the BIOS function is through the 8088 software interrupts. Each BIOS entry point is available through its own interrupt, which can be found in the interrupt vector listing. The software interrupts 10H through 1AH each access a different BIOS routine. For example, to determine the amount of memory available in the system,

```
INT 12H
```

will invoke the memory size determination routine in BIOS and return the value to the caller.

Parameter Passing

All parameters passed to and from the BIOS routines go through the 8088 registers. The prologue of each BIOS function indicate the registers used on the call and the return. For the memory size example above, no parameters are passed, and the result, memory size in 1K Byte increments is returned in the AX register.

Where a BIOS function has several possible operations, the AH register is used on input to indicate the desired operation. For example, to set the time of day, the following code is required.

3-2
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.

While to read the time of day:
MOV AH, 0 ; function is to read the time of day.
INT 1AH ; read the timer.

As a general rule, the BIOS routines preserve 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 prologue of each BIOS function.

## Interrupt Vector Listing

<table>
<thead>
<tr>
<th>Interrupt Number</th>
<th>Name</th>
<th>BIOS Initialization</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Divide by Zero</td>
<td>None</td>
</tr>
<tr>
<td>1</td>
<td>Single Step</td>
<td>None</td>
</tr>
<tr>
<td>2</td>
<td>Non Maskable</td>
<td>NMI_INT (F000:E2C3)</td>
</tr>
<tr>
<td>3</td>
<td>Breakpoint</td>
<td>None</td>
</tr>
<tr>
<td>4</td>
<td>Overflow</td>
<td>None</td>
</tr>
<tr>
<td>5</td>
<td>Print Screen</td>
<td>PRINT_SCREEN (F000:FF54)</td>
</tr>
<tr>
<td>6</td>
<td>Unused</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Unused</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Time of Day</td>
<td>TIMER_INT (F000:FEA5)</td>
</tr>
<tr>
<td>9</td>
<td>Keyboard</td>
<td>KB_INT (F000:E987)</td>
</tr>
<tr>
<td>10</td>
<td>Diskette</td>
<td>DISK_INT (F000:EF57)</td>
</tr>
<tr>
<td>11</td>
<td>Video</td>
<td>VIDEO_1_0 (F000:F065)</td>
</tr>
<tr>
<td>12</td>
<td>Equipment Check</td>
<td>EQUIPMENT (F000:F84D)</td>
</tr>
<tr>
<td>13</td>
<td>Memory</td>
<td>MEMORY_SIZE_DETERMINE (F000:F841)</td>
</tr>
<tr>
<td>14</td>
<td>Diskette</td>
<td>DISKETTE_1_0 (F000:EC59)</td>
</tr>
<tr>
<td>15</td>
<td>Communications</td>
<td>RS232_1_0 (F000:E739)</td>
</tr>
<tr>
<td>16</td>
<td>Cassette</td>
<td>CASSETTE_1_0 (F000:F859)</td>
</tr>
<tr>
<td>17</td>
<td>Keyboard</td>
<td>KEYBOARD_1_0 (F000:E82E)</td>
</tr>
<tr>
<td>18</td>
<td>Printer</td>
<td>PRINTER_1_0 (F000:EF02)</td>
</tr>
<tr>
<td>19</td>
<td>Cassette BASIC</td>
<td>BOOTSTRAP (F000:EF6F)</td>
</tr>
<tr>
<td>20</td>
<td>Time of Day</td>
<td>TIME_OF_DAY (F000:FE6E)</td>
</tr>
<tr>
<td>1B User Supplied</td>
<td>Keyboard Break</td>
<td>DUMMY_RETURN (F000:FF53)</td>
</tr>
<tr>
<td>1C Routines</td>
<td>Timer Tick</td>
<td>DUMMY_RETURN (F000:FF53)</td>
</tr>
<tr>
<td>1D BIOS Parameters</td>
<td>Video Initialization</td>
<td>VIDEO_PARMS (F000:F0A4)</td>
</tr>
<tr>
<td>1E BIOS Parameters</td>
<td>Diskette Parameters</td>
<td>DISK_BASE (F000:EFC7)</td>
</tr>
<tr>
<td>1F</td>
<td>Video Graphics Chars</td>
<td>None</td>
</tr>
</tbody>
</table>

3-3
Vectors With Special Meanings

Interrupt 1BH - Keyboard Break Address

This vector points to the code to be exercised when the CTRL BREAK keys are depressed on the keyboard. The vector is invoked while responding to the keyboard interrupt, and control should be returned via an IRET instruction. The power on routines initialize this vector to point to an IRET instruction, so that nothing happens when CTRL BREAK keys are depressed 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 set to the 8259 controller. Also, all I/O devices should be reset in case an operation was underway at that time.

Interrupt 1CH - Timer Tick

This vector points to the code to be executed on every tick of the system clock. This vector is invoked while responding to the timer interrupt, and control should be returned via an IRET instruction. The power on routines initialize this vector to point to an IRET instruction, so that nothing happens unless the application modifies the pointer. It is the responsibility of the application to save and restore all registers that will be modified.

Interrupt 1DH - 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 routine.

Interrupt 1EH - 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 to reflect the specifications of the other drives attached may be necessary.
Interrupt 1FH - Graphics Character Extensions

When operating in the graphics modes of the Color/Graphics Monitor Adapter (320 x 200 or 640 x 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 other 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 8 bytes of graphic information. At power on this vector is initialized to 0:0, and it is the responsibility of the user to change this vector if the additional code points are required.

Other Read/Write Memory Usage

The IBM ROM BIOS routines use 256 bytes of memory starting at absolute 400 to 4FF. Locations 400-407 contain the base addresses of any RS232 cards attached to the system, 0’s if none attached. These locations, in order, represent the 0 to 3 values used as the parameter to the RS232 BIOS routine. Locations 408-40F provide the same function, but for the PRINTER.

Memory locations 300-3FF are used as a stack area during the power on initialization, and the bootstrap, when control passed to it from power on. If the user desires the stack in a different area, it must be set by the application.

Note: Use the Interrupt Vector Listing as an aid to locate these topics in the ROM BIOS listing, Appendix “A”.

BIOS Programming Tip

When programming with BIOS you should keep in mind that if an error is reported by the diskette code, to reset the diskette adapter and retry the operation. A specified number of retries should be required on reads to ensure the problem is not due to motor start-up.
BIOS Memory Map

Figure 24. BIOS MEMORY MAP
BIOS Cassette Logic
Software Algorithms

Interrupt 15

The cassette routine will be called with the request type in AH and the address of the bytes to be read or written will be specified by (ES):(BX) and the number of bytes to read/write will be specified by (CX). The actual number of bytes read will be returned in (DX). Read block and write block will automatically turn the motor on at the start and off at the end. The requests are as follows:

(AH) = 0  Turn the cassette motor on.
(AH) = 1  Turn the cassette motor off.
(AH) = 2  (Read Block ) Read (CX) bytes into memory beginning at address (ES):(BX) and return actual number of bytes read in (DX). Return the cassette status in (AH).
(AH) = 3  (Write Block) Write (CX) bytes onto the cassette beginning at address (DS):(BX). Return the cassette status in (AH).

STATUS:

AH = 00  No errors
AH = 01  CRC-Error (Read Block)
AH = 02  No data transitions
AH = 04  No leader
AH = 80  Invalid command
Note: The carry flag will be set on any error.

Cassette Write

The WRITE BLOCK routine writes a tape block on the cassette. The tape block is described in Data Record Architecture page (3-10).

The WRITE BLOCK routine turns on the cassette motor and a synchronization bit (0) and then writes 256 bytes of all ones, the leader, to the tape. Next, one or more data blocks are written (depends on number in CX). After each data block of 256 bytes, a two byte CRC is written. The data bytes are taken from the memory location pointed at by ES.

The WRITE BYTE routine disassembles the byte and writes it a bit at a time to the cassette. The method used is to set TIMER 2 to the period of the desired data bit. The timer is set to a period of 1.0 millisecond for a one bit and 0.5 millisecond for a zero bit.
The timer is set mode 3 which means it will output a square wave with period given by its count register. The timer's period is changed on the fly for each data bit to be written to the cassette. If the number of data bytes to be written is not an integral multiple off 256, then after the last desired data byte from memory has been written, the data block will be extended to 256 bytes by writing multiples of the last data byte. The last block will be closed with two CRC bytes as usual. After the last data block, a trailer consisting of four bytes of all one bits will be written. Finally, the motor will be turned off. There are no errors reported by this routine.

Cassette Read

The READ BLOCK routine turns on the cassette motor and then delays for approximately 0.5 secs for it to come up to speed.

The READ BLOCK routine then searches for leader and must detect all one bits for approximately 1/4 of leader length before it can look for the sync byte. If a correct sync byte (X '16') is not found, the routine goes back and searches for leader again. The data is read a bit at a time and assembled into bytes. After each byte is assembled it is written into memory at location ES:BX and then BX is incremented by one.

After each multiple of 256 data bytes are read, the CRC is read and compared to the CRC generated. If a CRC error is detected, the routine will exit with the carry flag set to indicate an error and status (AH) - 01 for CRC error. DX will contain the number of bytes written into memory.

Note: The Time of Day Interrupt (IRQ0) is disabled during the cassette read operation.
Data Record Architecture

<table>
<thead>
<tr>
<th>LEADER</th>
<th>SYNC BIT</th>
<th>SYNC BYTE</th>
<th>DATA BLOCK</th>
<th>CRC</th>
<th>DATA BLOCK</th>
<th>CRC</th>
</tr>
</thead>
</table>

Motor On

- Leader 256 bytes (of ones)
- Sync byte ASCII Sync Char (X‘16’)
- Sync byte (X ‘16’)
- Data Blocks 256 bytes
- CRC -- 2 bytes -- for each data block

Error Recovery

Error recovery is handled by software. A cyclic redundancy check (CRC) is used to detect errors. The polynomial used is:

\[ G(X) = X^{16} \ll X^{12} \ll X^5 \ll X \]

Which is the polynomial used by the SDLC interface. Essentially, as bits are written/read from tape, they are passed through the CRC-register in software. After a block of data is written, the complemented value of the calculated CRC-register is written on tape. On reading the cassette data, the CRC bytes are read and compared to the generated CRC value. If the read CRC does not equal the generated one, the processor’s carry flag is set and status (AH) is set to X‘01’ to indicate a CRC error has occurred. Also, the routine is exited on CRC error.
Keyboard Encoding and Usage

Encoding

The keyboard routine provided by IBM in 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 0-255, an extended code for certain extended keyboard functions and functions that are 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 exact codes. Use keyboard Scan Code diagram for reference page 2-17.

Table 25. Character Codes

<table>
<thead>
<tr>
<th>KEY #</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>½</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>@</td>
<td>NULL (000)</td>
<td>Note 1</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>#</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>$</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>%</td>
<td>-1</td>
<td></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></td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>*</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>(</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>)</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Backspace (008)</td>
<td>US (031)</td>
<td>Note 1</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Backspace (008)</td>
<td>US (031)</td>
<td>Note 1</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>+</td>
<td>-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>q</td>
<td>Q</td>
<td>DC1 (017)</td>
<td>Note 1</td>
</tr>
<tr>
<td>16</td>
<td>w</td>
<td>W</td>
<td>ETB (023)</td>
<td>Note 1</td>
</tr>
<tr>
<td>17</td>
<td>e</td>
<td>E</td>
<td>ENQ (005)</td>
<td>Note 1</td>
</tr>
<tr>
<td>18</td>
<td>r</td>
<td>R</td>
<td>DC2 (018)</td>
<td>Note 1</td>
</tr>
<tr>
<td>19</td>
<td>t</td>
<td>T</td>
<td>DC4 (020)</td>
<td>Note 1</td>
</tr>
<tr>
<td>20</td>
<td>y</td>
<td>Y</td>
<td>EM (025)</td>
<td>Note 1</td>
</tr>
<tr>
<td>21</td>
<td>u</td>
<td>U</td>
<td>NAK (021)</td>
<td>Note 1</td>
</tr>
<tr>
<td>22</td>
<td>i</td>
<td>I</td>
<td>HT (009)</td>
<td>Note 1</td>
</tr>
<tr>
<td>23</td>
<td>o</td>
<td>O</td>
<td>SI (015)</td>
<td>Note 1</td>
</tr>
<tr>
<td>24</td>
<td>p</td>
<td>P</td>
<td>DLE (016)</td>
<td>Note 1</td>
</tr>
<tr>
<td>25</td>
<td>[</td>
<td>[</td>
<td>ESC (027)</td>
<td>-1</td>
</tr>
<tr>
<td>26</td>
<td>]</td>
<td>]</td>
<td>GS (029)</td>
<td>-1</td>
</tr>
<tr>
<td>KEY #</td>
<td>BASE CASE</td>
<td>UPPER CASE</td>
<td>CTRL</td>
<td>ALT</td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
<td>------------</td>
<td>------------</td>
<td>-----</td>
</tr>
<tr>
<td>28</td>
<td>CR</td>
<td>CR</td>
<td>LF (010)</td>
<td>-1</td>
</tr>
<tr>
<td>29</td>
<td>CTRL -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</td>
<td>SHIFT -1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>43</td>
<td>\</td>
<td>}</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>/</td>
<td>&lt;</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>52</td>
<td>.</td>
<td>&gt;</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</td>
<td>SHIFT -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</td>
<td>ALT -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>CAPS -1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

Note 1: Refer to Extended Codes Page (3-13).

Note 2: Refer to Special Handling Page (3-15).
Keys 71-83 have meaning only in base case, in NUMLOCK (or shifted) states, or in CTRL state. It should be noted that the shift key temporarily reverses the current NUMLOCK state.

<table>
<thead>
<tr>
<th>KEY #</th>
<th>NUM LOCK</th>
<th>BASE CASE</th>
<th>ALT</th>
<th>CTRL</th>
</tr>
</thead>
<tbody>
<tr>
<td>71</td>
<td>7</td>
<td>Home (Note 1)</td>
<td>Note 1</td>
<td>Clear Screen</td>
</tr>
<tr>
<td>72</td>
<td>8</td>
<td>↑ (Note 1)</td>
<td>Note 1</td>
<td>-1</td>
</tr>
<tr>
<td>73</td>
<td>9</td>
<td>PageUp (Note 1)</td>
<td>Note 1</td>
<td>Top of Text &amp; Home</td>
</tr>
<tr>
<td>74</td>
<td>-</td>
<td>-</td>
<td>Note 1</td>
<td>-1</td>
</tr>
<tr>
<td>75</td>
<td>4</td>
<td>← (Note 1)</td>
<td>Note 1</td>
<td>Reverse Word (Note 1)</td>
</tr>
<tr>
<td>76</td>
<td>5</td>
<td>-1</td>
<td>Note 1</td>
<td>-1</td>
</tr>
<tr>
<td>77</td>
<td>6</td>
<td>→ (Note 1)</td>
<td>Note 1</td>
<td>Adv Word (Note 1)</td>
</tr>
<tr>
<td>78</td>
<td>+</td>
<td>+</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>79</td>
<td>1</td>
<td>End (Note 1)</td>
<td>Note 1</td>
<td>Erase to EOL (Note 1)</td>
</tr>
<tr>
<td>80</td>
<td>2</td>
<td>↓ (Note 1)</td>
<td>Note 1</td>
<td>-1</td>
</tr>
<tr>
<td>81</td>
<td>3</td>
<td>PageDown (Note 1)</td>
<td>Note 1</td>
<td>Erase to EOS (Note 1)</td>
</tr>
<tr>
<td>82</td>
<td>0</td>
<td>INS</td>
<td>Note 1</td>
<td>-1</td>
</tr>
<tr>
<td>83</td>
<td>.</td>
<td>DEL (Notes 1,2)</td>
<td>Note 2</td>
<td>Note 2</td>
</tr>
</tbody>
</table>

Note 1: Refer to Extended Codes Page (3-13).
Note 2: Refer to Special Handling Page (3-15).

Extended Codes

A. Extended Functions

For certain functions that can not 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 26. Keyboard Extended Functions

<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-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 &amp; 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 &amp; Home Cursor</td>
</tr>
<tr>
<td>82</td>
<td>INS</td>
</tr>
<tr>
<td>83</td>
<td>DEL</td>
</tr>
<tr>
<td>84-93</td>
<td>F11-F20 (Upper Case F1-F10)</td>
</tr>
<tr>
<td>94-103</td>
<td>F21-F30 (CTRL F1-F10)</td>
</tr>
<tr>
<td>104-113</td>
<td>F31-F40 (ALT F1-F10)</td>
</tr>
<tr>
<td>114</td>
<td>CTRL PRTSC (Start/Stop Echo to Printer) Key 55</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 EOL</td>
</tr>
<tr>
<td>118</td>
<td>CTRL PG DN Erase 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 PG UP TOP 25 Lines of Text &amp; Home Cursor</td>
</tr>
</tbody>
</table>

B. Shift States

Most shift states are handled within the keyboard routine transparently 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:


CTRL - Temporarily shifts keys 3, 7, 12, 14, 16-28, 30-38, 43-50, 55, 59-71, 73, 75, 77, 79, 81 to CTRL state. Used with ALT and DEL to cause “system reset” function described in Section I.3. Used with SCROLL LOCK to cause “break” function described in Section I.3. Used with NUMLOCK to cause “pause” function described in Section I.3.
ALT - Temporarily shifts keys 2-13, 16-25, 30-38, 44-50, and 59-68 to ALT state. Used with CTRL and DEL to cause system reset function described in Section 1.3.

ALT has a special use to allow the user to enter any character code (0-255) into the system from the keyboard. The user holds down the ALT key and types the decimal value of characters using the numeric keyboard (keys 71-73, 75-77, 79-82). The ALT key is then released. If more than three digits are typed, a modulo 256 result is created. These three keys are interpreted as a character code (000-255) and are transmitted through the keyboard routine to the system or application program. ALT is handled internal to keyboard routine.

CAPS LOCK - Shifts keys 16-25, 30-38, 44-50 to upper case. A second depression of CAPS LOCK reverses the action. Handled internal to keyboard routine.

NUM LOCK - Shifts keys 71-73, 75-77, 79-83 to numeric state. A second depression of NUM LOCK reverses the action. Handled internal to keyboard routine.

SCROLL LOCK - Interpreted by appropriate application programs as indicating that the use of the cursor control keys should cause windowing over the text rather than cursor movement. A second depression of SCROLL LOCK reverses the action. The keyboard routine simply records the current shift state of SCROLL LOCK. It is up to the system or application program to perform the function.

C. Shift Key Priorities and Combinations

If combinations of ALT, CTRL and SHIFT are pressed and only one is valid, the precedence is as follows: Highest is ALT, then CTRL, then SHIFT. The only valid combination is ALT CTRL, which is used in system reset.

Special Handling

A. System Reset

The combination of ALT CTRL DEL (Key 83) will result in the keyboard routine initiating the equivalent of a system reset/reboot. Handled internal to keyboard routine.
B. Break

The combination CTRL BREAK will result in the keyboard routine signaling interrupt -1A. Also, the extended characters (AL = 00H, AH = 00H) will be returned.

Power up initialization, this interrupt is set up to cause the break sequence to be ignored. It is up to the system or application initialization code to change the interrupt vector in order to support an actual "break" function.

C. Pause

The combination CTRL NUM-LOCK will cause the keyboard interrupt routine to loop, waiting for any key except NUM-LOCK to be pressed. This provides a system/application transparent method of suspending list/print/etc. temporarily, and then resuming. The "Unpause" key is thrown away. Handled internal to keyboard routine.

D. The following keys will have their typematic action suppressed by the keyboard routine: CTRL, SHIFT, ALT, NUM-LOCK, SCROLL-LOCK, CAPS LOCK, INS.

E. Print Screen

The combination SHIFT-PRINT SCREEN (Key 55) will result in an interrupt invoking the print screen routine. This routine works in alpha/graphics mode, with unrecognizable characters printing as blanks.

The keyboard routine does its own buffering. The buffer is big enough to support a fast typist. If a key is entered when the buffer is full, the key will be ignored and the "bell" will be sounded.
Keyboard Usage

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

Table 27. Keyboard - 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>↑</td>
<td>Full screen editor, word processor</td>
</tr>
<tr>
<td>Page up, scroll backwards 25 lines &amp; home</td>
<td>PG UP</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</td>
<td>END</td>
<td>Editors; word processors</td>
</tr>
<tr>
<td>Place cursor at end of line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move cursor down</td>
<td>↓</td>
<td>Full screen editor, word processor</td>
</tr>
<tr>
<td>Page down, scroll forwards 25 lines &amp; home</td>
<td>PG DN</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, etc</td>
</tr>
<tr>
<td>Start/Stop Echo screen to printer</td>
<td>PRTSC</td>
<td>Any time</td>
</tr>
<tr>
<td></td>
<td>CTRL K55</td>
<td></td>
</tr>
<tr>
<td>Delete from cursor to EOS</td>
<td>CTRL PG DN</td>
<td>Text, command entry</td>
</tr>
<tr>
<td>FUNCTION</td>
<td>KEY(S)</td>
<td>COMMENT</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------</td>
<td>-------------------------------------------------------------------------</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>
<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, NUMLOCK</td>
<td>Stop list, stop program, etc. 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 PG UP</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 stickers 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>
### Table 28. BASIC Screen Editor Special Functions

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>KEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carriage return</td>
<td>←</td>
</tr>
<tr>
<td>Line feed</td>
<td>CTRL ←</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 NUMLOCK</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>

### Table 29. DOS Special Functions

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>KEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suspend</td>
<td>CTRL NUMLOCK</td>
</tr>
<tr>
<td>Echo to printer</td>
<td>CTRL-PRTSC</td>
</tr>
<tr>
<td>(Key 55 any case)</td>
<td></td>
</tr>
<tr>
<td>Stop echo to printer</td>
<td>CTRL-PRTSC</td>
</tr>
<tr>
<td>(Key 55 any case)</td>
<td></td>
</tr>
<tr>
<td>Exit current function</td>
<td>CTRL BREAK</td>
</tr>
<tr>
<td>(break)</td>
<td></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 till 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>
### Low Memory Maps (0-'0600'x)

#### Table 30. Interrupt Vectors (0-7F)

<table>
<thead>
<tr>
<th>ADDRESS HEX</th>
<th>INTERRUPT HEX</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>0–3</td>
<td>0</td>
<td>Divide by Zero</td>
</tr>
<tr>
<td>4–7</td>
<td>1</td>
<td>Single step</td>
</tr>
<tr>
<td>8–B</td>
<td>2</td>
<td>Non-Maskable Interrupt (NMI)</td>
</tr>
<tr>
<td>C–F</td>
<td>3</td>
<td>Break Point Instruction ('CC'x)</td>
</tr>
<tr>
<td>10–13</td>
<td>4</td>
<td>Overflow</td>
</tr>
<tr>
<td>14–17</td>
<td>5</td>
<td>Print Screen</td>
</tr>
<tr>
<td>18–1F</td>
<td>6,7</td>
<td>Reserved</td>
</tr>
<tr>
<td>20–23</td>
<td>8</td>
<td>Timer (18.2 per second)</td>
</tr>
<tr>
<td>24–27</td>
<td>9</td>
<td>Keyboard Interrupt</td>
</tr>
<tr>
<td>28–37</td>
<td>A,B,C,D</td>
<td>Reserved</td>
</tr>
<tr>
<td>38–3B</td>
<td>E</td>
<td>Diskette Interrupt</td>
</tr>
<tr>
<td>3C–3F</td>
<td>F</td>
<td>Reserved</td>
</tr>
<tr>
<td>40–43</td>
<td>10</td>
<td>Video I/O Call</td>
</tr>
<tr>
<td>44–47</td>
<td>11</td>
<td>Equipment Check Call</td>
</tr>
<tr>
<td>48–4B</td>
<td>12</td>
<td>Memory Check Call</td>
</tr>
<tr>
<td>4C–4F</td>
<td>13</td>
<td>Diskette I/O Call</td>
</tr>
<tr>
<td>50–53</td>
<td>14</td>
<td>RS232 I/O Call</td>
</tr>
<tr>
<td>54–57</td>
<td>15</td>
<td>Cassette I/O Call</td>
</tr>
<tr>
<td>58–5B</td>
<td>16</td>
<td>Keyboard I/O Call</td>
</tr>
<tr>
<td>5C–5F</td>
<td>17</td>
<td>Printer I/O Call</td>
</tr>
<tr>
<td>60–63</td>
<td>18</td>
<td>ROM Basic Entry Code</td>
</tr>
<tr>
<td>64–67</td>
<td>19</td>
<td>Boot Strap Loader</td>
</tr>
<tr>
<td>68–6B</td>
<td>1A</td>
<td>Time of Day Call</td>
</tr>
<tr>
<td>6C–6F</td>
<td>1B</td>
<td>Get Control on Keyboard Break: Note 1</td>
</tr>
<tr>
<td>70–73</td>
<td>1C</td>
<td>Get Control on timer interrupt: Note 1</td>
</tr>
<tr>
<td>74–77</td>
<td>1D</td>
<td>Pointer to video initialization table: Note 2</td>
</tr>
<tr>
<td>78–7B</td>
<td>1E</td>
<td>Pointer to diskette parameter table: Note 2</td>
</tr>
<tr>
<td>7C–7F</td>
<td>1F</td>
<td>Pointer to table (1KB) for graphics character Generator for ASCII 128–255. Defaults to 0:0</td>
</tr>
</tbody>
</table>

**Notes:**

1. Initialized at power up to point to an IRET Instruction.
2. Initialized at power up to point to tables in ROM.
Table 31. BASIC and DOS Reserved Interrupts (80-3FF)

<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>88–8B</td>
<td>22</td>
<td>DOS Terminate Address</td>
</tr>
<tr>
<td>8C–8F</td>
<td>23</td>
<td>DOS CTRL–BRK 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>A0–FF</td>
<td>28–3F</td>
<td>Reserved for DOS</td>
</tr>
<tr>
<td>100–1FF</td>
<td>40–7F</td>
<td>Not Used</td>
</tr>
<tr>
<td>200–217</td>
<td>80–85</td>
<td>Reserved By BASIC</td>
</tr>
<tr>
<td>218–3C3</td>
<td>86–F0</td>
<td>Used by BASIC Interpreter while BASIC is Running.</td>
</tr>
<tr>
<td>3C4–3FF</td>
<td>F1–FF</td>
<td>Not Used</td>
</tr>
</tbody>
</table>

Table 32. Reserved Memory Locations (400-5FF)

<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>490–4CF</td>
<td>DOS</td>
<td>Used by DOS Mode Command</td>
</tr>
<tr>
<td>4D0–4EF</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>Reserved for DOS and BASIC</td>
</tr>
<tr>
<td>500</td>
<td>DOS</td>
<td>Print Screen status flag store. 0—Print screen not active or successful print screen operation. 1—Print screen in progress. 255—Error encountered during print screen operation.</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>
BASIC Workspace Variables

If you do DEF SEG (Default workspace segment)

<table>
<thead>
<tr>
<th>OFFSET</th>
<th>LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line number of current line being executed</td>
<td>X '2E' 2</td>
</tr>
<tr>
<td>Line number of last error</td>
<td>X '347' 2</td>
</tr>
<tr>
<td>Offset into segment of start of program text</td>
<td>X '30' 2</td>
</tr>
<tr>
<td>Offset into of start of variables (end of program text 1-1)</td>
<td>X '358' 2</td>
</tr>
<tr>
<td>Keyboard buffer contents</td>
<td></td>
</tr>
<tr>
<td>if 0—no characters in buffer</td>
<td>X '6A' 1</td>
</tr>
<tr>
<td>if 1—characters in buffer</td>
<td></td>
</tr>
<tr>
<td>if you POKE &amp; H6A, 0 you</td>
<td></td>
</tr>
<tr>
<td>flush any characters in buffer</td>
<td></td>
</tr>
</tbody>
</table>

Example:

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

<table>
<thead>
<tr>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>X '64'</td>
<td>X '00'</td>
</tr>
</tbody>
</table>
APPENDICES

Contents:
Appendix A: ROM BIOS Listing

Appendix B: Assembly Instruction Set Reference

Appendix C: OT Characters, Keystrokes and Colors

Appendix D: Logic Diagrams

Appendix E: Unit Specifications
## CONTENTS

<table>
<thead>
<tr>
<th>ITEM</th>
<th>LINE NUMBER</th>
<th>ADDRESS</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equates and Data Areas</td>
<td>1</td>
<td>----</td>
<td>A-2</td>
</tr>
<tr>
<td>Power-on Self-test</td>
<td>198</td>
<td>E016</td>
<td>A-4</td>
</tr>
<tr>
<td>Boot Strap Loader</td>
<td>1355</td>
<td>E6F2</td>
<td>A-20</td>
</tr>
<tr>
<td>I/O Support</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Asynchronous Communications</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(RS 232) I/O</td>
<td>1410</td>
<td>E729</td>
<td>A-20</td>
</tr>
<tr>
<td>Keyboard I/O</td>
<td>1640</td>
<td>E82E</td>
<td>A-23</td>
</tr>
<tr>
<td>Diskette I/O</td>
<td>2255</td>
<td>EC59</td>
<td>A-32</td>
</tr>
<tr>
<td>Printer I/O</td>
<td>3007</td>
<td>EFD2</td>
<td>A-42</td>
</tr>
<tr>
<td>Display (VIDEO) I/O</td>
<td>3119</td>
<td>F045</td>
<td>A-43</td>
</tr>
<tr>
<td>Cassette I/O</td>
<td>4977</td>
<td>F859</td>
<td>A-68</td>
</tr>
<tr>
<td>System Configuration Analysis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory-Size-Determination</td>
<td>4903</td>
<td>F841</td>
<td>A-67</td>
</tr>
<tr>
<td>Equipment-Determination (Options)</td>
<td>4933</td>
<td>F84D</td>
<td>A-67</td>
</tr>
<tr>
<td>Graphics-Character Generator</td>
<td>5503</td>
<td>FA6E</td>
<td>A-75</td>
</tr>
<tr>
<td>Time-of-day</td>
<td>5642</td>
<td>FE6E</td>
<td>A-77</td>
</tr>
<tr>
<td>Print Screen</td>
<td>5817</td>
<td>FF53</td>
<td>A-79</td>
</tr>
<tr>
<td>Notes for the BIOS Listing</td>
<td></td>
<td></td>
<td>A-81</td>
</tr>
</tbody>
</table>
LOC OBJ

LINE SOURCE
1 TITLE(IBM BIOS FOR IBM PERSONAL COMPUTER)
2 "--------------------------
3 |================================
4 |
5 0060 5 PORT_A EQU 60H ;8255 PORT A ADDR
6 0061 6 PORT_B EQU 61H ;8255 PORT B ADDR
7 0062 7 PORT_C EQU 62H ;8255 PORT C ADDR
8 0063 8 CMD_PORT EQU 63H
9 0020 9 INTAD0 EQU 20H ;18259 PORT
10 0021 10 INTAD1 EQU 21H ;18259 PORT
11 0020 11 EOI EQU 20H
12 0040 12 TIMER EQU 40H
13 0043 13 TIM_CTL EQU 43H ;8255 Timer Control Port ADDR
14 0040 14 TIMPER EQU 40H ;8255 Timer/Counter 0 Port ADDR
15 0081 15 THINT EQU 01 ;Timer 0 Interrupt Mask
16 0008 16 DMA05 EQU 08 ;DMA Status Reg Recv Port ADDR
17 0000 17 DMA EQU 00 ;DMA Channel 0 Address Reg Port ADDR
18 0040 18 MAX_PERIOD EQU 540H
19 0040 19 MIN_PERIOD EQU 410H ?
20 0060 20 KBM_INTR EQU 60H ;Keyboard Data in Addr Port
21 0062 21 KBMINT EQU 02 ;Keyboard Interrupt Mask
22 0060 22 KB_DATA EQU 60H ;Keyboard Scan Code Port
23 0061 23 KB_CTL EQU 61H ;Control bits for Keyboard Sense Data
24 |
25 | ; 000B INTERRUPT LOCATIONS
26 |
27 0000 27 ABS0 SEGMENT AT 0
28 0000 28 START EQU LABEL BYTE
29 0008 29 ORG 2*4
30 0008 30 NMI_PTR EQU LABEL WORD
31 0014 31 ORG 5*4
32 0014 32 INTS_PTR EQU LABEL WORD
33 0020 33 ORG 8*4
34 0020 34 INT_ADDR EQU LABEL WORD
35 0020 35 INT_PTR EQU LABEL DWORD
36 0040 36 ORG 10H*4
37 0040 37 VIDEO_INT EQU LABEL WORD
38 0074 38 ORG 11H*4
39 0074 39 PARM_PTR EQU LABEL DWORD ;Pointer to VideoParms
40 0078 40 ORG 01EH*4 ;Interrupt IEH
41 0078 41 DISM_POINTER EQU LABEL DWORD
42 007C 42 ORG 01FH*4 ;Location of Pointer
43 007C 43 EXT_PTR EQU LABEL DWORD ;Pointer to Extension
44 7C00 44 ORG 7C00H
45 7C00 45 BOOT_LOCN EQU LABEL FAR
46 4E10 46 ABS0 ENDS
47 |
48 | ;---------------------------
49 | ; STACK -- USED DURING INITIALIZATION ONLY
50 | ;---------------------------
51 | STACK SEGMENT AT 30H
52 |
53 0000 (120 ???) 52 DW 120 DUPH ?
54 0100 53 TOS LABEL WORD
55 54 STACK ENDS
56 |
57 | ; ROM BIOS DATA AREAS
58 | ;--------------------------
59 0040 59 DATA SEGMENT AT 40H
60 0000 (4 ???) 60 RS232_BASE DW 4 DUPH ? ;Addresses of RS232 Adapters
61 0000 (4 ???) 61 PRINTER_BASE DW 4 DUPH ? ;Addresses of Printers
62 0010 ?? 62 EQUIP_FLAG DW ? ;Installed Hardware
63 0012 ?? 63 HFG_TST DB ? ;Initialization Flag
64 0013 ?? 64 MEMORY_SIZE DW ? ;Memory Size in K Bytes
65 0015 ?? 65 IO_RAM_SIZE DW ? ;Memory in I/O Channel
66 |
67 | ; KEYBOARD DATA AREAS
68 |
69 0017 ?? 69 KB_FLAG DB ?
70 |
71 | ; ------- SHIFT FLAG EQUATES WITHIN KB_FLAG
72 |
73 0050 73 INS_STATE EQU 00H ;Insert State is Active
74 0050 74 CAPS_STATE EQU 40H ;Caps Lock State Has Been Toggled
75 0050 75 NUM_STATE EQU 20H ;Num Lock State Has Been Toggled
76 0050 76 SCROLL_STATE EQU 10H ;Scroll Lock State Has Been Toggled
77 0050 77 ALT_SHIFT EQU 08H ;Alternate Shift Key Depressed
78 |
79 |
80 |
81 |
82 A-2
LOC OBJ   LNE   SOURCE

004A   70   CTI_SLFT  EQU   04H   ; CONTROL SHIFT KEY DEPRESSED
0002   79   LEFT_SFT  EQU   02H   ; LEFT SHIFT KEY DEPRESSED
0001   80   RIGHT_SFT  EQU   01H   ; RIGHT SHIFT KEY DEPRESSED
01
001B ??  82   KB_FLAG_1  DB   ?   ; SECOND BYTE OF KEYBOARD STATUS
0000   84   IHS_SFT  EQU   00H   ; INSERT KEY IS DEPRESSED
0040   85   CAPS_SFT  EQU   04H   ; CAPS LOCK KEY IS DEPRESSED
0020   86   NUM_SFT  EQU   20H   ; NUM LOCK KEY IS DEPRESSED
0010   87   SCROLL_SFT  EQU   10H   ; SCROLL LOCK KEY IS DEPRESSED
0000   88   HOLD_STATE  EQU   00H   ; SUSPEND KEY HAS BEEN Toggled
01
0019 ??  90   ALT_INPUT  DB   ?   ; STORAGE FOR ALTERNATE KEYPAD ENTRY
001A ??  91   BUFFER_HEAD  DW   ?   ; POINTED TO HEAD OF KEYBOARD BUFFER
001C ??  92   BUFFER_TAIL  DW   ?   ; POINTED TO TAIL OF KEYBOARD BUFFER
001E (16 ???? )   93   KB_BUFFER  DW   16 DUP(?)   ; ROOM FOR 16 ENTRIES
003E   94   KB_BUFFER_END  LABEL  WORD
95
96   ----- HEAD = TAIL INDICATES THAT THE BUFFER IS EMPTY
97
0045   98   NUM_KEY  EQU   69   ; SCAN CODE FOR NUMBER LOCK
0046   99   SCROLL_KEY  EQU   70   ; SCAN CODE FOR SCROLL KEY
0030   100   TIME_KEY  EQU   56   ; ALTUNATE SFT KEY SCAN CODE
001D   101   CTL_KEY  EQU   29   ; SCAN CODE FOR CONTROL KEY
003A   102   CAPS_KEY  EQU   58   ; SCAN CODE FOR SHIFT LOCK
002A   103   LEFT_KEY  EQU   42   ; SCAN CODE FOR LEFT SHIFT
0036   104   RIGHT_KEY  EQU   54   ; SCAN CODE FOR RIGHT SHIFT
0052   105   INS_KEY  EQU   62   ; SCAN CODE FOR INSERT KEY
0055   106   DEL_KEY  EQU   03   ; SCAN CODE FOR DELETE KEY
107
108   ;----------------------------------------------------------------------------
109   ;  DISKETTE DATA AREAS
110   ;----------------------------------------------------------------------------
005E ??   111   SEEK_STATUS  DB   ?   ; DRIVE RECALIBRATION STATUS
112   ; BIT 0-3 = DRIVE 3-0 NEEDS RECAL BEFORE
113   ; NEXT SEEK IF BIT IS = 0
0080   114   INT_FLAG  EQU   08H   ; INTERRUPT OCCURRENCE FLAG
005F ??   115   MOTOR_STATUS  DB   ?   ; MOTOR STATUS
116   ; BIT 0-3 = DRIVE 3-0 IS CURRENTLY RUNNING
117   ; BIT 7 = CURRENT OPERATION IS A WRITE, REQUIRES DELAY
0040 ??   118   MOTOR_COUNT  DB   ?   ; TIME COUNTER FOR DRIVE TURN OFF
119   ; TWO SECONDS OF COUNTS FOR MOTOR TURN OFF
120
121   ;
0061 ??   122   DISKETTE_STATUS  DB   ?   ; SINGLE BYTE OF RETURN CODE INFO FOR STATUS
0080   123   TIME_OUT  EQU   00H   ; ATTACHMENT FAILED TO RESPOND
0040   124   BAD_SEEK  EQU   40H   ; SEEK OPERATION FAILED
0020   125   BAD_NECE  EQU   20H   ; NEC CONTROLLER HAS FAILED
0010   126   BAD_CRC  EQU   10H   ; BAD CRC ON DISKETTE READ
0009   127   DMA_BOUNDARY  EQU   09H   ; ATTEMPT TO DMA ACROSS 64K Boundary
0008   128   BAD_DMA  EQU   08H   ; DMA OVRUN ON OPERATION
0004   129   REQUESTED_SECTOR_NOT_FOUND  EQU   04H   ; REQUESTED SECTOR NOT FOUND
0003   130   WRITE_PROTECT  EQU   03H   ; WRITE ATTEMPTED ON WRITE PROT DISK
0002   131   BAD_ADDR_MARK  EQU   02H   ; ADDRESS MARK NOT FOUND
0001   132   BAD_CHD  EQU   01H   ; BAD COMMAND PASSED TO DISKETTE I/O
0002 (17 ???? )   134   NEC_STATUS  DB   7 DUP(?)   ; STATUS BYTES FROM NEC
135
136   ;----------------------------------------------------------------------------
137   ;  VIDEO DISPLAY DATA AREA
138   ;----------------------------------------------------------------------------
0049 ??   139   CRT_MODE  DB   ?   ; CURRENT CRT MODE
004A ??  140   CRT_COLDS  DW   ?   ; NUMBER OF COLORS ON SCREEN
004C ??  141   CRT_LEN  DW   ?   ; LENGTH OF REG IN BYTES
004E ??  142   CRT_START  DW   ?   ; STARTING ADDRESS IN REG BUFFER
0050 (1B ???? )   143   CURSOR_POS  DW   8 DUP(?)   ; CURSOR FOR EACH OF UP TO 8 PAGES
0060 ???  144   CURSOR_MODE  DW   ?   ; CURRENT CURSOR MODE SETTING
0062 ??  145   ACTIVATE_PAGE  DB   ?   ; CURRENT PAGE BEING DISPLAYED
0063 ??  146   ADDR_6045  DW   ?   ; BASE ADDRESS FOR ACTIVE DISPLAY CARD
0065 ??  147   MEM_PAGE  DW   ?   ; CURRENT SETTING OF THE 0X1B REGISTER
0066 ??  148   CRT_PALLETT  DB   ?   ; CURRENT PALLETT SETTING COLOR CARD
149
150   ;----------------------------------------------------------------------------
151   ;  CASSSETTE DATA AREA
152   ;----------------------------------------------------------------------------
0067 ??  155   EDGE_CNT  DW   ?   ; TIME COUNTER AT DATA EDGE
0069 ??  154   CRC_REG  DW   ?   ; CURRENT CRC REGISTER
LOC OBJ  LINE  SOURCE

006B ??  155  LAST_VAL DB ?  ;LAST INPUT VALUE
156
157
158
159

006C ??  160  TIMER_LOW DW ?  ;LOW WORD OF TIMER COUNT
006E ??  161  TIMER_HIGH DW ?  ;HIGH WORD OF TIMER COUNT
0070 ??  162  TIMER_OFH DB ?  ;TIMER HAS ROLLED OVER SINCE LAST READ
163  ;COUNTS_SEC EQU 18
164  ;COUNTS_MIN EQU 1092
165  ;COUNTS_HOUR EQU 65543
166  ;COUNTS_DAY EQU 1573040 + 100000
167
168
169
170

0071 ??  171  BIOS_BREAK DB ?  ;BIT 7 = 1 IF BREAK KEY HAS BEEN DEPRESSED
0072 ??  172  RESET_FLAG DW ?  ;WORD = 1234H IF KEYBOARD RESET UNDERWAY
173  DATA ENDS
174
175
176
177

0050  178  XXDATA SEGMENT AT 50H
0000 ??  179  STATUS_BYTE DB ?
180  XXDATA ENDS
181
182
183

D000  184  VIDEO_RAM SEGMENT AT 0000H
185
186  REGEN LABEL BYTE
187  REGEN LABEL WORD
188  (16304 ??) DB 16304 DUP(?)
189  VIDEO_RAM ENDS
190
191
192  ;ROM RESIDENT CODE
193
194

F000  195  CODE SEGMENT AT 0F000H
0000 (57344 ??) DB 57344 DUP(?) ;FILL LOWEST 56K
196
197
198

E000  199  353730303030H ;COPYRIGHT NOTICE
20434F50522E20
4942420231393B
31

020  201  ;INITIAL RELIABILITY TESTS -- PHASE 1
202  ASSUME CS=CODE,SS=CODE,ES=DATA
203  ;DATA DEFINITIONS
204
205

E016 D0E0 R  206  C1 DW DC1 ;RETURN ADDRESS FOR DUMMY STACK
E018 E0E1 R  207  C2 DW DC2 ;RETURN ADDRESS FOR DUMMY STACK
208
209

210  ENTRY REQUIREMENTS:
211  ES = ADDRESS OF STORAGE SEGMENT BEING TESTED
212  DS = ADDRESS OF STORAGE SEGMENT BEING TESTED
213  WHEN ENTERING AT STGST_CNT, CX MUST BE LOADED WITH THE BYTE COUNT.
214
215
216

E01A  217  ;EXPECTED DATA PATTERN VS THE ACTUAL DATA READ.
218
219

E01A  220  STGST PROC NEAR
E01A B90040  221  MOV CX,4000H ;SETUP CNT TO TEST A 16K BLK
E01D  222  STGST_CNT:
E01D FC  223  CLO
E01E 6ED9  224  MOV DX, CX ;SAVE BYTE CNT (4K FOR VIDEO OR 16K)
E020 B0FFF  225  MOV AX, 0FFFH ;GET DATA PATTERN TO WRITE
E023 B55AA  226  MOV DX, 0A55H ;SETUP OTHER DATA PATTERNS TO USE
E026 20FF  227  SUB DI, DI ;DI = OFFSET 0 RELATIVE TO ES:DI
E028 F3  228  REP STOSH ;WRITE STORAGE LOCATIONS
E029 AA

A-4
Appendix A

LOC OBJ LINE SOURCE

E02A 229 C3: 1 STG01
E02A 4F 230 DEC DI 1 POINT TO LAST BYTE JUST WRITTEN
E02B FD 231 STD 1 SET DIR FLAG TO GO ENDMARKS
E02C 0BF7 232 C4: MOV SI,DI 1 SETUP BYTE CHN
E030 AC 234 C5: LODSB 1 READ CHAP FROM STORAGE
E031 32C4 235 XOR AL,AL 1 DATA READ AS EXPECTED?
E033 7255 236 JNE C7 1 NO - GO TO ERROR ROUTINE
E035 4E42 237 IN AL,PORT_C 1 DID A PARITY ERROR OCCUR?
E037 C4C0 238 AND AL,ALCM
E039 B006 239 MOV AL,0 1 AL=0 DATA COMPARE OK
E03B 7510 240 JNZ C7 1 WRITE BACK IN C7
E03D 80FC00 241 CMP AH,0 1 READING ZERO PATTERN?
E040 7493 242 JE C6 1 CONTINUE READING TILL END
E042 80C2 243 MOV AL,DL 1 GET NEXT DATA PATTERN TO WRITE
E044 AA 244 STOSB 1 WRITE IN BYTE LOC WE JUST READ
E045 245 C6: MOV SI,DI 1 WRITE-MORE
E045 E2E9 246 LOOP C5 1 CONTINUE TILL 16K/4K BLOCK TESTED
E047 80FC00 247 CMP AH,0 1 ZERO PATTERN WRITTEN TO STG
E04A 7493 248 JE C7 1 YES - RETURN TO CALLER
E04C 8AEO 249 MOV AH,AL 1 SETUP TO NEW VALUE TO COMPARE
E04E 86F2 250 XCHG DH,DL 1 MOVE ZERO DATA PATTERN TO DL
E050 FC 251 CLD 1 SET DIR FLAG TO GO FORWARD
E051 47 252 INC DI 1 SET POINTER TO BGS LOCATION
E052 7408 253 JE C4 1 READ/WRITE FORWARD IN STG
E054 4F 254 DEC DI 1)
E055 B0100 255 MOV BX,1 1 SETUP SI AND DI PATTERNS
E058 E000 256 JNP SHORT C3 1 READ/WRITE BACKWARD IN STG
E05A 257 C7: RET
E05A C3 258 STGST ENDP

---

E05B 267 START: CLI 1 DISABLE INTERRUPTS
E05C B000 268 MOV AH,000H 1 SET SF, CF, ZF, AND AF FLAGS ON
E05E 9E 269 SAHF
E05F 734E 270 JNC ERR01 1 GO TO ERR ROUTINE IF CF NOT SET
E061 754C 271 JNZ ERR01 1 GO TO ERR ROUTINE IF ZF NOT SET
E063 794A 272 JNC ERR01 1 GO TO ERR ROUTINE IF PF NOT SET
E065 7948 273 JNS ERR01 1 GO TO ERR ROUTINE IF SF NOT SET
E067 9F 274 LAHF 1 LOAD FLAG IMAGE TO AH
E068 B105 275 MOV CL,5 1 LOAD CNT REG WITH SHIFT CNT
E06A 02EC 276 SHR AH,CL 1 SHIFT AF INTO CARRY BIT POS
E06C 7341 277 JNC ERR01 1 GO TO ERR ROUTINE IF AF NOT SET
E06E 8045 278 MOV AL,03H 1 SET THE OF FLAG ON
E070 D0E0 279 MOV AH,1 1 SETUP FOR TESTING
E072 713B 280 JID ERR01 1 GO TO ERR ROUTINE IF DF NOT SET
E074 32E4 281 XOR AH,AH 1 SET AH = 0
E076 9E 282 SAHF 1 CLEAR SF, CF, ZF, AND PF
E077 7236 283 JC ERR01 1 GO TO ERR ROUTINE IF CF ON
E079 7346 284 JZ ERR01 1 GO TO ERR ROUTINE IF SF ON
E07B 7332 285 JS ERR01 1 GO TO ERR ROUTINE IF SF ON
E07D 7A30 286 JP ERR01 1 GO TO ERR ROUTINE IF PF ON
E07F 9F 287 LAHF 1 LOAD FLAG IMAGE TO AH
E080 B185 288 MOV CL,5 1 LOAD CNT REG WITH SHIFT CNT
E082 D2EC 289 SHR AH,CL 1 SHIFT AF' INTO CARRY BIT POS
E084 7229 290 JC ERR01 1 GO TO ERR ROUTINE IF ON
E086 D0E4 291 SHL AH,1 1 CHECK THAT DF' IS CLEAR
E088 7025 292 JO ERR01 1 GO TO ERR ROUTINE IF ON

---

293 1 READ/WRITE THE 8088 GENERAL AND SEGMENTATION REGISTERS
295 1 WITH ALL ONE'S AND ZERO'S.

296 E06A B0FFFF 297 MOV AX,OFFFH 1 SETUP ONE'S PATTERN IN AX
E06D F9 298 STC
E06E E000 299 C6: MOV DS,AX 1 WRITE PATTERN TO ALL REGS
E090 6CB0 300 MOV BA,DS
E092 8EC3 301 MOV ES,DX
E094 6C11 302 MOV CX,ES
E096 8E01 303 MOV SS,CX
E098 8EO2 304 MOV DX,SS
E09A 8EE2 305 MOV SP,DX
E09C 8EAC 306 MOV BP,SP
E09E 86F5 307  MOV SI,RP
E09F 86FE 308  MOV DI,SI
E0A2 7307 309  JNC C9  ; TEST 1
E0A4 35C7 310  MOV AX,DI ; PATTERN MAKE IT THRU ALL REGS
E0A6 7507 311  JNZ ERROR1 ; END - GO TO ERR ROUTINE
E0A8 FF0 312  CLC
E0AA 71E3 313  JNC C8
E0AB C9: 314  JNE C9  ; TEST 1
E0AB 0B07 315  OR AX,DI ; ZERO PATTERN MAKE IT THRU
E0AD 7401 316  JZ C10 ; YES - GO TO NEXT TEST
E0AF F4 317  ERROR1: HLT  ; HALT SYSTEM

1:---------------------------------------------------------------------
20: ; ROS CHECKSUM TEST 1
31: ; DESCRIPTION
32: ; A CHECKSUM IS DONE FOR THE OK ROS MODULE CONTAINING PDB AND BIOS.
33: ;----------------------------------------------------------------------
E0B0
34: C10:
35: E0B0 E000 325  MOV AL,0  ; DISABLE NMI INTERRUPTS
36: E0B2 E6A0 326  OUT 0AH,AL
37: E0B4 E603 327  OUT BX,AL
38: E0B6 B099 328  MOV AL,99H  ; INITIALIZE DMA PAGE REG
39: E0B8 E663 329  OUT COM_PORT,AL  ; WRITE 6255 CMD/MODE REG
40: E0BA 00C7 330  MOV AL,OFCH  ; DISABLE PARITY CHECKERS AND
41: E0BC E661 331  OUT PORT_B,AL  ; GATE SMS SGS,CASS MOTOR OFF
42: E0CC 840 332  SUB AL,AL
43: E0CC BA0003 333  MOV DX,300H
44: E0C3 EE 334  OUT DX,AL  ; DISABLE COLOR VIDEO
45: E0C4 FEC0 335  INC AL
46: E0C6 E4003 336  MOV DX,300H
47: E0C9 EE 337  OUT DX,AL  ; DISABLE DMA VIDEO,EN HIGH REG
48: E0CA BA000F0 338  MOV AX,0000H  ; SETUP SS SEG REG
49: E0CD E800 339  MOV SS,AX
50: E0CE BB0003 350  MOV DX,0000H  ; SETUP STARTING ROS ADDR
51: E0C2 BC160 351  MOV SP,OFFSET C1  ; SETUP RETURN ADDRESS
52: E0D5 93301 352  JMP ROS_CHECKSUM
53: E0DB 7505 353  C11:  ; THE ERROR1: HALT SYSTEM IF ERROR
54: 1:---------------------------------------------------------------------
55: E0D0 E004 354  MOV AL,04  ; DISABLE DMA CONTROLLER
56: E0D0 E608 355  OUT DMA00,AL
57: 356
58: E0DB E054 357  ; VERIFY THAT TIMER 1 FUNCTIONS OK
59: E0DE E643 360  MOV TIMER+3,AL
60: E0EE 2DC9 361  SUB CX,CX
61: E0E4 8A09 362  MOV BL,CL
62: E0E6 8AC1 363  MOV AL,CL  ; SET INITIAL TIMER CNT TO 0
63: E0E8 E641 364  OUT TIMER+1,AL
64: 365
65: E0EA E040 366  MOV AL,40H  ; TIMER_BITS_ON
66: E0EC E643 367  OUT TIMER+3,AL
67: E0EE E441 368  IN AL,TIMER+1 ; READ TIMER 1 COUNT
68: E0F0 0A08 369  OR BL,AL  ; ALL BITS ON IN TIMER
69: E0F2 0BF7 370  CMP BL,OFFH  ; YES - SEE IF ALL BITS GO OFF
70: E0F5 7044 371  JZ C13 ; TIMER_BITS_OFF
71: E0F7 E2F1 372  LOOP C12
72: E0FB E0B4 373  JMP SHORT ERROR1 ; TIMER 1 FAILURE, HALT SYS
73: E0FB 374  C13: ; TIMER_BITS_OFF
74: E0FB EBC3 375  MOV AL,0L  ; SET TIMER 1 CNT
75: E0FD 2BC9 376  SUB CX,CX
76: E0FF E641 377  OUT TIMER+1,AL
77: E100 378
78: E101 E040 379  MOV AL,40H  ; TIMER_BITS_ON
79: E103 E643 380  OUT TIMER+3,AL
80: E105 E441 381  IN AL,TIMER+1 ; READ TIMER 1 COUNT
81: E107 2208 382  AND BL,AL
82: E109 7404 383  JZ C15 ; WRAP_DMA_REG
83: E10B 42F4 384  LOOP C14 ; TIMER_LOOP

A-6
LOC OBJ LINE SOURCE

E100 E8A0 305 JMP SHORT ERR01
306
307 ; INITIALIZE TIMER 1 TO REFRESH MEMORY
308
E10F 309 C15: MOV AL,54H ; WRAP DMA REG
E110 B004 310 OUT TIMER+3,AL ; INITIALIZE TIMER 1 Mode
E111 E643 311 MOV AL,1B ; WRITE TIMER MODE REG
E113 8012 312 OUT TIMER+1,AL ; SETUP DIVISOR FOR REFRESH
E115 E641 313 OUT TIMER+1,AL ; WRITE TIMER 1 CNT REG
E117 6600 314 OUT DMA+0DH,AL ; SEND MASTER CLEAR TO DMA
315
316 ; WRAP DMA CHANNELS ADDRESS AND COUNT REGISTERS

317
E119 B0FF 318 MOV AL,OFFH ; WRITE PATTERN FFH TO ALL REGS
E11B A608 319 MOV BL,AL ; SAVE PATTERN FOR COMPARE
E11C B0F8 320 MOV BH,AL ;
E11F B0000 321 MOV CX,AL ; SETUP LOOP CNT
E122 B0000 322 MOV DX,DX ; SETUP I/O PORT ADDR OF REG
E125 EE 323 MOV CX,AL ; INITIALIZE DMA ADDR OF REG
E126 EE 324 MOV DX,AL ; INITIALIZE DMA ADDR OF REG
E127 B00101 325 MOV AX,0101H ; AX TO ANOTHER PORT BEFORE RD
E12A EC 326 MOV BX,AL ; INITIALIZE DMA ADDR OF REG
E12B AE0 327 MOV AH,AL ; SAVE LSB OF 16-BIT REG
E12D EC 328 MOV AX,AL ; READ 16-BIT DMA CH REG, LSB
E12E 30DB 329 CMP BX,AX ; PATTERN READ AS WRITTEN!
E130 7003 330 JE C18 ; NEXT REG
E132 EY7AF 331 JMP ERR01 ; END - MALT THE SYSTEM
E135 418 332 JE C18 ; INITIALIZE AND START DMA FOR MEMORY REFRESH.
E139 42 333 INC DX ; SET I/O PORT TO NEXT CH REG
E136 E2ED 334 LOOP C17 ; WRITE PATTERN TO NEXT REG
E138 FADD 335 INC AL ; SET PATTERN TO ZERO
E13A 74DF 336 JZ C16 ; WRITE TO CHANNEL REGS
E13F 418 ; INITIALIZE AND START DMA FOR MEMORY REFRESH.
E140 40F 340 MOV AL,OFFH ; SET CNT OF 64K FOR RAM REFRESH
E14E 6001 341 OUT DMA+1AL ; INITIALIZE RAM
E14F 6001 342 OUT DMA+1AL ; INITIALIZE RAM
E152 B05B 343 MOV AL,0B8H ; SET DMA MODE, CH 0, READ, AUDITINT
E154 E50B 344 MOV DX,0B8H ; SET DMA MODE REG
E156 B000 345 MOV AL,0 ; ENABLE DMA CONTROLLER
E158 E608 346 MOV DX,AL ; SETUP DMA COMMAND REG
E15A E60A 347 MOV DX,AL ; ENABLE CHANNEL 0 FOR REFRESH
E15C B041 348 MOV AL,4H ; SET MODE FOR CHANNEL 1
E15E B049 349 MOV DX,AL ; ENABLE CHANNEL 2
E15G B042 350 MOV AL,2H ; SET MODE FOR CHANNEL 2
E15I B008 351 MOV DX,AL ; SET MODE FOR CHANNEL 3
E155 B043 352 MOV AL,43H ; SET MODE FOR CHANNEL 3
E156 B046 353 OUT DMA+0B8H,AL ; INITIALIZE MEMORY SIZE AND FILL MEMORY WITH DATA
445
446 ; DETERMINE MEMORY SIZE AND FILL MEMORY WITH DATA

E158 B64000 446 MOV AX,DATA ; INITIALIZE DMA MODE CH 0, READ, AUDITINT
E159 B600 447 MOV DS,AX ; END - MALT THE SYSTEM
E150 E6117200 448 MOV BX,RESET_FLAG ; INITIALIZE DMA MODE CH 0
E161 2BC0 449 MOV AL,DATA ; SAVE RESET_FLAG IN BX
E163 8EC0 450 MOV AX,DATA ; SET ESA AND DS TO 0
E165 8ED0 451 MOV DS,AX ; STORAGE, VERIFY STORAGE ADDRESSABILITY.
E167 80FF 452 MOV DS,AX ; INITIALIZE THE 0259 INTERRUPT CONTROLLER CHIP FOR CHECKING
E169 E60 453 MOV CX,AX ; MAINTAIN 16K MEMORY IN 16-BIT MODE
E16B 2A0C 454 MOV AL,DATA ; DETERMINE BASE RAM SIZE
E16C 0044 455 MOV AX,DATA ; ISOLATE RAM SIZE SHS
E16D 810C 456 MOV CL,10 ; CALCULATE MEMORY SIZE
E16F 8EC0 457 MOV AL,CL ; DETERMINE BASE RAM SIZE
E170 9EC5 458 MOV AX,CL ; DETERMINE BASE RAM WITH DATA

A-7
E179 E2FD 462 LOOP C19 ; LOOP TIL ALL ZERO
463
E17B E462 464 IN AL,PORT_C
E17D 240F 467 AND AL,0FH
E17F 7610 468 JZ C21
E181 BC0010 469 MOV DX,1000H ; SEGMENT FOR I/O RAM
E184 8AE0 470 MOV AH,AL
E186 B000 471 MOV AL,0
E188 C20 472 C20: ; FILL_ID:
E18A 0ECC 473 MOV ES,DX
E18B B90000 474 MOV CX,8000H ; FILL 32K BYTES
E18D 20FF 475 SUB DI,DI
E190 F5 476 REP STOSB
E194 AA
E195 D20000B 477 ADD DX,8000H ; NEXT SEGMENT VALUE
E195 FECC 478 DEC AH
E197 75EF 479 JNZ C20 ; FILL_ID
E199 408
E19B B013 480 ; INITIALIZE THE 8259 INTERRUPT CONTROLLER CHIP
E19D 0620 481 ;-----------------------------------------
E19F 0D00 482 ; CHECK FOR MANUFACTURING TEST 2 TO LOAD TEST PROGRAMS FROM KEYBOARD.
E1A1 0621 483 ;-----------------------------------------
E1A3 B000 490 MOV SI,DATA ; POINT OS TO DATA SEG
E1A4 06DE 493 MOV DS,DI
E1A6 81E7200 494 MOV AL,0 ; SETUP ICW4 - BUFDRD,8006 Mode
E1A7 8E4000 495 MOV ES,AX ; POINT OS TO 1ST 16K OF STG
E1A9 891E72 496 MOV RESETLAG,BX ; RESTORE RESETLAG
E1A9 813E7200 497 CMP RESETLAG,1234H ; RESETLAG SET?
E1A9 743B 498 JE C25 ; YES - SKIP STG TEST
E1AB 8EDE 499 MOV DS,AX ; POINT DS TO 1ST 16K OF STG
E1AC 8E00 500 MOV SP,3FF0H ; ESTABLISH TEMPORARY STACK
E1A8 600F 501 MOV DX,3FH ; R/W STORAGE
E1A9 8BF8 502 MOV AX,DATA
E1AD 83010H 503 MOV ESP,AX
E1A7 8BF8 504 MOV AX,DATA
E1AC 81E7200 505 MOV BX,BASE ; BASE ADDRESS
E1A8 882400 506 MOV AL,24H
E1A1 882400 507 INC BX
E1A6 882400 508 INC BX
E1A0 882400 509 MOV [BX],CS
E1A8 8B7004 510 CALL KBD_RESET ; READ IN KB RESET CODE TO DL
E1A9 00F605 511 MOV DL,665H ; IS THIS MANUFACTURING TEST 2?
E1A9 750E 512 JNZ C23 ; JUMP IF NOT MAN. TEST
E1B1 B2FF 513 MOV DL,255 ; READ IN TEST PROGRAM
E1B3 E29A04 514 CALL SP_TEST
E1B5 8AC3 515 MOV AL,EL
E1B7 AA 516 STOSB
E1B9 FECA 517 DEC DL
E1B9 7566 518 JNZ C22 ; JUMP IF NOT DONE YET
E1B9 C0DE 519 JIF 3EH ; SET INTERRUPT TYPE 62 ADDRESS F0H
E1B9 1519 520 PUSH CS ; PUT SS BACK
E1B9 17 521 POP SS
E1B9 FA 522 CLI
E1B9 D10E0 523 MOV SP,OFFSET C2 ; SETUP RETURN ADDRESS
E1B9 E20FE 524 JMP STGSTST ; GO TO RD/WR STG SUBROUTINE
E1B9 7403 525 JE C25 ; GO TO NEXT TEST IF OK
E1B9 E20FF 526 JMP ERR31
E1B9 527
E1B9 528 ; SETUP STACK SEG AND SP
E1B9 529
E1B9 530 C25:
E1B9 E3000 531 MOV AX,STACK ; GET STACK VALUE
E1B9 E3000 532 MOV SS,AX ; SET THE STACK UP
E1B9 E3000 533 MOV SP,OFFSET TOS ; STACK IS READY TO GO
E1B9 534
E1B9 535 ; SETUP THE NMI INTERRUPT VECTOR POINTER
LOC OBJ  LINE SOURCE

E1FA 26C7060800C35E R  537  MOV  ES:MMI_PTR,OFFSET MMU_INT
E201 26C7060A0000F9 R  538  MOV  ES:MMI_PTR+2, CODE
E206 692A60  539  JMP  TS76  ;GO TO NEXT TEST
E20B  540  ;ROSCHECKSUM PROC NEAR  ;NEXT ROCS MODULE
E20B B90020  542  MOV  CX,0192  ;NUMBER OF BYTES TO ADD
E20E 326C  543  XOR  AL,AL
E210  544  C26:  ;ADD AL,CS:I8X)
E21E 62E A  546  INC  BX  ;POINT TO NEXT BYTE
E21F 0016  547  LOOP  C26  ;ADD ALL BYTES IN ROCS MODULE
E216 6668  548  OR  AL,AL  ;SUM = 0?
E218  C3  549  RET
E219  550  ;ROSCHECKSUM END
E219  551  ;-----------------------------
E219  552  ;INITIAL RELIABILITY TEST -- PHASE 2
E219  553  ;----------------------------------
E219  554  ASSUME  CS:CODE, DS:ABSO
E219  555  E219 509452495459290  43644344820312
E21E  556  D120  'PARITY CHECK 2'
E227  557  D1L  EQW  $-D1
E227  558  D220  'PARITY CHECK 1'
E227  559  D2L  EQW  $-D2
E227  560  ;---------------------------------------------
E227  561  ;TESTS.06
E227  562  ;O259 INTERRUPT CONTROLLER TEST
E227  563  ;DESCRIPTION
E227  564  ;READ/WRITE THE INTERRUPT MASK REGISTER (IMR) WITH ALL ONES AND ZEROS.
E227  565  ;ENABLE SYSTEM INTERRUPTS. MASK DEVICE INTERRUPTS OFF. CHECK FOR
E227  566  ;NEG INTERUPTS (UNEXPECTED).
E227  567  ;---------------------------------------------
E22E  568  TS76:  ;SET UP ES REG
E22E  569  SUB  AX,AX  ;SET UP ES REG
E237  570  MOV  ES,AX
E237  571  E235 568752  ;------ SET UP THE INTERRUPT S POINTER TO A DUMMY
E237  572  ;---------------------
E239  573  MOV  ES:INT_PTR,OFFSET PRINT SCREEN  ;PRINT SCREEN
E240  574  MOV  ES:INT_PTR+2, CODE  ;
E240  575  E240  576  ;TEST THE IMR REGISTER
E247  577  E247 56971  ;DISABLE INTERRUPTS
E24E  578  CLI  ;DISABLE INTERRUPTS
E254  580  MOV  AL,0  ;SET IMR TO ZERO
E254  581  E244 6E21  582  OUT  INTA01,AL
E254  582  583  IN  AL,INTA01
E254  584  585  E240 6A0C  586  OR  AL,AL
E254  587  588  E250 7528  589  JNZ  D6
E256  590  591  E252 60FF  592  MOV  AL,OFFH  ;DISABLE DEVICE INTERRUPTS
E256  593  594  E254 6E21  595  OUT  INTA01,AL  ;WRITE TO IMR
E256  596  597  E256 6E21  598  IN  AL,INTA01  ;READ IMR
E256  599  590  E256 7528  591  ADD  AL,AL
E258  592  593  E258 7521  594  JNZ  D6
E25C  595  596  E25C 597  ;SET ALL IMR BIT ON?
E25C  598  ;------ CHECK FOR HOT INTERRUPTS
E25C  599  E25C 597  599  MOV  CX,  ;SET DIR FLAG TO GO FORWARD
E260  600  BF2000  595  MOV  D1,OFFSET INT_PTR  ;SET ADDRESS OF INT PROC TABLE
E263  596  D3:  ;VECTOR:
E263  597  E263 6B9A2E  598  MOV  AX,OFFSET D11  ;MOVE ADDR OF INT PROC TO TBL
E266  599  AD  600  STOSW  ;MOVE ADDR OF INT PROC SEG
E267  600  B00F00  599  MOV  AX,CODE  ;MOVE ADDR OF INT PROC SEG
E26A  600  AD  600  STOSW
E268  63C304  601  ADD  BX,4  ;SET BX TO POINT TO NEXT VAL
E26E  602  E2F3  603  LOOP  D3  ;VECTOR
E26F  604  ;INTERRUPTS ARE MASKED OFF. CHECK THAT NO INTERRUPTS OCCUR.
E270  605  E270 32E4  606  XOR  AH,AH  ;CLEAR AH REG
E272  607  FB  608  STI  ;ENABLE EXTERNAL INTERRUPTS
E273  609  60A  60A  SUB  CX,CK  ;WAIT 1 SEC FOR ANY INTRS THAT
E275  60B  60C  60C  D4: LOOP D4  ;MIGHT OCCUR
E277  60D  60E  60E  D5: LOOP D5  ;
E279  610  611  611  OR  AH,AH  ;DID ANY INTERRUPTS OCCUR?
E27D 7400 612 JZ D7 ;NO - GO TO NEXT TEST
E27D EAO1 613 D6: MOV DX,101H ;SPEAK IF ERROR
E220 EAOA 614 CALL ERR_BEEP ;GO TO DEEP SUBROUTINE
E223 FA 615 CLI
E224 F4 616 HLT ;HALT THE SYSTEM

E225 617 -----------------------------
E225 618 D7:
E225 619 MOV AH,0 ;RESET TIMER INTERRUPT FLAG
E225 61A MOV CH,CH ;CLEAR THE CH REG
E225 61B MOV AL,0FEH ;MASK ALL INTERRUPTS EXCEPT Level 0
E225 61C OUT INTA0D,AL ;WRITE THE 8259 IMR
E225 61D MOV CL,16H ;SET PROGRAM LOOP CNT
E225 61E OUT TIMERO,AL ;WRITE TIMER 0 CNT REG
E225 61F OUT TIMERO,AL ;WRITE TIMER 0 CNT REG
E226 620 MOV AH,0 ;RESET INTERRUPT RECEIVED FLAG
E226 621 MOV AL,0FEH ;REENABLE INTERRUPT 0 INTERRUPTS
E226 622 OUT INTA0D,AL
E226 623 MOV AL,0 ;SET TIMER 0 INTERRUPT
E226 624 MOV AH,OFFH ;TIMEOUT OCCURED?
E226 625 JNZ 02 ;YES - CHECK TIMER OP FOR SLOW TIME
E226 626 LOOP 0D ;WAIT FOR INTERRUPT FOR SPECIFIED TIME
E226 627 JMP 0D ;TIMER 0 INTERRUPT OCCURED - ERR
E226 628 MOV AH,0 ;RESET AX CONTENTS
E226 629 MOV CL,18 ;SET PGM LOOP COUNT
E226 62A MOV AL,OFFH ;WRITE TIMER 0 CNT REG
E226 62B MOV AH,1 ;RESET TIMER INTERRUPT FLAG
E226 62C PUSH AX ;SAVE REG AX CONTENTS
E226 62D MOV AH,1 ;SAVE REG AX CONTENTS
E226 62E MOV AL,OFFH ;MASK ALL INTERRUPTS OFF
E226 62F OUT INTA0D,AL
E227 630 MOV AL,EOI ;RESET INTerruptS OFF
E227 631 MOV AH,OFFH ;RESET TIMER 0 INTERRUPT
E227 632 MOV AL,OFFH ;WRITE TIMER 0 CNT REG
E227 633 MOV AL,1 ;SET TIMER 0 CNT REG
E227 634 MOV AX,EO1 ;SET TIMER 0 CNT REG
E227 635 MOV AX,EO1 ;SET TIMER 0 CNT REG
E227 636 MOV AH,0 ;RESET AX CONTENTS
E227 637 MOV CL,18 ;SET PGM LOOP COUNT
E227 638 MOV AH,OFFH ;TIMEOUT OCCURRED?
E227 639 JNZ 02 ;YES - CHECK TIMER OP FOR SLOW TIME
E227 63A MOV AH,1 ;RESET TIMER INTERRUPT FLAG
E227 63B PUSH AX ;SAVE REG AX CONTENTS
E227 63C MOV AH,1 ;SAVE REG AX CONTENTS
E227 63D MOV AL,OFFH ;MASK ALL INTERRUPTS OFF
E227 63E OUT INTA0D,AL
E227 63F MOV AH,EOI ;RESET INTERRUPT 0 INTERRUPTS
E228 640 MOV AH,OFFH ;WRITE TIMER 0 CNT REG
E228 641 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E228 642 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E228 643 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E228 644 MOV AH,0 ;RESET axial CONTENTS
E228 645 MOV AL,OFFH ;MASK ALL INTERRUPTS OFF
E228 646 OUT INTA0D,AL
E228 647 MOV AH,EOI ;RESET INTERRUPT 0 INTERRUPTS
E228 648 MOV AH,OFFH ;WRITE TIMER 0 CNT REG
E228 649 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E228 64A MOV AH,1 ;RESET TIMER 0 INTERRUPT
E228 64B MOV AH,1 ;RESET TIMER 0 INTERRUPT
E228 64C MOV AH,1 ;RESET TIMER 0 INTERRUPT
E228 64D MOV AH,1 ;RESET TIMER 0 INTERRUPT
E228 64E MOV AH,1 ;RESET TIMER 0 INTERRUPT
E228 64F MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 650 MOV AX,EO1 ;SET TIMER 0 CNT REG
E229 651 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 652 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 653 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 654 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 655 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 656 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 657 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 658 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 659 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 65A MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 65B MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 65C MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 65D MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 65E MOV AH,1 ;RESET TIMER 0 INTERRUPT
E229 65F MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 660 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 661 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 662 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 663 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 664 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 665 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 666 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 667 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 668 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 669 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 66A MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 66B MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 66C MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 66D MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 66E MOV AH,1 ;RESET TIMER 0 INTERRUPT
E230 66F MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 670 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 671 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 672 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 673 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 674 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 675 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 676 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 677 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 678 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 679 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 67A MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 67B MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 67C MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 67D MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 67E MOV AH,1 ;RESET TIMER 0 INTERRUPT
E231 67F MOV AH,1 ;RESET TIMER 0 INTERRUPT
E232 680 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E232 681 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E232 682 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E232 683 MOV AH,1 ;RESET TIMER 0 INTERRUPT
E232 684 MOV AH,1 ;RESET TIMER 0 INTERRUPT

A-10
INITIAL RELIABILITY TEST -- PHASE 3

ASSUME CS:CODE,DS:DATA

; E26: INITIAL RELIABILITY TEST INIT

; Establish BIOS subroutine call interrupt vectors

; Setup timer 0 to mode 3

; Setup timer 0 to blink led if manufacturing test mode

; Assume DS: DATA

; General setup

; Setup timer 0 to blink led if manufacturing test mode

; Test OS checks

; A checksum is done for the 4 ROS modules containing basic code

; Verify checksum

; This completes the initial reliability test
; INITIALIZE AND START CRT CONTROLLER (6845)
; TEST VIDEO READ/WRITE STORAGE.
; DESCRIPTION
; RESET THE VIDEO ENABLE SIGNAL.
; SELECT ALPHANUMERIC MODE. 40 X 25. B & W.
; READ/WRITE DATA PATTERNS TO STG. CHECK STG ADDRESSABILITY.

755 ;-----------------------------
756 ;TEST.08
757 ; INITIALIZE AND START CRT CONTROLLER (6845)
758 ; TEST VIDEO READ/WRITE STORAGE.
759 ; DESCRIPTION
760 ; RESET THE VIDEO ENABLE SIGNAL.
761 ; SELECT ALPHANUMERIC MODE. 40 X 25. B & W.
762 ; READ/WRITE DATA PATTERNS TO STG. CHECK STG ADDRESSABILITY.

;INITIALIZE AHO START CRT cmnROllEp.
(68451

;DESCRIPTION

;RESET THE VIDEO ENABLE SIGNAL.

;SELECT ALPHANUMERIC MODE. 40 X 25. B & W.

;READ/WRITE DATA PATTERNS TO STG. CHECK STG ADDRESSABILITY.

;-----------------------------

;TEST.08

;INITIALIZE AND START CRT cmnROllEp.

;DESCRIPTION

;RESET THE VIDEO ENABLE SIGNAL.

;SELECT ALPHANUMERIC MODE. 40 X 25. B & W.

;READ/WRITE DATA PATTERNS TO STG. CHECK STG ADDRESSABILITY.

;-----------------------------

;TEST.08

;INITIALIZE AND START CRT cmnROllEp.

;DESCRIPTION

;RESET THE VIDEO ENABLE SIGNAL.

;SELECT ALPHANUMERIC MODE. 40 X 25. B & W.

;READ/WRITE DATA PATTERNS TO STG. CHECK STG ADDRESSABILITY.

;-----------------------------

;TEST.08

;INITIALIZE AND START CRT cmnROllEp.

;DESCRIPTION

;RESET THE VIDEO ENABLE SIGNAL.

;SELECT ALPHANUMERIC MODE. 40 X 25. B & W.

;READ/WRITE DATA PATTERNS TO STG. CHECK STG ADDRESSABILITY.

;-----------------------------

;TEST.08

;INITIALIZE AND START CRT cmnROllEp.

;DESCRIPTION

;RESET THE VIDEO ENABLE SIGNAL.

;SELECT ALPHANUMERIC MODE. 40 X 25. B & W.

;READ/WRITE DATA PATTERNS TO STG. CHECK STG ADDRESSABILITY.

;-----------------------------

;TEST.08

;INITIALIZE AND START CRT cmnROllEp.

;DESCRIPTION

;RESET THE VIDEO ENABLE SIGNAL.

;SELECT ALPHANUMERIC MODE. 40 X 25. B & W.

;READ/WRITE DATA PATTERNS TO STG. CHECK STG ADDRESSABILITY.
LOC OBJ

084 ;---------------------------------------------
085 iTEST.10
086 ; CRT INTERFACE LINES TEST
087 ;DESCRIPTION
088 ; SENSE ON/OFF TRANSITION OF THE VIDEO ENABLE AND HORIZONTAL
089 ; SYNC LINES.
090 ;---------------------------------------------

ESC0 50 031 POP AX ; GET VIDEO SENSE SW INFO
ESC1 50 032 PUSH AX ; SAVE IT
ESC2 B0FC30 053 CMP AH,30H ; 1B/W CARD ATTACHED?
ESC5 BABA03 054 MOV DX,03BAH ; SETUP ADDR OF EM STATUS PORT
ESC8 7A43 055 JE E11 ; YES - GO TEST LINES
ESC4 BA03 056 MOV DX,03DAH ; COLOR CARD IS ATTACHED
ESC0 057 E11: MOV AH,0 ; LINE_CNT:
ESC0 B400 058 MOV AH,0 ; OFLOOP_CNT:
ESC5 2BC9 060 SUB CX,CX
ESC0 EC 061 IN AL,DX ; READ CRT STATUS PORT
ESC0 2C4 062 AND AL,AX ; CHECK VIDEO/HORIZ LINE
ESC0 7504 063 JNZ E14 ; ITS ON - CHECK IF IT GOES OFF
ESC0 EF9 064 LOOP E13 ; LOOP TILL ON OR TIMEOUT
ESC8 ED13 065 JMP SHORT E17 ; PRINT ERROR MSG
ESC8 2BC9 066 E14: SUB CX,CX
ESC0 EC 067 IN AL,DX ; READ CRT STATUS PORT
ESC0 2C4 068 AND AL,AX ; CHECK VIDEO/HORIZ LINE
ESC0 7404 069 JZ E16 ; ITS ON - CHECK NEXT LINE
ESC5 EF9 070 LOOP E15 ; LOOP IF OFF TILL IT GOES ON
ESC5 B801 071 JMP SHORT E17 ;
ESC5 852 E16: MOV DX,10H
ESC5 B103 053 MOV CL,3 ; GET NEXT BIT TO CHECK
ESC5 D2EC 054 SHR AL,CL
ESC5 75E4 055 JNZ E12 ; GET CHECK HORIZONTAL LINE
ESC5 E06 056 JMP SHORT E18 ; DISPLAY CURSOR ON SCREEN
ESC5 857 E17: MOV DX,102H
ESC8 D401 058 MOV DX,102H
ESC8 E302 059 CALL ERR_BEEP ; IO BEEP SPEAKER
ESC8 860 E18: ; DISPLAY_CURSOR:
ESC8 85A E61 MOV AL,.4 ; GET VIDEO SENSE SW 1 (AH)
ESC8 B400 062 MOV AH,0 ; SET MODE AND DISPLAY CURSOR
ESC8 C010 063 INT 10H ; CALL VIDEO I/O PROCEDURE

064 ;---------------------------------------------
065 iTEST.11
066 ; ADDITIONAL READ/WRITE STORAGE TEST
067 ;DESCRIPTION
068 ; WRITE/READ DATA PATTERNS TO ANY READ/WRITE STORAGE AFTER THE BASIC
069 ; RAM SIZE.
070 ; STORAGE ADDRESSABILITY IS CHECKED.
071 ;---------------------------------------------

ESC8 85F 072 E19: ASSUME DS:DATA
ESC8 B400 R 073 MOV AX,DATA
ESC8 B000 074 MOV AX,DATA
ESC8 85F 075 ; DETERMINE RAM SIZE ON PLANAR BOARD
ESC8 85D 076 ; DETERMINE RAM SIZE ON PLANAR BOARD
ESC8 A261000 R 077 MOV AX,BYTE PTR EQUIP_FLAG ; GET SENSE SW INFO
ESC8 80C40 078 AND AL,0CH ; ISOLATE RAM SIZE SW
ESC8 8094 079 MOV AL,4
ESC8 F6E4 080 MUL AX
ESC8 0410 081 ADD AL,16 ; ADD BASIC 16K
ESC8 8800 082 MOV DX,AX ; SAVE PLANAR RAM SIZE IN DX
ESC8 B508 083 MOV BX,AX
ESC8 85E 084 ; AND IN BX
ESC8 85F 085 ; DETERMINE IO CHANNEL RAM SIZE
ESC8 807 086 ; DETERMINE IO CHANNEL RAM SIZE
ESC8 E462 088 IN AL,PORT_C ; READ IO CH RAM SIZE SW
ESC8 240F 089 AND AL,0FH ; ISOLATE FROM OTHER BITS
ESC8 B420 090 MOV AH,32
ESC8 F6E4 091 MUL AX
ESC8 A31500 R 092 MOV IO_RAM_SIZE,AX ; SAVE IO CHANNEL RAM SIZE
ESC8 83FB40 093 CMP BX,40H ; PLANAR RAM SIZE = 64K?
ESC8 7402 094 JE E20 ; YES - ADD IO CHN RAM SIZE
ESC8 2B0C 095 SUB AX,AX ; NO - DO T ADD ANY IO RAM
ESC8 7402 096 E20: ADD AX,BX ; ADD IO SIZE:
ESC8 05C3 097 ADD AX,BX ; SUM TOTAL RAM SIZE
ESC8 A1300 R 098 MOV MEMORY_SIZE,AX ; SETUP MEMORY SIZE PARM
ESC8 013E72003412 R 099 CMP RESET_FLAG,1234H ; POD INITIATED BY KBD RESET?
LOC OBJ  LINE  SOURCE

E428 744D  900  JE  E22  ;YES - SKIP MEMORY TEST

E42D 88000D  904  MOV  BX,480H
E430 89100D  905  MOV  CX,16

E433 906  E21:  ;ANY MORE STG TO BE TESTED?
E433 3D01  907  CMP  DX,CX  ;SET POINTER TO NEXT 16K BLK
E437 7646  908  JBE  E23  ;NO - GO TO NEXT TEST
E439 06C3  909  MOV  DS,BX  ;SET UP STG ADDR IN DS AND ES
E43B 03C110  910  MOV  ES,BX
E43E 01C30004  912  ADD  BX,480H  ;SET POINTER TO NEXT 16K BLK
E442 51  913  PUSH  CX  ;SAVE REGS
E443 53  914  PUSH  BX
E444 9E  915  POP  BX  ;RESTORE REGS
E445 E8D2FB  917  POP  OX
E448 8ED8  918  POP  OX
E449 88C3  919  ADD  DS,16  ;INCREMENT STACK BYTE COUNTER
E44C 74E6  920  JE  E21  ;CHECK IF MORE STG TO TEST

E451 83C3  921  MOV  DX,DS  ;CONVERT FAILING HIGH-ORDER BYTE
E454 81C30004  923  ADD  BX,400H  ;SET POINTER TO NEXT 16K BLK
E457 EB04  925  MOV  AL,0H  ;CONVERT AND PRINT CODE
E45A EC4A  926  MOV  AL,0H  ;GET FAILING BIT PATTERN
E45C 240F  928  MOV  AL,0FH  ;ISOLATE LEFTMOST NIBBLE
E45E 0602  929  MOV  AL,CH  ;GET FAILING BIT PATTERN
E461 EB04  930  MOV  CL,4  ;AND ISOLATE RIGHTMOST NIBBLE
E463 E82700  931  MOV  AL,OH  ;GET FAILING BIT PATTERN
E465 943  932  ADD  CL,1  ;AND ISOLATE LEFTMOST NIBBLE
E467 EB0E00  933  MOV  AL,CH  ;GET FAILING BIT PATTERN
E46A EC4C  934  MOV  AL,CH  ;GET FAILING BIT PATTERN
E46C 240F  935  MOV  AL,OFH  ;SET FLAG RESULT
E46E EB7700  936  MOV  AL,CL  ;GET FAILING BIT PATTERN
E471 D0E2E2  937  MOV  SI,OFFSET ASCII_TABLE  ;GET FAILING BYTE COUNT
E474 E90400  938  MOV  DX,ES  ;SET UP STG ADDR IN DS AND ES
E477 E85002  939  CALL  P_MSG  ;PRINT ERROR MSG
E478 8E  940  JMP  SHORT E21  ;GO TO NEXT TEST
E47A E94000  941  JMP  TST12  ;GO TO NEXT TEST
E47D 8E  942  JMP  TST12  ;GO TO NEXT TEST
E47D B04000  943  MOV  AX,DATA  ;POINT DS TO DATA SEGMENT
E480 06DB  944  MOV  DS,AX  ;CHG MADE 3/27/81
E482 B016500  945  MOV  DX,IO_RAM_SIZE  ;SET IO CHANNEL RAM SIZE
E484 0802  946  OR  DX,DX  ;SET FLAG RESULT
E486 74F0  947  JZ  E22  ;IND IO RAM, GO TO NEXT TEST
E48A B90000  948  MOV  AX,0  ;HAS IO RAM BEEN TESTED
E490 61F00010  949  JMP  SHORT E21  ;SETUP ERR LOCA FOR IO RAM
E493 E80010  94A  MOV  BX,IO_RAM  ;SETUP ERR LOCA FOR IO RAM
E496 EB90  94B  JMP  SHORT E21  ;SETUP ERR LOCA FOR IO RAM

E496 955 61  ;-----------------------------------------------
E497 956  ;-----------------------------------------------
E497 957  ;-----------------------------------------------
E498 958  ;-----------------------------------------------
E498 959  ;-----------------------------------------------
E498 960  ;-----------------------------------------------
E499 961  ;-----------------------------------------------
E499 962  ;-----------------------------------------------
E499 963  ;-----------------------------------------------
E499 964  ;-----------------------------------------------
E499 965  ;-----------------------------------------------
E499 966  ;-----------------------------------------------
E499 967  ;-----------------------------------------------
E499 968  ;-----------------------------------------------
E499 969  ;-----------------------------------------------
E499 970  ;-----------------------------------------------
E499 971  ;-----------------------------------------------
E499 972  ;-----------------------------------------------
E499 973  ;-----------------------------------------------

A-14
LOC OBJ

LINE

974

SOURCE

;

--------------------------_.---------------------._--------------------------

975

; INITIAL RELIABILITY TEST --. PHASE 4

97.

;

ASSUME

977

E4A7 Z0333031

0004
E4AB 313331
00-03
E4AE 363031
0003

976

Fl

979

CS :CODE .DS:OATA

DB

• 301'

FlL

EOU

$-Fl

; KEYBOARD MESSAGE

98'

FZ

DB

96'

F2L

EOU

j

962

F3

DB

'131 '
$-FZ
'601 •

F3L

EOU

$-F3

; DISKETTE MESSAGE

F4

LABEL

WORD

; PRINTER SOURCE TABLE

963

CASSETTE MESSAGE

96'

E4Bl

965

E481 BC03

96.

OW

36CH

£483 7803

987

OW

378H

E485 780Z

968

E487

989

'4E

99'

ASCICT6L

991

;

992

ITEST.12

E481 30313Z33343536

OW

2:78H

LABEL

WORD
'0123456 789ABCDEF'

DB

373133941424344
4546

----------------------------------- ------._-

993
994

KEYBOARD TEST

;OESCRIPTION

995

PESET THE KEYBOARD AND CHECK THAT SCAN CODE

996

TO THE CPU.

AA' IS RETURNED

CHECK FOR STUCK KEYS.

---_.---_.-----._- --------------------------

997

;

E4C7

998

TSTl2:

E4C7 B84000

999

MOV

AX,DATA

E4CA 8ED8

1000

MOV

DS.AX

E4CC 803El£:0001

1001

CM"

tlFG_TST .1

;MAUUfACTUIUHG TEST MODE?

£401 7439
£403 £88201

1002
1003

JE

'7

CAll

KBD_RESET

£406 £32B

1004

JCXZ

,.

; YES - SKIP KEYBOARD TEST
J ISSUE SOFTWARE RESET TO KEYBRD

E4D8 B040

1005

I10V

Al.4DH

;POINT os TO DATA SEG

; PRINT ERR HSG IF NO INTERRUPT
;ENABLE KEYBOARD

E40A £661

1006

OUT

PORT_B.Al

E40C BOFBAA

1007

eMP

BL,OAAH

;SCAN CODE AS EXPECTED?

E4DF 7522

1008

JNE

F.

;NO - DISPLAY ERROR t1SG

1009
1010

CHECK FOR STUCK KEYS

1011
E4E1 Boce

101Z

MDV

AL,OCcH

E4E3 E661

1013

OUT

PORT_B.AL

E4E5 B04C

1014

MOV

AL,4tH

E4E7 E661

lOIS

OUT

PORT_B,AL

E4E9 2BC9

1016

SUB

cX,ex

E4EB

1017

E4EB E2FE

1018

lOOP

F5

;DElAY FOR A WHILE

FS:

ICU! KBD. SET CLK LINE HIGH
;ENABLE KBO,CLK IN NEXT BYTE

; KBD_WAIT:

E4ED E460

1019

IN

AL,KBD_IN

;CHECK fOR STUCK KEYS

E4EF 3(00

1020

CMP

Al,O

;SCAN CODE ::;. O?

E4F1 7419

1021

JE

F7

; YES - tONTINUE TESTING

E4F3 SAE8

1022

MOV

CH,Al

E4F5 BI04

1023

MOV

CL,4

E4F7 02E8

1024

SHR

AL,Cl

;RIGHT-JUSTIFY HIGH BYTE

E4F9 E89CFf

;SAVE SCAN CODE

1025

CAll

XLAT_P!;IINT_CODE

;eOHVERT AHD PRINT

E4Fe SAC5

1026

HOV

Al,eH

;RECOVER SCAN ceDE

E4F£ 240F

1027

AND

Al,OFH

;ISOLATE lOW ORDER BYTE

E500 E89SFF

1028

CALL

XLAT_PIUNT_CODE

;CONVERT AND PRINT

E503 BEA7E4

R

1029

F6:

MOV

5I.OFFSET F1

i GET MSG ADDR

ES06 B90400

1030

MOV

CX, fIt

; GET t1SG BYTE COUNT

E509 E8BEOI

1031

CALL

P_I1SG

j

PRINT MSG ON SCREEN

1032
1033

SETUP INTERRUPT VECTOR TABLE

1034

F7:

E50C

1035

ESOC 28CO

1036

SUB

AX,AX

ESOE SEeD

1037

HDV

ES,AX

E510 893000

1038

MOV

CX,24*2

;GET VECTOR CNT

E513 OE

1039

PUSH

CS

;SETUP OS SEG REG

ES14 If

1040

POP

OS

; SETUP_INT_TABlE:

ES15 BEF3FE

1041

HOV

SI.OFEF3H

£518 BF20DD

1042

MOV

DI.OfFSET INT_PTR

ESIB Fe

1043
1044

CLD

ESIC F3

REP

; OffSET VECTOR_TABLE

110VSW

ESID AS

A-IS


<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1045</td>
<td>TEST.13</td>
<td></td>
</tr>
<tr>
<td>1047</td>
<td>CASETTE DATA WRAP TEST</td>
<td></td>
</tr>
<tr>
<td>1048</td>
<td>DESCRIPTION</td>
<td></td>
</tr>
<tr>
<td>1049</td>
<td>TURN CASETTE MOTOR OFF, WRITE A BIT OUT TO THE CASETTE DATA BUS.</td>
<td></td>
</tr>
<tr>
<td>1050</td>
<td>VERIFY THAT CASETTE DATA READ IS WITHIN A VALID RANGE.</td>
<td></td>
</tr>
<tr>
<td>1053</td>
<td>TURN THE CASETTE MOTOR OFF.</td>
<td></td>
</tr>
<tr>
<td>1056</td>
<td>MVW AL,DATA</td>
<td>SET DS REG TO DATA SEG</td>
</tr>
<tr>
<td>1057</td>
<td>MVW DS,AX</td>
<td></td>
</tr>
<tr>
<td>1058</td>
<td>MVW AL,040H</td>
<td>SET TIMER 2 SPK OUT, AND CASST</td>
</tr>
<tr>
<td>1059</td>
<td>OUT PORT_D,AL</td>
<td>OUT BITS ON, CASETTE M6 OFF</td>
</tr>
<tr>
<td>1061</td>
<td>WRITE A BIT</td>
<td></td>
</tr>
<tr>
<td>1062</td>
<td>MVW AL,OFFH</td>
<td>DISABLE TIMER INTERRUPT</td>
</tr>
<tr>
<td>1063</td>
<td>MVW DS,AL</td>
<td></td>
</tr>
<tr>
<td>1064</td>
<td>MVW AL,OFFH</td>
<td>SET TIM 2, LSB, MSB, MD 3</td>
</tr>
<tr>
<td>1065</td>
<td>OUT TIMER 2,AL</td>
<td>WRITE 855 CMD/MODE REG</td>
</tr>
<tr>
<td>1066</td>
<td>MVW AX,1235</td>
<td>SET TIMER 2 CNT FOR 1000 USEC</td>
</tr>
<tr>
<td>1067</td>
<td>OUT TIMER+2,AL</td>
<td>WRITE TIMER 2 COUNTER REG</td>
</tr>
<tr>
<td>1068</td>
<td>MVW AL,08H</td>
<td>WRITE MSB</td>
</tr>
<tr>
<td>1069</td>
<td>OUT TIMER+2,AL</td>
<td></td>
</tr>
<tr>
<td>1070</td>
<td>READ CASETTE INPUT</td>
<td></td>
</tr>
<tr>
<td>1071</td>
<td>IN AL,PORT_C</td>
<td>READ VALUE OF CASS IN BIT</td>
</tr>
<tr>
<td>1074</td>
<td>AND AL,10H</td>
<td>ISOLATE FROM OTHER BITS</td>
</tr>
<tr>
<td>1075</td>
<td>MVW LAST,10L</td>
<td></td>
</tr>
<tr>
<td>1076</td>
<td>CALL READ_HALF_BIT</td>
<td></td>
</tr>
<tr>
<td>1077</td>
<td>CALL READ_HALF_BIT</td>
<td></td>
</tr>
<tr>
<td>1078</td>
<td>JCXZ F8</td>
<td>CAS_ERR</td>
</tr>
<tr>
<td>1079</td>
<td>CMP BX,MAX_PERIOD</td>
<td></td>
</tr>
<tr>
<td>1080</td>
<td>JNC F8</td>
<td>CAS_ERR</td>
</tr>
<tr>
<td>1081</td>
<td>CMP BX,MIN_PERIOD</td>
<td></td>
</tr>
<tr>
<td>1082</td>
<td>JNC F9</td>
<td></td>
</tr>
<tr>
<td>1083</td>
<td>F8:</td>
<td>CAS_ERR:</td>
</tr>
<tr>
<td>1084</td>
<td>MVW SI,OFFSET F2</td>
<td>CASETTE WRAP FAILED</td>
</tr>
<tr>
<td>1085</td>
<td>MOV AX,F2L</td>
<td></td>
</tr>
<tr>
<td>1086</td>
<td>CALL PRO_MSG</td>
<td></td>
</tr>
<tr>
<td>1087</td>
<td>;TEST.14</td>
<td></td>
</tr>
<tr>
<td>1089</td>
<td>DISKETTE ATTACHMENT TEST</td>
<td></td>
</tr>
<tr>
<td>1090</td>
<td>DESCRIPTION</td>
<td></td>
</tr>
<tr>
<td>1091</td>
<td>CHECK IF IPIF DRIVE IS ATTACHED TO SYSTEM. IF ATTACHED,</td>
<td></td>
</tr>
<tr>
<td>1092</td>
<td>VERIFY STATUS OF HEC FDC AFTER A RESSET. ISSUE A MEAL AND SEEK</td>
<td></td>
</tr>
<tr>
<td>1093</td>
<td>CHD FDC AND CHECK STATUS. COMPLETE SYSTEM INITIALIZATION THEN</td>
<td></td>
</tr>
<tr>
<td>1094</td>
<td>PASS CONTROL TO THE BOOT LOADER PROGRAM.</td>
<td></td>
</tr>
<tr>
<td>1095</td>
<td>;</td>
<td></td>
</tr>
<tr>
<td>1096</td>
<td>F9:</td>
<td></td>
</tr>
<tr>
<td>1097</td>
<td>MVW AL,OFFC</td>
<td>ENABLE TIMER AND KBD INTS</td>
</tr>
<tr>
<td>1098</td>
<td>OUT INTA01,AL</td>
<td></td>
</tr>
<tr>
<td>1099</td>
<td>MVW AL,BYTE PTR EQUIP_FLAG</td>
<td>GET SENSE SWS INFO</td>
</tr>
<tr>
<td>1100</td>
<td>TEST AL,01H</td>
<td>IPIF DISKETTE DRIVE ATCHT</td>
</tr>
<tr>
<td>1101</td>
<td>JNZ F10</td>
<td>YES - TEST DISKETTE CONTR</td>
</tr>
<tr>
<td>1102</td>
<td>JMP F22</td>
<td>AND - SKIP THIS TEST</td>
</tr>
<tr>
<td>1103</td>
<td>F10:</td>
<td>DISK_TEST:</td>
</tr>
<tr>
<td>1104</td>
<td>MVW AL,OFFC</td>
<td>ENABLE DISKETTE, KEYBOARD,</td>
</tr>
<tr>
<td>1106</td>
<td>OUT INTA01,AL</td>
<td>AND TIMER INTERRUPTS</td>
</tr>
<tr>
<td>1107</td>
<td>MVW AH,0</td>
<td></td>
</tr>
<tr>
<td>1108</td>
<td>INT 13H</td>
<td>RESET HEC FDC</td>
</tr>
<tr>
<td>1109</td>
<td>TEST AH,OFFH</td>
<td>STATUS OK?</td>
</tr>
<tr>
<td>1110</td>
<td>JNZ F13</td>
<td>AND - FDC FAILED</td>
</tr>
<tr>
<td>1111</td>
<td>;</td>
<td></td>
</tr>
<tr>
<td>1112</td>
<td>TURN DRIVE 0 MOTOR ON</td>
<td></td>
</tr>
<tr>
<td>1113</td>
<td>MOV DX,05F2H</td>
<td>SET ADDR OF FDC CARD</td>
</tr>
<tr>
<td>1114</td>
<td>MOV AL,ICH</td>
<td></td>
</tr>
<tr>
<td>1115</td>
<td>OUT DX,AL</td>
<td>READ FDC CONTROL REG</td>
</tr>
<tr>
<td>1116</td>
<td>SUB CX,CX</td>
<td></td>
</tr>
<tr>
<td>1117</td>
<td>F11:</td>
<td>MOTOR_WAIT:</td>
</tr>
<tr>
<td>1118</td>
<td>LOOP F11</td>
<td></td>
</tr>
<tr>
<td>1119</td>
<td>F12:</td>
<td>WAIT FOR 1 SECOND</td>
</tr>
<tr>
<td>1120</td>
<td>LOOP F12</td>
<td>MOTOR_WAIT:</td>
</tr>
<tr>
<td>1121</td>
<td>XOR DX,DX</td>
<td>SELECT DRIVE 0</td>
</tr>
</tbody>
</table>
E655 B501 1122 MOV CH,1 ;SELECT TRACK 1
E657 B016B00 R 1123 MOV SEEK_STATUS,DL
E5AB 86F308 1124 CALL SEEK ;RECALIBRATE DISKETTE
E56E 7207 1125 JC F13 ;GO TO ERR SUBROUTINE IF ERR
E590 B522 1126 MOV CH,34 ;SELECT TRACK 34
E592 86EC08 1127 CALL SEEK ;SEEK TO TRACK 34
E596 7509 1128 JNC F14 ;OK, TURN MOTOR OFF
E597 1129 F13: ;DISK_ERR:
E597 B0AE4 R 1130 MOV SI,OFFSET F3 ;GET ADDR OF MSG
E59A B90000 1131 MOV CX,FL ;GET MSG BYTE COUNT
E59D B0201 1132 CALL P_MSG ;GET PRINT ERROR MSG
1133
1134 TURN DRIVE 0 MOTOR OFF
1135
1136 F14: ;DR0_OFF:
1137 MOV AL,GCH ;TURN DRIVE 0 MOTOR OFF
1138 MOV DX,3F2H ;FOC CTL ADDRESS
1139 OUT DX,AL
1140
1141 SETUP PRINTER AND R5232 BASE ADDRESSES IF DEVICE ATTACHED
1142
1143 F15: ;JMPP_BOOT:
1144 MOV BUFFER_HEADER,OFFSET KB_BUFFER ;SETUP KEYBOARD PARAMETERS
1145 MOV BUFFER_TAIL,OFFSET KB_BUFFER
1146 MOV BP,OFFSET F4 ;PRR_SRC_TBL
1147 MOV SI,0 ;PRR_BASE:
1148 MOV DX,C5:BP ;GET PRINTER BASE ADDR
1149 MOV AL,AAH ;WRITE DATA TO PORT A
1150 MOV EE EE 1151 OUT DX,AL
1152 MOV EE2A0 1153 SUB AL,AL ;READ PORT A
1154 MOV SC0 1155 CMP AL,AAAH ;DATA PATTERN SAME
1156 MOV 7506 1157 JNC F17 ;NO - CHECK NEXT PRY CD
1158 MOV 09940000 R 1159 MOV PRINTER_BASE$SI.DX ;YES - STORE PRT BASE ADDR
1160 MOV CA 1161 INC SI ;INCREMENT TO NEXT WORD
1162 MOV CB 1163 INC SI
1164 MOV 46 1165 INC BP ;POINT TO NEXT BASE ADDR
1166 MOV 46 1167 INC BP
1168 MOV SC0 1169 INC BP
1170 MOV 01F087E4 R 1171 MOV CP,OFFSET FA4 ;ALL POSSIBLE ADDRS CHECKED?
1172 MOV 75E4 1173 JNC F16 ;PRR_BASE
1174 MOV 80000 1175 MOV BX,0 ;PRR_POINTER TO R5232 TABLE
1176 MOV BAF03 1177 MOV DX,3F2H ;CHECK IF R5232 CD 1 ATTCH?
1178 EDAA EC 1179 IN AL,DX ;READ INTR 2D REG
1180 EDAD AF06 1181 TEST AL,0FH
1182 ED0D 7508 1183 JNZ F18
1184 ED5F C7070000F003 R 1185 MOV R5232_BASE,BX,1,3FH ;SETUP R5232 CD #1 ADDR
1186 ED5E 45 1187 INC BX
1188 ED5E 45 1189 INC BX
1190 ED57 BAF02 1191 MOV BX,2F2H ;CHECK IF R5232 CD 2 ATTCH
1192 ED5E EC 1193 IN AL,DX ;READ INTERRUPT ID REG
1194 ED5F AF06 1195 TEST AL,0FH
1196 ED5D 7508 1197 JNZ F19 ;BASE_END
1198 ED5F C7070000F802 R 1199 MOV R5232_BASE,BX,1,2FH ;SETUP R5232 CD #2
1200 ED5F 45 1201 INC BX
1202 ED5E 45 1203 INC BX
1204 1205 ;----- SET UP EQUIP FLAG TO INDICATE NUMBER OF PRINTER AND R5232 CARDS
1206 E5F7 1207 F19: ;BASE_END:
1208 ED57 B0C6 1209 MOV AX,SI ;SI HAS 2* NUMBER OF R5232
1210 ED59 B105 1211 MOV CL,3 ;SHFT COUNT
1212 ED58 D2C8 1213 MOV R0,AL,CL ;ROTATE RIGHT 3 POSITIONS
1214 ED50 0A3C 1215 OR AL,6L ;OR IN THE PRINTER COUNT
1216 ED5F A2100 R 1217 MOV BYTE PTR EQUIP_FLAG+1,AL ;STORE AS SECOND BYTE
1218 ED62 BA0102 1219 MOV DX,10H
1220 ED65 EC 1221 IN AL,DX
1222 ED66 A0F 1223 TEST AL,0FH
1224 ED68 7505 1225 JNZ F20 ;NO GAME CARD
1226 ED6A 0000110000 R 1227 OR BYTE PTR EQUIP_FLAG+1,16
1228 ED6F 1229 F20: ;NO_GAME_CARD:
1230 1231 ENABLE NMI INTERRUPTS
1232 1233 ED6F B000 1234 MOV AL,60H
1235 ED61 ED40 1236 OUT 0A0H,AL

APPENDIX A
LOC OBJ LINE SOURCE

E63 003I120001 R 1199 CMP MFG_TST.1 :MFG MODE?
E618 7406 1200 JE F21 ; LOAD_BOOT_STRAP
E61A DA0100 1201 MOV DX,1 ;
E61D ED0100 1202 CALL ERR_BEEP ;BEEP 1 SHORT TONE
E620 E9F00 1203 F21: JMP BOOT_STRAP ;GO TO THE BOOT LOADER
E623 1204 F22: ; LOOP_POD:
E623 003I120001 R 1205 CMP MFG_TST.1 ;MANUFACTURING TEST MODE?
E628 7503 1206 JNE F23 ;IND - GO TO BOOT LOADER
E62A E9EFA 1207 JMP START ;YES - LOOP POWER-ON-DIAGS
E62D E9F00 1208 F23: JMP F15 ;GO TO BOOT:
E62D E97FF 1209 F24: ; LOOP_POD:

E630 1210 ---- SUBROUTINES FOR POWER ON DIAGNOSTICS
E633 1211 ; INITIAL RELIABILITY TEST -- SUBROUTINES
E636 1212 ; SUBROUTINES FOR POWER ON DIAGNOSTICS
E639 1213 ; THIS PROCEDURE WILL ISSUE ONE LONG TONE (3 SECS) AND ONE OR
E63C 1214 ; MORE SHORT TONES (1 SEC) TO INDICATE A FAILURE ON THE PLANAR
E63F 1215 ; BOARD, A BAD RAM MODULE, OR A PROBLEM WITH THE CRT.
E642 1216 ; ENTRY PARAMETERS:
E645 1217 ; DH = NUMBER OF LONG TONES TO BEEP
E648 1218 ; DL = NUMBER OF SHORT TONES TO BEEP.
E64B E9F00 1219 ---- SUBROUTINES FOR POWER ON DIAGNOSTICS

E650 WC 1220 PUSH ;SAVE FLAGS
E653 FA 1221 CLI ;DISABLE SYSTEM INTERRUPTS
E656 1222 PUSH DS ;SAVE DS REG CONTENTS
E659 B0000 R 1223 MOV AX,DATA ;POINT DS TO DATA SEG
E65C B0DB 1224 MOV DS,AX
E65F 8A60 1225 OR DH,OH ; ANY LONG ONES TO BEEP
E662 7416 1226 JZ G3 ; NO, DO THE SHORT ONES
E665 1227 G1: ; LONG_BEEP:
E668 B306 1228 MOV BL,6 ; COUNTER FOR BEEPS
E66B E9500 1229 CALL BEEP ; DO THE BEEP
E66E E9FF 1230 LOOP G2 ; DELAY BETWEEN BEEPS
E671 FECE 1231 DEC DH ; ANY MORE TO DO
E674 76F5 1232 JIZ G1 ; DO IT
E677 E9EFA 1233 CMP MFG_TST.1 ; MFG TEST MODE?
E67A 7503 1234 JIE G3 ; YES - CONTINUE BEEPING SPEAKER
E67D E9D0 1235 MOV AL,0CDH ; STOP BLINKING LED
E67F E9F0 1236 OUT PORT_B,AL
E682 E9E8 1237 JMP SHORT G1
E685 1238 G5: ; SHORT_BEEP:
E688 B301 1239 MOV BL,1 ; COUNTER FOR A SHORT BEEP
E68B E99000 1240 CALL BEEP ; DO THE SOUND
E68E E9FF 1241 LOOP G4 ; DELAY BETWEEN BEEPS
E691 F9CA 1242 DEC DL ; SOME WITH SHORTS
E694 75F5 1243 JIZ G3 ; DO SOME MORE
E697 E9FF 1244 LOOP G5 ; LONG DELAY BEFORE RETURN
E69A E9F0 1245 OUT PORT_B,AL
E69F E9F0 1246 POP DS ;RESTORE ORIG CONTENTS OF DS
E6A4 1247 POPF ;RESTORE FLAGS TO ORIG SETTINGS
E6A5 C3 1248 RET ; RETURN TO CALLED

E6A6 1249 ERR_BEEP ENDP
E6A8 1250 ---- ROUTINE TO SOUND BEEPER
E6B0 1251 BEEP PROC NEAR
E6B3 B006 1252 MOV AL,1011010B ;SEL TIM 2,LSB,MSB,BINARY
E6B6 E943 1253 OUT TIMER+3,AL ;WRITE THE TIMER MODE REG
E6B9 E0305 1254 MOV AX,533H ;DIVISOR FOR 1000 HZ
E6BC E942 1255 OUT TIMER+2,AL ;WRITE TIMER 2 CNT - LSB
E6C8 E9A4 1256 MOV AL,AH
E6CD E942 1257 OUT TIMER+2,AL ;WRITE TIMER 2 CNT - MSB
E6D8 E961 1258 IN AL,PORT_B ;GET CURRENT SETTING OF PORT
E6DC E9AE 1259 MOV AH,AL ; SAVE THAT SETTINGS
E6E7 0C03 1260 OR AL,03 ;TURN SPEAKER ON
E6E9 E961 1261 OUT PORT_B,AL
E6EC E9C0 1262 SUB CX,CX ;SET CNT TO WAIT 500 MS
E6ED E9FF 1263 LOOP G7 ;DELAY BEFORE TURNING OFF
E6F1 FECD 1264 DEC BL ;DELAY CNT EXPIRED?
E6F5 75A 1265 JIZ G7 ;IND - CONTINUE BEEPING SPK
E6F8 8AC4 1266 MOV AL,AH ; RECOVER VALUE OF PORT

A-18
ENTRY REQUIREMENTS:

1331 ; SI = OFFSET(ADDRESS) OF MESSAGE BUFFER
1332 ; CX = MESSAGE BYTE COUNT
1333 ; MAXIMUM MESSAGE LENGTH IS 36 CHARACTERS
1334 ;

E6CA 1335 PROC P_MSG NEAR
E6CA B80000 R 1336 MDV AX.DATA ;POINT DS TO DATA SEG
E6CD 8E08 R 1337 MDV DS:AX
E6CF 003E120001 R 1338 CMP MFG.TST.1 ;MFG TEST MODE
E6D4 7505 1339 JNE G1Z ;NO - DISPLAY ERROR MSG
E6D6 B601 ; MDV OH.1 ;YES - SETUP TO BEEP SPEAKER
E6D8 E95FF 1340 JMP ERP_BEEP ;YES - BEEP SPEAKER
E6DB 1342 G1Z: ;WRITE_MSG:
E6DB 2B0A04 1343 MDV AL.CS:151I ;PUT CHAR IN AL
E6DE 46 1344 INC SI ;POINT TO NEXT CHAR
E6DF B700 1345 MDV OH.0 ;SET PAGE B TO ZERO
E6E1 B40E 1346 MDV AH.14 ;WRITE CHAR (TTY-INTERFACE)
E6E3 C010 1347 INT 10H ;CALL VIDEO.IO
E6E5 E2F4 1348 LOOP G1Z ;CONTINUE TILL MSG WRITTEN
E6E7 B0000E 1349 MDV AX.0ED9 ;POSITION CURSOR TO NEXT LINE
E6EA C010 1350 INT 10H ;SEND CARRIAGE RETURN AND

BEEP ENDP
1328 ; THIS SUBROUTINE WILL PRINT A MESSAGE ON THE DISPLAY
1329 ;
1330 ;ENTRY REQUIREMENTS:
1331 ; SI = OFFSET(ADDRESS) OF MESSAGE BUFFER
1332 ; CX = MESSAGE BYTE COUNT
1333 ; MAXIMUM MESSAGE LENGTH IS 36 CHARACTERS
1334 ;

LOC OBJ LINE SOURCE
E605 E661 1275 OUT PORT_B,AL ;RETURN TO CALLER
E607 C3 1276 RET ;RETURN TO CALLER
E608 E661 1277 BEEP ENDP
1278 ; THIS PROCEDURE WILL SEND A SOFTWARE RESET TO THE KEYBOARD.
1279 ; SCAN CODE AA' SHOULD BE RETURNED TO THE CPU.
1280 ;
1281 ;

E60B 1282 KBD_RESET PROC NEAR
E60B B00C 1283 MDV AL.OCH ;SET KBD CLK LINE LOW
E60C E661 1284 OUT PORT_B,AL ;WRITE 8255 PORT B
E60C B95629 1285 MDV CX.19552 ;HOLD KBD CLK LOW FOR 20 MS
E60F E2FE 1286 G0: LOOP G0 ;LOOP FOR 20 MS
E610 B0CC 1287 MDV AL.OCH ;SET CLK, ENABLE LINES HIGH
E693 E661 1288 OUT PORT_B,AL
E695 1289 SP_TEST: ;ENTRY FOR MANUFACTURING TEST 2
E695 B04C 1290 MDV AL.OCH ;SET KBD CLK HIGH, ENABLE LOW
E697 E661 1291 OUT PORT_B,AL
E699 B0FD 1292 MDV AL.ODFH ;ENABLE KEYBOARD INTERRUPTS
E69B E621 1293 OUT INTA01,AL ;WRITE 8259 IMR
E69F F8 1294 STI ;ENABLE SYSTEM INTERRUPTS
E6A0 B400 1295 MDV AH.0 ;RESET INTERRUPT INDICATOR
E6A0 2BC9 1296 SUB CX,CX ;SETUP INTERRUPT TIMEOUT CNT
E6A2 66C4FF 1297 G9: TEST AH.OFFH ;DID A KEYBOARD INTR OCCUR?
E6A5 7502 1298 JNZ G10 ;YES - READ SCAN CODE RETURNED
E6A7 E2F9 1299 LOOP G9 ;NO - LOOP TILL TIMEOUT
E6A9 E660 1300 G10: ; IN AL,PORT_A ;READ KEYBOARD SCAN CODE
E6A9 B40D 1301 MDV BL.AL ;SET SCAN CODE JUST READ
E6A9 B0CC 1302 MDV AL.OCH ;CLEAR KEYBOARD
E6AF E661 1303 OUT PORT_B,AL
E6B1 C3 1304 RET ;RETURN TO CALLER
E6B5 E661 1305 KBD_RESET ENDP
1306 ;
1307 ; BLINK LED PROCEDURE FOR MFG BURN-IN AND RUN-IN TESTS
1308 ; (LED WILL BLINK APPROXIMATELY .25 SECOND)
1309 ;

E6B2 1310 BLINK_INT PROC NEAR
E6B3 FB 1311 STI
E6B5 SI 1312 PUSH CX ;SAVE CX REG CONTENTS
E6B6 SI 1313 PUSH AX ;SAVE AX REG CONTENTS
E6B7 E661 1314 IN AL,PORT_B ;READ CURRENT VAL OF PORT B
E6B7 240F 1315 SUB AL,0BFH ;POSITION CURSOR TILL MSG WRITTEN
E6B9 E661 1316 OUT PORT_B,AL ;BLINK LED
E6B9 2BC9 1317 SUB CX,CX
E6B9 0C40 1318 G11: LOOP G11
E6B9 0C40 1319 OR AL,40H ;STOP BLINKING LED
E6C1 E661 1320 OUT PORT_B,AL
E6C3 B020 1321 MDV AL.EOF ;RESTORE AX REG
E6C5 B620 1322 OUT INTA00,AL ;RESTORE AX REG
E6C5 5B 1323 POP AX
E6C6 59 1324 POP CX
E6C7 CF 1325 IRET

E6C7 1326 BLINK_INT ENDP
1327 ;
1328 ; THIS SUBROUTINE WILL PRINT A MESSAGE ON THE DISPLAY
1329 ;
1330 ;ENTRY REQUIREMENTS:
1331 ; SI = OFFSET(ADDRESS) OF MESSAGE BUFFER
1332 ; CX = MESSAGE BYTE COUNT
1333 ; MAXIMUM MESSAGE LENGTH IS 36 CHARACTERS
1334 ;

E6CA 1335 PROC P_MSG NEAR
E6CA B80000 R 1336 MDV AX.DATA ;POINT DS TO DATA SEG
E6CD 8E08 R 1337 MDV DS:AX
E6CF 003E120001 R 1338 CMP MFG.TST.1 ;MFG TEST MODE
E6D4 7505 1339 JNE G1Z ;NO - DISPLAY ERROR MSG
E6D6 B601 ; MDV OH.1 ;YES - SETUP TO BEEP SPEAKER
E6D8 E95FF 1340 JMP ERP_BEEP ;YES - BEEP SPEAKER
E6DB 1342 G1Z: ;WRITE_MSG:
E6DB 2B0A04 1343 MDV AL.CS:151I ;PUT CHAR IN AL
E6DE 46 1344 INC SI ;POINT TO NEXT CHAR
E6DF B700 1345 MDV OH.0 ;SET PAGE B TO ZERO
E6E1 B40E 1346 MDV AH.14 ;WRITE CHAR (TTY-INTERFACE)
E6E3 C010 1347 INT 10H ;CALL VIDEO.IO
E6E5 E2F4 1348 LOOP G1Z ;CONTINUE TILL MSG WRITTEN
E6E7 B0000E 1349 MDV AX.0ED9 ;POSITION CURSOR TO NEXT LINE
E6EA C010 1350 INT 10H ;SEND CARRIAGE RETURN AND
E6EC 050AE E6EC 050AE 1351 MOV AX,0E9H ;LINE FEED CHAR
E6EC 050C E6EC 050C 1352 INT 10H
E6EC 050E E6EC 050E 1353 RET
E6EC 0510 E6EC 0510 1354 P MSG E9H
E6EC 0512 E6EC 0512 1355 ------ INT 19 ----------------------------------
E6EC 0514 E6EC 0514 1356 :BOOT STRAP LOADER
E6EC 0516 E6EC 0516 1357 ; IF A 5 1/4" DISKETTE DRIVE IS AVAILABLE
E6EC 0518 E6EC 0518 1358 ; ON THE SYSTEM, TRACK 0, SECTOR 1 IS READ INTO THE
E6EC 051A E6EC 051A 1359 ; BOOT LOCATION (SEGMENT 0 OFFSET 7CD0)
E6EC 051C E6EC 051C 1360 ; AND CONTROL IS TRANSFERRED THERE.
E6EC 051E E6EC 051E 1361 ; IF THERE IS NO DISKETTE DRIVE, OR IF THERE IS
E6EC 0520 E6EC 0520 1362 ; IS A HARDWARE ERROR CONTROL IS TRANSFERRED
E6EC 0522 E6EC 0522 1364 ; TO THE CASSETTE BASIC ENTRY POINT.
E6EC 0524 E6EC 0524 1365 ;
E6EC 0526 E6EC 0526 1366 ; IPL ASSUMPTION:
E6EC 0528 E6EC 0528 1367 ; 6255 PORT 60H BIT 0
E6EC 052A E6EC 052A 1369 ; = 1 IF IPL FROM DISKETTE
E6EC 052C E6EC 052C 1370 ; Assumes CS:CODE,DS:DATA
E6EC 052E E6EC 052E 1371 :BOOT STRAP PROC NEAR
E6EC 0530 E6EC 0530 1372
E6EC 0532 E6EC 0532 1373 E6F2 FB
E6EC 0534 E6EC 0534 1374 E6F3 B84000 R
E6EC 0536 E6EC 0536 1375 E6F6 0E00
E6EC 0538 E6EC 0538 1376 E6F8 A11000 R
E6EC 053A E6EC 053A 1377 E6FD 7423
E6EC 053C E6EC 053C 1378 E6FD 7423
E6EC 053E E6EC 053E 1379
E6EC 0540 E6EC 0540 1380 ;------- MUST LOAD SYSTEM FROM DISKETTE -- CX HAS RETRY COUNT
E6EC 0542 E6EC 0542 1381
E6EC 0544 E6EC 0544 1382 E6F8 B90400
E6EC 0546 E6EC 0546 1383 E70E
E6EC 0548 E6EC 0548 1384 E702 31
E6EC 054A E6EC 054A 1385 E703 0400
E6EC 054C E6EC 054C 1386 E705 CD13
E6EC 054E E6EC 054E 1387 E707 7216
E6EC 0550 E6EC 0550 1388 E709 0402
E6EC 0552 E6EC 0552 1389 E700 BBO000
E6EC 0554 E6EC 0554 1390 E70E BEC3
E6EC 0556 E6EC 0556 1391 E710 BBO07C
E6EC 0558 E6EC 0558 1392 E713 BAO000
E6EC 055A E6EC 055A 1393 E716 B91000
E6EC 055C E6EC 055C 1394 E719 B901
E6EC 055E E6EC 055E 1395 E71B CD13
E6EC 0560 E6EC 0560 1396 E71D 59
E6EC 0562 E6EC 0562 1397 E71E 7304
E6EC 0564 E6EC 0564 1398 E720 E220
E6EC 0566 E6EC 0566 1399
E6EC 0568 E6EC 0568 1400 ;------- UNABLE TO IPL FROM THE DISKETTE
E6EC 056A E6EC 056A 1401
E6EC 056C E6EC 056C 1402 E722 CD16
E6EC 056E E6EC 056E 1403 E722 CD16
E6EC 0570 E6EC 0570 1404
E6EC 0572 E6EC 0572 1405 ;------ IPL WAS SUCCESSFUL
E6EC 0574 E6EC 0574 1406
E6EC 0576 E6EC 0576 1407 E724
E6EC 0578 E6EC 0578 1408 E724 EA007C0000
E6EC 057A E6EC 057A 1409 :BOOT STRAP PROC NEAR
E6EC 057C E6EC 057C 1410 ;------ INT 19----------------------------------
E6EC 057E E6EC 057E 1411 ;RS232_IO
E6EC 0580 E6EC 0580 1412 ; THIS ROUTINE PROVIDES BYTE STREAM I/O TO THE COMMUNICATIONS
E6EC 0582 E6EC 0582 1413 ; PORT ACCORDING TO THE PARAMETERS:
E6EC 0584 E6EC 0584 1414 ; (AH)=0 INITIALIZE THE COMMUNICATIONS PORT
E6EC 0586 E6EC 0586 1415 ; (AL) HAS PARMS FOR INITIALIZATION
E6EC 0588 E6EC 0588 1416 ;
E6EC 058A E6EC 058A 1417 ; 7 6 5 4 3 2 1 0
E6EC 058C E6EC 058C 1418 ; ------ BAUD RATE -- PARITY-- STOPBIT --WORD LENGTH--
E6EC 058E E6EC 058E 1419
E6EC 0590 E6EC 0590 1420 ; 000 0 1 10 7 BITS
E6EC 0592 E6EC 0592 1421 ; 001 150 01 2 11 0 BITS
E6EC 0594 E6EC 0594 1422 ; 010 300 11 EVEN
E6EC 0596 E6EC 0596 1423 ; 011 400
E6EC 0598 E6EC 0598 1424 ; 100 1200
E6EC 059A E6EC 059A 1425 ; 101 2400
E6EC 059C E6EC 059C 1426 ; 110 4800
E6EC 059E E6EC 059E 1427 ; 111 9600

A-20
E729
1471 A1 LABEL WORD
E729 1704
E72B 0003
E72B 0001
E72F 0000
E731 0000
E733 0000
E735 0000
E737 0000
E739
RS232_TO
PROC
FAR

ASSUME CS:CODE,DS:DATA

E729 1704
1472 DW 1047 ; 110 Baud ; Table of Init Value
E72B 0003
1473 DW 768 ; 150
E72B 0001
1474 DW 354 ; 300
E72F 0000
1475 DW 192 ; 600
E731 0000
1476 DW 96 ; 1200
E733 0000
1477 DW 48 ; 2400
E735 0000
1478 DW 24 ; 4800
E737 0000
1479 DW 12 ; 9600

E739
1481 RS232_TO

;------- Vector to Appropriate Routine

E739 FB
1483 ; STI ; Interrupts Back On
E73A 1E
1485 PUSH DS ; Save Segment
E73B 52
1486 PUSH DX
E73C 56
1487 PUSH SI
E73D 57
1488 PUSH DI
E73E 51
1489 PUSH CX
E73F 0020
1490 MOV SI,DX ; RS232 Value to SI
E741 0100
1492 SHL SI,1 ; Word Offset
E743 0000000 R 1493 MOV DX,DATA
E744 BEDA
1494 MOV DS,DX ; Set Up Our Segment
E746 00000000 R 1495 MOV DX,R8232_BASE[SI] ; Get Base Address
E74C 00000000 R 1496 OR DX,DX ; Test for 0 Base Address
E74E 7416
1497 JZ A3 ; Return
E750 0AE4
1498 OR AH,AL ; Test for AH=0
E752 7418
1499 JZ A4 ; Common Init
E754 FECC
1500 DEC AH ; Test for AH=1
E756 744E
1501 JZ A5 ; Send AL
E758 FECC
1502 DEC AH ; Test for AH=2
E75A 7503
1503 JNZ A2
E75C 00000000 R 1504 JMP AI2 ; Receive Into AL
LOC OBJ  | LINE  | SOURCE
---|---|---
E75F  | 1505 | A2:  
E75F FECC  | 1506 | DEC AH  | TEST FOR AH=3
E761 7901  | 1507 | JNZ A3  
E763 8EB900  | 1508 | JMP A16  | COMMUNICATION STATUS
E766  | 1509 | A3:  | RETURN FROM RS232
E766 59  | 1510 | POP CX  
E767 5F  | 1511 | POP DI  
E768 SE  | 1512 | POP SI  
E769 8A  | 1513 | POP DX  
E76A 1F  | 1514 | POP DS  
E76B 8F  | 1515 | IRET  | RETURN TO CALLER, NO ACTION
E76C  | 1516 |  
E76C 8AE0  | 1520 | MOV AH,AL  | SAVE INITIAMS IN AH
E76E 83C203  | 1521 | ADD DX,3  | POINT TO B250 CONTROL REGISTER
E771 B080  | 1522 | MOV AL,00H  
E773 EE  | 1523 | OUT DX,AL  | SET DLB=1
E774  | 1524 |  
E774 8AD4  | 1525 | MOV DL,AL  | DETERMINE BAUD RATE DIVISOR
E776 DOC2  | 1526 |  
E778 DOC2  | 1527 | MOV DL,1  | GET PARMS TO DL
E77A DOC2  | 1528 | ROL DL,1  | GET BAUD RATE TERM TO LOW BITS
E77C DOC2  | 1529 | ROL DL,1  | GET PARMS TO DL
E77E 81E2000  | 1530 | ADD DX,0EH  | ISOLATE THEM
E782 B2F2E7  | 1531 | MOV DI,OFFSET A1  | BASE OF TABLE
E785 3F3A  | 1532 | ADD DI,DX  | PUT INTO INDEX REGISTER
E787 B0900000  | 1533 | MOV DX,RS232_BASE[SI]  | POINT TO HIGH ORDER OF DIVISOR
E78B 62  | 1534 | INC DX  
E78C 2EA4501  | 1535 | MOV AL,CS:[DI]+1  | GET HIGH ORDER OF DIVISOR
E790 90  | 1536 | OUT DX,AL  | SET MS OF DIX TO 0
E791 4A  | 1537 | DEC DX  
E792 2BA05  | 1538 | MOV AL,CS:[DI]  | GET LOW ORDER OF DIVISOR
E795 EE  | 1539 | OUT DX,AL  | GET PARMS TO DL
E796 83C203  | 1540 | ADD DX,3  | GET PARMS TO DL
E799 8AC4  | 1541 | MOV AL,AH  | GET PARMS BACK
E79B 241F  | 1542 | NND AL,01FH  | STRIP OFF THE BAUD BITS
E79D 83EA02  | 1543 | OUT DX,AL  | LINE CONTROL TO 8 BITS
E79E 83EA02  | 1544 | SUB DX,2  
E7A1 B000  | 1545 | MOV AL,0  
E7A3 EE  | 1546 | OUT DX,AL  | INTERRUPT ENABLES ALL OFF
E7A4 ED79  | 1547 | JMP SHORT A18  | COM_ STATUS
E7AA  | 1548 |  
E7AA A5:  | 1549 |  
E7A6 50  | 1550 | PUSH AX  | SEND CHARACTER IN (AL) OVER COMM LINE
E7A7 83C204  | 1551 | ADD DX,4  | MODEM CONTROL REGISTER
E7AA B003  | 1552 | MOV AL,3  | DIV AND RTS
E7AC EE  | 1553 | OUT DX,AL  | DATA TERMINAL READY, REQUEST TO SEND
E7AD 33C9  | 1554 | XOR CX,CX  | INITIALIZE TIME OUT COUNT
E7AF 83C202  | 1555 | ADD DX,2  | MODEM STATUS REGISTER
E7B2  | 1556 |  
E7B2 A6:  | 1557 |  
E7B2 EC  | 1558 | IN AL,DX  | WAIT_DATA_SET_READY
E7B3 A800  | 1559 | TEST AL,20H  | DATA SET READY
E7B5 7500  | 1560 | JNZ A7  | TEST_CLEAR_TO_SEND
E7B7 E2F9  | 1561 | LOOP A6  | WAIT_DATA_SET_READY
E7B9 59  | 1562 | POP AX  
E7BA 06CC50  | 1563 | OR AH,00  | INDICATE TIME OUT
E7BD E8A7  | 1564 | JMP A3  | RETURN
E7BF 56C9  | 1565 | A7:  | TEST_CLEAR_TO_SEND
E7C1  | 1566 |  
E7C1 A8:  | 1567 |  
E7C1 EC  | 1568 | IN AL,DX  | WAIT_CLEAR_TO_SEND
E7C2 A610  | 1569 | TEST AL,10H  | GET MODEM STATUS
E7C4 7500  | 1570 | JNZ A9  | CLEAR_TO_SEND
E7C6 E2F9  | 1571 | LOOP A8  | WAIT_CLEAR_TO_SEND
E7C8 50  | 1572 | POP AX  | TIME OUT HAS OCCURRED
E7C9 00CC00  | 1573 | OR AH,80H  
E7CC E800  | 1574 | JMP A3  | RETURN
E7CE  | 1575 |  
E7CE A9:  | 1576 |  
E7CE 4A  | 1577 | DEC DX  | CLEAR_TO_SEND
E7CF 2EC9  | 1578 | SUB CX,CX  | INITIALIZE WAIT COUNT
E7D1  | 1579 | A10:  | WAIT_SEND

A-22
APPENDIX A

E70D EC 1502 IN AL,DX ; GET STATUS
E70E A920 1503 TEST AL,20H ; IS TRANSMITTER READY
E70F 7500 1504 JNZ A11 ; OUT_CHAR
E70E E5F9 1505 LOOP A10 ; GO BACK FOR MORE, AND TEST FOR TIME OUT
E70D 5B 1506 POP AX ; RECOVER ORIGINAL INPUT
E70F 80CC00 1507 OR AH,80H ; SET THE TIME OUT BIT
E70C E800 1508 JMP A3 ; RETURN
E70E 1509 A11: ; OUT_CHAR
E70D 83EA05 1509 SUB DX,S ; DATA PORT
E70E 59 1510 POP CX ; RECOVER IN CX TEMPORARILY
E70E 8AC1 1512 MOV AL,CL ; GET OUT CHAR TO AL FOR OUT, STATUS IN AH
E70E 4E ; OUT DX,AL ; OUTPUT CHARACTER
E70E E97EFF 1514 JMP A3 ; RETURN
E70E 1515 ;----- RECEIVE CHARACTER FROM COMMO LINE
E70E 1516
E70E 002671007F R 1519 AND BIOS_BREAK,07FH ; TURN OFF BREAK BIT IN BYTE
E70E 00C2004 1600 ADD DX,4 ; MODEM CONTROL REGISTER
E70F 80B01 1601 MOV AL,1 ; DATA TERMINAL READY
E70F EE 1602 OUT DX,AL
E70E 83C002 1603 ADD DX,2 ; MODEM STATUS REGISTER
E70F E8C9 1604 SUB CX,CX ; ESTABLISH TIME OUT COUNT
E70F 1605 A13: ; WAIT_DSR
E70F 8C 1606 IN AL,DX ; MODEM STATUS
E70E 8202 1607 TEST AL,20H ; DATA SET READY
E70F 7507 1608 JNZ A15 ; IS IT READY YET
E70E 82F9 1609 LOOP A13 ; WAIT UNTIL IT IS
E70F 1610 A14: ; TIME_OUT_ERR
E70E B460 1611 MOV AH,40H ; SET TIME OUT ERROR
E70E 1612 JMP A3 ; RETURN WITH ERROR
E70E 1613 A15: ; WAIT_DSR_END
E70E 4A 1614 DEC DX ; LINE STATUS REGISTER
E70E 1615 A16: ; WAIT_RECV
E70E EC 1616 IN AL,DX ; GET STATUS
E70E 8001 1617 TEST AL,1 ; RECEIVE BUFFER FULL
E70E 7509 1618 JNZ A17 ; GET CHAR
E70E F606710000 R 1619 TEST BIOS_BREAK,06H ; TEST FOR BREAK KEY
E70E 74F4 1620 JZ A16 ; LOOP IF NOT
E70E 1621 JMP A14 ; SET TIME OUT ERROR
E70E 1622 A17: ; GET_CHAR
E70E 241E 1623 AND AL,0011110B ; TEST FOR ERROR CONDITIONS ON RECVC CHAR
E70E 8AE0 1624 MOV AH,4L ; SAVE THIS PART OF STATUS FOR LATER OPERATION
E70E 1625 MOV DX,RS232_BASE[SI] ; DATA PORT
E70E EC 1626 IN AL,DX ; GET CHARACTER FROM LINE
E70E E97EFF 1627 JMP A3 ; RETURN
E70E 1628
E70E 1629 ;----- COMMO PORT STATUS ROUTINE
E70E 1630
E70E 85F 1631 A18: ; RS232 to END
E70E 8B000000 R 1632 MOV DX,RS232_BASE[SI]
E70E 83C205 1633 ADD DX,5 ; CONTROL PORT
E70E 86C 1634 IN AL,DX ; GET LINE CONTROL STATUS
E70E 827 8A0 1635 MOV AH,AL ; PUT IN AH FOR RETURN
E70E 42 1636 INC DX ; POINT TO MODEM STATUS REGISTER
E70E 828 1637 IN AL,DX ; GET MODEM CONTROL STATUS
E70E 83B0 1638 JMP A3 ; RETURN
E70E 1639
E70E 1640 ;----- INT 16 ---------------------------------------------
E70E 1641 ; KEYBOARD I/O
E70E 1642 ; THESE ROUTINES PROVIDE KEYBOARD SUPPORT
E70E 1643 ; INPUT
E70E 1644 ; (AH)=0 READ THE NEXT ASCII CHARACTER STRUCK FROM THE KEYBOARD
E70E 1645 ; RETURN THE RESULT IN (AL), SCAN CODE IN (AH)
E70E 1646 ; (AH)=1 SET THE Z FLAG TO INDICATE IF AN ASCII CHARACTER IS AVAILABLE
E70E 1647 ; TO BE READ.
E70E 1648 ; (ZF)=1 -- NO CODE AVAILABLE
E70E 1649 ; (ZF)=0 -- CODE IS AVAILABLE
E70E 1650 ; IF ZF = 0, THE NEXT CHARACTER IN THE BUFFER TO BE READ IS
E70E 1651 ; IN AX, AND THE ENTRY REMAINS IN THE BUFFER
E70E 1652 ; (AH)=2 RETURN THE CURRENT SHIFT STATUS IN AL REGISTER
E70E 1653 ; THE BIT SETTINGS FOR THIS CODE ARE INDICATED IN THE
E70E 1654 ; EQUATIONS FOR KB_FLAG
E70E 1655 ; OUTPUT
E70E 1656 ; AS NOTED ABOVE, ONLY AX AND FLAGS CHANGED
E70E 1657 ; ALL REGISTERS RETAINED
E70E 1658 ;---------------------------------
LOC OBJ

1659 ASSUME CS:CODE,DS:DATA

E08E FB 1661  STI  ; INTERRUPTS BACK ON
E08F 1662  PUSH DS  ; SAVE CURRENT DS
E090 1663  PUSH BX  ; SAVE BX TEMPORARILY
E091 BD6000 1664  MOV BX DATA  ;
E354 1665  MOV DS:BX  ; ESTABLISH POINTER TO DATA REGION
E356 1666  OR AH, AH  ; AH=0
E358 740B 1667  JZ K1  ; ASCII_READ
E35A FECC 1668  DEC AH  ; AH=1
E35C 7420 1669  JZ K2  ; ASCII_STATUS
E35E FECC 1670  DEC AH  ; AH=2
E360 742D 1671  JZ K3  ; SHIFT_STATUS
E362 5B 1672  POP BX  ; RECOVER REGISTER
E364 IF 1673  POP DS  ;
E366 CF 1674  IRET  ; INVALID COMMAND

1679 ;------ READ THE KEY TO FIGURE OUT WHAT TO DO
E368 K1: 1679  STI  ; ASCII_READ
E369 FB 1679  STI  ; INTERRUPTS BACK ON DURING LOOP
E36F 90 1680  MOV  ; ALLOW AN INTERRUPT TO OCCUR
E370 FA 1681  CLI  ; INTERRUPTS BACK OFF
E374 081E1A00 1682  MOV BX,BUFFER_HEAD  ; GET POINTER TO HEAD OF BUFFER
E378 381E1C00 1683  CMP BX,BUFFER_TAIL  ; TEST END OF BUFFER
E37C 74F3 1684  JZ K1  ; LOOP UNTIL SOMETHING IN BUFFER
E380 0807 1685  MOV AX,[BX]  ; GET SCAN CODE AND ASCII CODE
E384 5B 1686  CALL K4  ; MOVE POINTER TO NEXT POSITION
E386 091E1A00 1687  MOV BX,BUFFER_HEAD,BX  ; STORE VALUE IN VARIABLE
E388 5B 1688  POP BX  ; RECOVER REGISTER
E38A IF 1689  POP DS  ; RECOVER SEGMENT
E38C CF 1690  IRET  ; RETURN TO CALLER

1692 ;------ ASCII STATUS

E38E K2: 1692  CLI  ; INTERRUPTS OFF
E390 FA 1695  MOV BX,BUFFER_HEAD  ; GET HEAD POINTER
E394 381E1C00 1696  CMP BX,BUFFER_TAIL  ; IF EQUAL (Z=1) THEN NOTHING THERE
E398 0807 1697  MOV AX,[BX]  ;
E39C FB 1699  STI  ; INTERRUPTS BACK ON
E39E 5B 1700  POP BX  ; RECOVER REGISTER
E3A0 IF 1701  POP DS  ; RECOVER SEGMENT
E3A2 CA0200 1702  RET 2  ; THROW AWAY FLAGS

1703 ;------ SHIFT STATUS

E3A6 K3: 1706  MOV AL,KD_FLAG  ; GET THE SHIFT STATUS FLAGS
E3AB 5B 1707  POP BX  ; RECOVER REGISTER
E3A9 IF 1709  POP DS  ; RECOVER REGISTERS
E3AC CF 1710  IRET  ; RETURN TO CALLER

1710 KEYBOARD_ID  ; ENDP

1712

1713 ;------ INCREMENT A BUFFER POINTER

1714

E3AF K4 1715  PROC NEAR
E3B5 85C3C2 1716  ADD BX,2  ; MOVE TO NEXT WORD IN LIST
E3B7 81F03EC0 1717  CMP BX,OFFSET KB_BUFFER_END  ; AT END OF BUFFER?
E3C7 7503 1718  JNE K5  ; NO, CONTINUE
E3CE BD1E00 1719  MOV BX,OFFSET KB_BUFFER  ; YES, RESET TO BUFFER BEGINNING
E3E1 1720 K5:
E3E1 C3 1721  RET
E3E2 1722 K4  ; ENDP

1723 ;------ TABLE OF SHIFT KEYS AND MASK VALUES

1724

E3E4 K6 1726  LABEL BYTE
E3E8 S2 1727  DB INS_KEY  ; INSERT KEY
E3ED 3A4546381D 1728  DB CAPS_KEY,MEM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
E3EE 2A36 1729  DB LEFT_KEY,RIGHT_KEY
E3F0 1730 K6L 1731 IRJ $-K6
E3F1 1732 ;------ SHIFT_MASK_TABLE
E3F3 1734 K7  LABEL BYTE
E3F7 80 1735  DB INS_SHIFT  ; INSERT MODE SHIFT

A-24
LOC OBJ   LINE   SOURCE
E994 BEE0  1798 MOV DS,AX ; SET UP ADDRESSING
E996 E640  1799 IN AL, KB_DATA ; READ IN THE CHARACTER
E996 500  1800  PUSH AX ; SAVE IT
E999 E641  1801 IN AL, KB_CTL ; GET THE CONTROL PORT
E99B 6AE0  1802 MOV AH, AL ; SAVE VALUE
E99C 0C00  1803 OR AL, 80H ; RESET BIT FOR KEYBOARD
E99F E641  1804 OUT KB_CTL, AL
E9A1 66E0  1805 XCHG AH, AL ; GET BACK ORİGINAL CONTROL
E9A3 E641  1806 OUT KB_CTL, AL ; KB HAS BEEN RESET
E9A5 5B0  1807 POP AX ; RECOVER SCAN CODE
E9A6 6AE0  1808 MOV AH, AL ; SAVE SCAN CODE IN AH ALSO
E9A9 000  1809
1810 i----- TEST FOR OVERRUN SCAN CODE FROM KEYBOARD
E9AB 3CFF  1811 CMP AL, 0FFH ; IS THIS AN OVERRUN CHAR
E9AA 7503  1812 JNZ K16 ; NO, TEST FOR SHIFT KEY
E9AE 675202  1814 JMP K22 ; BUFFER_FULL_BEEP
E9A7 1815
1816 i----- TEST FOR SHIFT KEYS
E9A8 1A18  1817 K16: ; TEST_SHIFT
E9A9 247F  1818 AND AL, 07FH ; TURN OFF THE BREAK BIT
E9A8 0C0  1819  PUSH CS
E9B0 070  1820 POP ES
E9B3 BF82E6  R 1821 MOV DI,OFFSET K6 ; SHIFT KEY TABLE
E9B4 B90800  1822 MOV CX, 6L ; LENGTH
E9B8 F20  1823 REPE SCAES ; LOOK THROUGH THE TABLE FOR A MATCH
E9B9 AE0  1824
1825 MOV AL, AH ; RECOVER SCAN CODE
E9BA 7643  1826 JE K17 ; JUMP IF MATCH FOUND
E9BB E90000  1827 JIP K25 ; IF NO MATCH, THEN SHIFT NOT FOUND
E9BC 1828
1829 i----- SHIFT KEY FOUND
E9C2 81EF8E0  R 1830 K17: SUB DI, OFFSET K4+1 ; ADJUST PTR TO SCAN CODE MATCH
E9C6 2EAD5A0E0  R 1832 MOV AH, CS:DI ; GET MASK INTO AH
E9CA A800  1833 TEST AL, 60H ; TEST FOR BREAK KEY
E9CB 7554  1834 JNZ K23 ; BREAK_SHIFT_FOUND
E9DE 1835
1836 i----- SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE
E9E7 00C10  1837 CMP AH, SCROLL_SHIFT
E9E2 7307  1839 JAE K10 ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
E9E3 1840
1841 i----- PLAIN SHIFT KEY, SET SHIFT ON
E9E4 0B561700  R 1842 OR KB_FLAG, AH ; TURN ON SHIFT BIT
E9E5 E9300  1844 JMP K26 ; INTERRUPT_RETURN
E9E6 1845
1846 i----- TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
E9E7 1847
E9E8 1848 K10: ; SHIFT-TOGGLE
E9EB F606170004  R 1849 TEST KB_FLAG, CTI_SHIFT ; CHECK CTL SHIFT STATE
E9F2 7560  1850 JNZ K25 ; JUMP IF CTL STATE
E9F2 3C52  1851 CMP AL, INS_KEY ; CHECK FOR INSERT KEY
E9F2 7525  1852 JNZ K22 ; JUMP IF NOT INSERT KEY
E9F6 F606170008  R 1853 TEST KB_FLAG, ALT_SHIFT ; CHECK FOR ALTERNATE SHIFT
E9F8 7403  1854 JZ K19 ; JUMP IF NOT ALTERNATE SHIFT
E9FA E65090  1855 JIP K25 ; JUMP IF ALTERNATE SHIFT
E9FB F606170020  R 1856 K19: TEST KB_FLAG, NUM_STATE ; CHECK FOR BASE STATE
E9FD 7500  1857 JNZ K21 ; JUMP IF NUM LOCK IS ON
E9F7 F606170003  R 1858 TEST KB_FLAG, LEFT_SHIFT; RIGHT_SHIFT
E9FC 746D  1859 JZ K22 ; JUMP IF BASE STATE
E9FE 1860
1861 K20: ; NUMERIC ZERO, NOT INSERT KEY
E9FF 0B3052  1862 MOV AX, 5230H ; PUT OUT AN ASCII ZERO
EA01 E90001  1863 JMP K57 ; BUFFER_FILL
EA02 1864
1865 K21: ; MIGHT BE NUMERIC
EA04 F606170003  R 1866 TEST KB_FLAG, LEFT_SHIFT; RIGHT_SHIFT
EA09 7WF3  1866 JZ K20 ; JUMP NUMERIC, NOT INSERT
EA0A 1867
1868 K22: ; SHIFT TOGGLE KEY HIT; PROCESS IT
EA0B 42618000  R 1869 TEST AH, KB_FLAG_1 ; IS KEY ALREADY DEPRESSED
EA0F 7540  1870 JNZ K26 ; JUMP IF KEY ALREADY DEPRESSED
EA11 02618000  R 1871 OR KB_FLAG_1, AH ; INDICATE THAT THE KEY IS DEPRESSED
EA15 3261700  1872 XOR KB_FLAG, AH ; TOGGLE THE SHIFT STATE
EA19 3C52  1873 CMP AL, INS_KEY ; TEST FOR 1ST MAKE OF INSERT KEY
EA1B 7541  1874 JNE K26 ; JUMP IF NOT INSERT KEY

A-26
EA12 B00052 1075 MOV AX,INS_KEY+256 ; SET SCAN CODE INTO AH, 0 INTO AL
EA20 E9F01 1076 JMP K57 ; PUT INTO OUTPUT BUFFER

EA23 00 1077
EA23 80FC10 1078 KZ5: ; BREAK-SHIFT-FOUND
EA26 731A 1081 CMP AH,SCROLL_SHIFT ; IS THIS A TOGGLE KEY
EA28 F004 1082 JAE K24 ; YES, HANDLE BREAK TOGGLE
EA2A 20261700 R 1083 HLT AH ; INVERT MASK
EA2E 3C00 1084 CMP AL,ALT_KEY+00H ; IS THIS ALTERNATE SHIFT RELEASE
EA30 7502 1085 JNE K26 ; INTERRUPT_RETURN

1086

1087

1088 K24: ; BREAK-Toggle
EA62 F604 1089 HLT AH ; INVERT MASK
EA64 20261000 R 1090 AND KB_FLAG,AH ; INDICATE NO LONGER DEPRESSED
EA66 EB14 1091 JMP SHORT K26 ; INTERRUPT_RETURN

1092

1093

1094

1095

1096

1097

1098

1099

1100

1101

1102

1103

1104

1105

1106

1107

1108

1109

1110

1111

1112

1113

1114

1115

1116

1117

1118

1119

1120

1121

1122

1123

1124

1125

1126

1127

1128

1129

1130

1131

1132

1133

1134

1135

1136

1137

1138

1139

1140

1141

1142

1143

1144

1145

1146

1147

1148

1149

1150

EA65 19 1118 POP DS
EA66 5F 1119 POP DI
EA66 5E 1120 POP SI
EA67 5A 1121 POP DX
EA68 59 1122 POP CX
EA69 5B 1123 POP BX
EA6A 56 1124 POP AX ; RESTORE STATE
EA6B CF 1125 IRET ; RETURN, INTERRUPTS BACK ON WITH FLAG CHANGE

1126

1127

1128

1129

1130

1131

1132

1133

1134

1135

1136

1137

1138

1139

1140

1141

1142

1143

1144

1145

1146

1147

1148

1149

1150

EA6C 1913 K27: ; INTERRUPT-RETURN

EA6E FA 1914 CLI ; TURN OFF INTERRUPTS
EA5F B020 1915 MOV AL,EOI ; END OF INTERRUPT COMMAND
EA61 E620 1916 OUT 020H,AL ; SEND COMMAND TO INTERRUPT CONTROL PORT
EA63 1917 K26: ; INTERRUPT-RETURN-NO-EOI
EA65 07 1918 POP ES
EA66 1F 1919 POP DS
EA66 5F 1920 POP DI
EA66 5E 1921 POP SI
EA67 5A 1922 POP DX
EA68 59 1923 POP CX
EA69 5B 1924 POP BX
EA6A 56 1925 POP AX
EA6B CF 1926 IRET ; RETURN, INTERRUPTS BACK ON WITH STATE

1927

1928

1929

1930

1931

1932

1933

1934

1935

1936

1937

1938

1939

1940

1941

1942

1943

1944

1945

1946

1947

1948

1949

1950

EA6C 1930 K28: ; NO-HOLD-STATE
EA6C F60610000 R 1931 TEST KB_FLAG,ALT_SHIFT ; ARE WE IN ALTERNATE SHIFT
EA71 7503 1932 JNZ K29 ; JUMP IF ALTERNATE SHIFT
EA73 E90F00 1933 JMP K30 ; JUMP IF NOT ALTERNATE

1934

1935

1936

1937

1938

1939

1940

1941

1942

1943

1944

1945

1946

1947

1948

1949

1950

EA6C 1937 K29: ; TEST-RESET
EA67 F60F70004 R 1938 TEST KB_FLAG,CTL_SHIFT ; ARE WE IN CONTROL SHIFT ALSO
EA7B 7431 1939 JZ K31 ; NO_RESET
EA7D 3C93 1940 CMP AL,DEL_KEY ; SHIFT STATE IS THERE, TEST KEY
EA7F 7520 1941 JNE K31 ; NO_RESET

1942

1943

1944

1945

1946

1947

1948

1949

1950

EA66 C70672083412 R 1945 MOV RESET_FLAG,1234H ; SET FLAG FOR RESET FUNCTION
EA67 E901FS 1946 JMP RESET ; JUMP TO POWER ON DIAGNOSTICS

1947

1948

1949

1950

EA80 52AF590514B04C040 DB 82,79,80,61,75,76,77
LOC OBJ | LINE | SOURCE
--- | --- | ---
EB10 B01E00 | R | 2026 | MOV BX,OFFSET KB_BUFFER ; RESET BUFFER TO EMPTY
EB13 B01E100 | R | 2027 | MOV BUFFER_HEAD,BX ;
EB17 B01E1C0 | R | 2028 | MOV BUFFER_TAIL,BX ;
EB18 C066710000 | R | 2029 | MOV BIOS_BREAK,00H ; TURN ON BIOS BREAK BIT
EB20 CD1B | 2030 | INT 18H ; BREAK INTERRUPT VECTOR
EB22 B00600 | 2031 | MOV AX, 0 ; PUT OUT DUMMY CHARACTER
EB25 E99400 | 2032 | JMP K57 ; BUFFER_FILL
EB28 | 2033 | K39: ; NO-BREAK
EB26 3C45 | 2035 | CMP AL,NM_KEY ; LOOK FOR PAUSE KEY
EB28 7521 | 2036 | JNE K41 ; NO-PAUSE
EB2C B06100008 | R | 2037 | OR KB_FLAG,1 ; HOLD_STATE ; TURN ON THE HOLD FLAG
EB31 B020 | 2038 | MOV AL,EOI ; END OF INTERRUPT TO CONTROL PORT
EB33 E620 | 2039 | OUT 020H,AL ; ALLOW FURTHER KEYSTROKE INTS
EB40 | 2040 | i----- DURING PAUSE INTERVAL, TURN CRT BACK ON
EB55 A93490007 | R | 2041 | CMP CRT_MODE,7 ; IS THIS BLACK AND WHITE CARD
EB5A 7407 | 2044 | JE K40 ; YES, NOTHING TO DO
EB5C B00503 | 2045 | MOV DX,0308H ; PORT FOR COLOR CARD
EB5F A06500 | R | 2046 | MOV AL,CRT_MODE_SET ; GET THE VALUE OF THE CURRENT MODE
EB64 EE | 2047 | OUT DX,AL ; SET THE CRT MODE, SO THAT CRT IS ON
EB65 | 2048 | K40: ; PAUSE-LOOP
EB69 F60610008 | R | 2049 | TEST KB_FLAG,1,HOLD_STATE
EB74 75F9 | 2050 | JNZ K40 ; LOOP UNTIL FLAG TURNED OFF
EB78 E916FF | 2051 | JMP K27 ; INTERRUPT_RETURN_NO_EOI
EB80 | 2052 | K41: ; NO-PAUSE
EB9D | 2053 | i----- TEST SPECIAL CASE KEY 55
EB9D 3C37 | 2055 | CMP AL,55
EBF4 7506 | 2057 | JNE K42 ; NOT-KEY-55
EB51 B0072 | 2058 | MOV AX,114*256 ; START/STOP PRINTING SWITCH
EB54 E95000 | 2059 | JMP K57 ; BUFFER_FILL
EB60 | 2060 | i----- SET UP TO TRANSLATE CONTROL SHIFT
EB57 | 2063 | K42: ; NOT-KEY-55
EB57 B092E8 | R | 2064 | MOV BX,OFFSET KB ; SET UP TO TRANSLATE CTL
EB5A 3C30 | 2065 | CMP AL,59 ; IS IT IN TABLE
EB5C 7303 | 2066 | JAE K43 ; CTL-TABLE-TRANSLATE
EB5E B07990 | 2067 | JMP K56 ; YES, GO TRANSLATE CHAR
EB61 | 2068 | K43: ; CTL-TABLE-TRANSLATE
EB61 B0CCE8 | R | 2069 | MOV BX,OFFSET K9 ; CTL TABLE SCAN
EB64 EE3C00 | 2070 | JMP K63 ; TRANSLATE_SCAN
EB71 | 2071 | i----- NOT IN CONTROL SHIFT
EB67 | 2074 | K44: ; NOT-CTL-SHIFT
EB67 3C47 | 2076 | CMP AL,71 ; TEST FOR KEYPAD REGION
EB69 7320 | 2077 | JAE K46 ; HANDLE KEYPAD REGION
EB6B F606170003 | R | 2078 | TEST KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT
EB7D 7540 | 2079 | JZ K54 ; TEST FOR SHIFT STATE
EB80 | 2080 | i----- UPPER CASE, HANDLE SPECIAL CASES
EB72 3C0F | 2083 | CMP AL,15 ; BACK TAB KEY
EB74 7506 | 2084 | JNE K45 ; NOT-BACK-TAB
EB76 B00508 | 2085 | MOV AX,15*256 ; SET PSEUDO SCAN CODE
EB77 E06190 | 2086 | JMP K57 ; BUFFER_FILL
EB78 | 2087 | i----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
EB7C 3C37 | 2089 | CMP AL,55 ; PRINT SCREEN KEY
EBDC 7508 | 2090 | JNE K46 ; NOT-PRINT-SCREEN
EBE7 | 2091 | i----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
EBE8 B020 | 2094 | MOV AL,EOI ; END OF CURRENT INTERRUPT
EBE2 E620 | 2095 | OUT 020H,AL ; SO FURTHER THINGS CAN HAPPEN
EBE4 CD05 | 2096 | INT 5H ; ISSUE PRINT SCREEN INTERRUPT
EBE6 E01AF | 2097 | JMP K27 ; GO BACK WITHOUT EOI OCCURRING
EBE9 | 2098 | K46: ; NOT-PRINT-SCREEN
EBE9 3C38 | 2100 | CMP AL,59 ; FUNCTION KEYS
EBE0 7206 | 2101 | JB K47 ; NOT-UPPER-FUNCTION
EBE0 E059E9 | R | 2102 | MOV BX,OFFSET K12 ; UPPER CASE PSEUDO SCAN CODES
LOC OBJ  

EBD0 2103 JMP $63 ; TRANSLATE_SCAN

EB93 2105 K97: ; NOT-UPPER-FUNCTION

EB95 BB1FE9 R 2106 MOV BX,OFFSET K11 ; POINT TO UPPER CASE TABLE

EB96 2107 JMP SHORT $56 ; OK, TRANSLATE THE CHAR

EB98 2109 ;----- KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION

EB9A 2111 K48: ; KEYPAD-REGION

EB9C 2112 TEST KB_FLAG,NUM_STATE ; ARE WE IN NUM_LOCK

EB9D 7520 JNZ K52 ; TEST FOR SURE

EB9F E6170003 R 2114 TEST KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE

EBA4 7520 2115 JNZ K53 ; IF SHIFTED, REALLY NUM STATE

EBAA 2116 ;---- BASE CASE FOR KEYPAD

EBA6 2119 K49: ; BASE-CASE

EBAD 2120 ;

EBAE 3C4A 2121 CMP AL,74 ; SPECIAL CASE FOR A COUPLE OF KEYS

EBBA 740B 2122 JE K50 ; MINUS

EBBB 3C4E 2123 CMP AL,78

EBBC 740C 2124 JE K51

EBBF 2125 SUB AL,71 ; CONVERT ORIGIN

EBB0 BB7AE9 2126 MOV BX,OFFSET K15 ; BASE CASE TABLE

EBB3 EB77 2127 JMP SHORT $64 ; CONVERT TO PSEUDO SCAN

EBB5 2128 ;

EBB6 BB204A 2129 K50: MOV AX,74*256+"-" ; MINUS

EBB8 EB22 2130 JMP SHORT $57 ; BUFFER.FILL

EBBA 2131

EBBA BB204E 2132 K51: MOV AX,78*256+"+" ; PLUS

EBBD EB10 2133 JMP SHORT $57 ; BUFFER.FILL

EBBE 2134 ;---- MIGHT BE NUM LOCK, TEST SHIFT STATUS

EBBF 2136 K52: ; ALMOST-NUM-STATE

EBBF F606170005 R 2138 TEST KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT

EBC4 7520 2139 JNZ K49 ; SHIFTED TEMP OUT OF NUM STATE

EBC6 2140

EBC6 2141 K53: ; REALLY_NUM_STATE

EBC8 2142 MOV BX,OFFSET K14 ; NUM STATE TABLE

EBCD EB0B 2144 JMP SHORT $56 ; TRANSLATE CHAR

EBCD 2145 ;---- PLAIN OLD LOWER CASE

EBCD 2146 ;

EBCC 2147 KB3B 2149 CMP AL,59 ; TEST FOR FUNCTION KEYS

EBCF 7204 2150 JB K59 ; NOT-LOWER-FUNCTION

EBD0 B000 2151 MOV AL,0 ; SCAN CODE IN AN ALREADY

EBD3 EB07 2152 JMP SHORT $57 ; BUFFER.FILL

EBD5 2153

EBD5 BB0E9 2154 K55: ; NOT-LOWER-FUNCTION

EBD5 BBEE58 R 2155 MOV BX,OFFSET K10 ; LC TABLE

EBD6 2156

EBD6 BBEE58 2157 ;----- TRANSLATE THE CHARACTER

EBD8 2158

EBD8 BB0B 2159 K56: ; TRANSLATE-CHAR

EBE0 FEC5 2160 DEC AL ; CONVERT ORIGIN

EBE0 2161 XLAT CS:K11 ; CONVERT THE SCAN CODE TO ASCII

EBE0 2162

EBE2 2163 ;----- PUT CHARACTER INTO BUFFER

EBE5 2164

EBE5 3CFF 2165 CMP AL,1 ; IS THIS AN IGNORE CHARACTER

EBE6 741F 2166 JE K59 ; YES, DO NOTHING WITH IT

EBE8 80FF0F 2167 CMP AH,1 ; LOOK FOR -1 PSEUDO SCAN

EBE9 741A 2168 JE K59 ; NEAR_INTERRUPT_RETURN

EBEC 2169

EBEC 2170 ;----- HANDLE THE CAPS LOCK PROBLEM

EBEC 2171 ;

EBED 2172 KB50: ; BUFFER-FILL-HOTTEST

EBED F606170040 R 2174 TEST KB_FLAG,CAPS_STATE ; ARE WE IN CAPS LOCK STATE

EBEA 7420 2175 JZ K61 ; SKIP IF NOT

EBEA 2176 ;----- IN CAPS LOCK STATE

EBEB 2177

EBEB F606170003 R 2179 TEST KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT ; TEST FOR SHIFT STATE

A-30
LOC OBJ LINE SOURCE

EBF 740F 2180 JZ K60 ; IF NOT SHIFT, CONVERT LOWER TO UPPER

2181

2182

EBF3 3C61 2184 CMP AL, 'A' ; FIND OUT IF ALPHABETIC

EBF5 7215 2185 JB K61 ; NOT_CAPS_STATE

EBF7 3C5A 2186 CMP AL, 'Z' ; FIND OUT IF ALPHABETIC

EBF9 7711 2187 JA K61 ; NOT_CAPS_STATE

EBF8 0420 2188 ADD AL, 'a' - 'A' ; CONVERT TO LOWER CASE

EBFD E800 2189 JMP SHORT K61 ; NOT_CAPS_STATE

EBF 2191 K59: ; HEAR-INTERRUPT-RETURN

EBFF E95CFE 2192 JMP K26 ; INTERRUPT_RETURN

2193

2194 ; ------ CONVERT ANY LOWER CASE TO UPPER CASE

2195

EC02 2196 K60: ; LOWER-TO-UPPER

EC02 3C61 2197 CMP AL, 'a' ; FIND OUT IF ALPHABETIC

EC04 7206 2198 JB K61 ; NOT_CAPS_STATE

EC06 3C74 2199 CMP AL, 'z' ; FIND OUT IF ALPHABETIC

EC08 770C 2200 JA K61 ; NOT_CAPS_STATE

EC0A 2C20 2201 SUB AL, 'a' - 'A' ; CONVERT TO UPPER CASE

EC0C 2202 K61: ; NOT-CAPS-STATE

EC0C E011C00 R 2203 MOV BX, BUFFER_TAIL ; GET THE END POINTER TO THE BUFFER

EC10 0F03 2205 MOV SI, BX ; SAVE THE VALUE

EC12 E060FC 2206 CALL K6 ; ADVANCE THE TAIL

EC15 3B11EA00 R 2207 CMP BX, BUFFER_HEAD ; HAS THE BUFFER WRAPPED AROUND

EC19 7409 2208 JE K62 ; BUFFER_FULL_BEEP

EC18 0994 2209 MOV SI, AX ; STORE THE VALUE

EC1D 0911C00 R 2210 MOV BUFFER_TAIL.BX, MOV THE POINTER UP

EC21 E934FE 2211 JMP K26 ; INTERRUPT_RETURN

EC22 2212

EC24 2213 K62: ; BUFFER-FULL-BEEP

EC26 E80000 2214 CALL ERROR_BEEP ; INTERRUPT_RETURN

EC27 E934FE 2215 JMP K26 ; INTERRUPT_RETURN

EC29 2216

EC2A 2217 ; ------ TRANSLATE SCAN FOR PSEUDO SCAN CODES

EC2A 2218

EC2A 2C3B 2222 SUB AL, 59 ; CONVERT ORIGIN TO FUNCTION KEYS

EC2C 2223 K64: ; TRANSLATE-SCAN-ORIG

EC2C 2D07 2224 XLAT C5:K9 ; CTL TABLE SCAN

EC2E BAO0 2225 MOV AH, AL ; PUT VALUE INTO AH

EC30 0000 2226 MOV AL, 0 ; ZERO ASCII CODE

EC32 E0A8 2227 JMP K57 ; PUT IT INTO THE BUFFER

EC32 2228

EC34 2229 KB_INT ENDP

EC34 2230 ERROR_BEEP PROC NEAR

EC34 50 2231 PUSH AX ; SAVE REGISTERS

EC35 53 2232 PUSH BX

EC35 51 2233 PUSH CX

EC37 BE000 2234 MOV BX, OC0H ; NUMBER OF CYCLES FOR 1/8 SECOND TONE

EC3A E461 2235 IN AL, KB_CTL ; GET CONTROL INFORMATION

EC3B 50 2236 PUSH AX ; SAVE

EC3D 2237 K65: ; BEEP-CYCLE

EC3D 24FC 2238 ADD AL, 0FCH ; TURN OFF TIMER GATE AND SPEAKER DATA

EC3F E661 2239 OUT KB_CTL, AL ; OUTPUT TO CONTROL

EC41 B4000 2240 MOV CX, 40H ; HALF CYCLE TIME FOR TONE

EC44 E2FE 2241 LOOP K66 ; SPEAKER OFF

EC46 0002 2242 MOV OR AL, 2 ; TURN ON SPEAKER BIT

EC48 E661 2243 OUT KB_CTL, AL ; OUTPUT TO CONTROL

EC4A B4000 2244 MOV CX, 40H ; SET UP COUNT

EC4C E2FE 2245 LOOP K67 ; ANOTHER HALF CYCLE

EC4F 4B 2246 DEC BX ; TOTAL TIME COUNT

EC55 75EB 2247 JNZ K45 ; DO ANOTHER CYCLE

EC52 50 2248 POP AX ; RECOVER CONTROL

EC53 E661 2249 OUT KB_CTL, AL ; OUTPUT THE CONTROL

EC55 59 2250 POP CX ; RECOVER REGISTERS

EC56 5B 2251 POP BX

EC57 50 2252 POP AX

EC58 C3 2253 RET

EC58 2254 ERROR_BEEP ENDP
NOTE: IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE APPROPRIATE ACTION IS TO RESET THE DISKETTE, THEN RETRY THE OPERATION.

ON READ ACCESSES, 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.
LOC OBJ  LINE  SOURCE

EC87 6AF0  2332  MOV DH,AL ; SAVE # SECTORS IN DH
EC89 0023F0077F  R  2333  AND MOTOR_STATUS,07FH ; INDICATE A READ OPERATION
EC90 0A84  2334  OH AH,AH ; AH=0
EC91 4727  2335  JZ DISK_RESET
EC92 4FCC  2336  DEC AH ; AH=1
EC94 747F  2337  JZ DISK_STATUS
EC96 C606410000  R  2338  MOV DISKETTE_STATUS,0 ; RESET THE STATUS INDICATOR
EC98 00FA04  2339  CMP DL,4 ; TEST FOR DRIVE IN 0-3 RANGE
EC9E 7313  2340  JAE J3 ; ERROR IF ABOVE
EC9F 4ECC  2341  DEC AH ; AH=2
ECAC 746A  2342  JZ DISK_READ
ECAE 4FCC  2343  DEC AH ; AH=3
ECA6 7903  2344  JNZ J2 ; TEST_DISK_VERF
ECA8 E96000  2345  JMP DISK_WRITE
ECA9  2346  JZ: ; TEST_DISK_VERF
ECAB 4ECC  2347  DEC AH ; AH=4
ECA0 7668  2348  JZ DISK_VERF
ECAF 4ECC  2349  DEC AH ; AH=5
ECB1 7668  2350  JZ DISK_FORMAT
ECB3  2351  J3: ; BAD COMMAND
ECB5 C606410001  R  2352  MOV DISKETTE_STATUS,BAD_CMD ; ERROR CODE. NO SECTORS TRANSFERRED
ECB8 C5  2353  RET ; UNDEFINED OPERATION
ECB9  2354  J1 ENDP

ECB9 2C58  2355  DISK_RESET PROC NEAR
ECB9 RAFF03  2359  MOV DX,03F2H ; ADAPTER CONTROL PORT
ECBC FA  2360  CLI ; NO INTERRUPTS
ECBD A03F00  R  2361  MOV AL,MOTOR_STATUS ; WHICH MOTOR IS ON
ECCE B104  2362  MOV CL,4 ; SHIFT COUNT
ECCE D2E9  2363  SAL AL,CL ; MOVE MOTOR VALUE TO HIGH NYBBLE
ECCE A4B2  2364  TEST AL,2MH ; SELECT CORRESPONDING DRIVE
ECCE 750C  2365  JNZ J5 ; JUMP IF MOTOR ONE IS ON
ECCE A800  2366  TEST AL,4MH ; JUMP IF MOTOR TWO IS ON
ECCE 7506  2367  JNZ J4 ; JUMP IF MOTOR TWO IS ON
ECCE A800  2368  TEST AL,8MH
ECCE 7406  2369  JZ J6 ; JUMP IF MOTOR ZERO IS ON
ECDF FEC0  2370  INC AL
ECDF 4ECC  2371  J6: ; INC AL
ECDF 4ECC  2372  J5: ; INC AL
ECDF 6C08  2373  J6: ; INC AL ; TURN ON INTERRUPT ENABLE
ECDF EE  2374  OUT DX,AL ; RESET THE ADAPTER
ECDF 6E063E000  R  2375  MOV SEEK_STATUS,0 ; SET RECAL REQUIRED ON ALL DRIVES
ECDF C606410000  R  2376  MOV DISKETTE_STATUS,0 ; SET OK STATUS FOR DISKETTE
EE00 C004  2377  OR AL,4 ; TURN OFF RESET
EE05 EE  2378  OUT DX,AL ; TURN OFF THE RESET
EE06 FB  2379  STI ; REENABLE THE INTERRUPTS
EE07 E2002  2380  CALL CHK_STAT ; DO Sense INTERRUPT STATUS FOLLOWING RESET
EEA4 A0200  R  2381  MOV AL,HEC_STATUS ; IGNORE ERROR RETURN AND DO ONLY TEST
EE0D 3C00  2382  JMP AL,0C0H ; TEST FOR DRIVE READY TRANSITION
EE0F 7407  2383  JZ J7 ; EVERYTHING OK
EEF9 8006140020  R  2384  OR DISKETTE_STATUS,BAD_REC ; SET ERROR CODE
EEF6 6B11  2385  JMP SHORT J9 ; RESET_RET

EEF8  2386  J9: ; DRIVE_READY
EEF9 DA03  2389  MOV AH,3H ; SPECIFY COMMAND
EEFA EA9701  2391  CALL NEC_OUTPUT ; OUTPUT THE COMMAND
EEFB B80100  2392  MOV BX,1 ; FIRST BYTE PARM IN BLOCK
EEFD E60001  2393  CALL GET_PARM ; TO THE NEC CONTROLLER
EE03 B00300  2394  MOV BX,1 ; SECOND BYTE PARM IN BLOCK
EE06 E66701  2395  CALL GET_PARM ; TO THE NEC CONTROLLER
EE09 7407  2396  J8: ; RESET_RET
EE0A C3  2397  RET ; RETURN TO CALLER
EE0B 2398  JMP SHORT J9 ; RESET_RET

EE0D  2399  DISK_RESET ENDP

EE0F  2400  DISKETTE_STATUS ROUTINE
EE10  2401
EE1A  2402  DISK_STATUS PROC NEAR
EE1B A04100  2403  MOV AL,DISKETTE_STATUS
EE1B C3  2404  RET

EE1D  2405  DISK_STATUS ENDP

A-33
DISKETTE TURN ON
ALLOW WRITE ROUTINE TO

ED06 E046 MOV AL,066H ; READ COMMAND FOR DMA
ED10 J9: ; DISK_READ Cont
ED10 E00901 CALL DMA_SETUP ; SET UP THE DMA
ED13 E044 MOV AH,066H ; SET UP READ COMMAND FOR NEC CONTROLLER
ED15 E036 JMP SHORT RM_OPN ; GO DO THE OPERATION

2415 DISK_READ E000

2417 ;------ DISKETTE VERIFY

ED07 E042 MOV AL,032H ; VERIFY COMMAND FOR DMA
ED19 E0F5 JMP J9 ; DO AS IF DISK READ

2422 DISK_VERIFY E008

2423 ;------ DISKETTE FORMAT

2425 ED1B DISK_FORMAT E01B

2426 ;------ DISKETTE READ

ED1B E00E3FO00 R 2427 OR MOTOR_STATUS,00H ; INDICATE WRITE OPERATION
ED20 E064 MOV AL,04AH ; WILL WRITE TO THE DISKETTE
ED22 E07A701 CALL DMA_SETUP ; SET UP THE DMA
ED25 E040 MOV AH,040H ; ESTABLISH THE FORMAT COMMAND
ED37 E024 JMP SHORT RM_OPN ; DO THE OPERATION
ED39 J10: ; CONTINUATION OF RM_OPN FOR FMT
ED49 E00700 MOV DX,7 ; GET THE
ED5C E04101 CALL GET_PARM ; BYTES/SECTOR VALUE TO NEC
ED5F E0090 MOV BX,9 ; GET THE
ED62 E03501 CALL GET_PARM ; SECTORS/TRACK VALUE TO NEC
ED65 E0400 MOV BX,15 ; GET THE
ED68 E03501 CALL GET_PARM ; GAP LENGTH VALUE TO NEC
ED6B E01100 MOV BX,17 ; GET THE FILLER BYTE
ED6E E94B00 JMP J16 ; TO THE CONTROLLER

2441 DISK_FORMAT E01E

2442 ;------ DISKETTE WRITE ROUTINE

2444 ED41 DISK_WRITE E041

2445 ED41 E00E3FO00 R 2446 OR MOTOR_STATUS,00H ; INDICATE WRITE OPERATION
ED46 E064 MOV AL,04AH ; DMA WRITE COMMAND
ED48 E08101 CALL DMA_SETUP
ED4B E044 MOV AH,045H ; NEC COMMAND TO WRITE TO DISKETTE

2450 DISK_WRITE E008

2451 ;------ ALLOW WRITE ROUTINE TO FALL INTO RM_OPN

2452 ;------------------------------------------

2453 ; RM_OPN

2454 ; THIS ROUTINE PERFORMS THE READ/WRITE/VERIFY OPERATION

2455 ;-------------------------------

2456 ED4D RM_OPN E04D

2456 ED4D 7300 JNC J11 ; TEST FOR DMA ERROR
ED4F E06410000 R 2457 MOV DISKETTE_STATUS,DMA_BOUNDARY ; SET ERROR
ED54 E009 MOV AL,0 ; NO SECTORS TRANSFERRED
ED56 E3 C3 RET ; RETURN TO MAIN ROUTINE
ED57 J11: ; DO_RM_OPN

2460 ED57 50 PUSH AX ; SAVE THE COMMAND

2462 ED55 ; TURN ON THE MOTOR AND SELECT THE DRIVE

2464 ED55 51 PUSH CX ; SAVE THE T/S PARMS

2466 ED59 8ACA MOV CL,DL ; GET DRIVE NUMBER AS SHIFT COUNT
ED5B E001 MOV AL,0 ; MASK FOR DETERMINING MOTOR BIT
ED5D E02E SAL AL,CL ; SHIFT THE MASK Bit
ED5F FA CLI ; NO INTERRUPTS WHILE DETERMINING MOTOR STATUS
ED60 E0644000F R 2471 MOV MOTOR_COUNT,0FH ; SET LARGE COUNT DURING OPERATION
ED64 E0463FO R 2472 TEST AL,MOTOR_STATUS,TEST THAT MOTOR FOR OPERATING
ED69 E751 JNZ J14 ; IF RUNNING, SKIP THE WAIT
ED6B E0063FO00 R 2474 AND MOTOR_STATUS,0FH ; TURN OFF ALL MOTOR BITS
ED70 E0063FO00 R 2475 OR MOTOR_STATUS,AL ; TURN ON THE CURRENT MOTOR
ED74 FB STI ; INTERRUPTS BACK ON

2476 ED75 8100 MOV AL,10H ; MASK BIT
ED77 D2E0 SAL AL,CL ; DEVELOP BIT MASK FOR MOTOR ENABLE
ED79 8AEC OR AL,DL ; GET DRIVE SELECT BITS IN
ED7C 8C00 OR AL,0CH ; NO RESET, ENABLE DMA/INT
ED7D 52 PUSH DX ; SAVE REG
ED7E BAF203 MOV DX,03F2H ; CONTROL PORT ADDRESS
ED81 EE OUT DX,AL

A-34
ED82 5A 2606  
PUSH AX  
RECOVER REGISTERS

ED83 F6635F0080  R 2608  
TEST MOTOR_STATUS,00H  
IS THIS A WRITE

ED90 7A12 2609  
JZ J14
NO, CONTINUE WITHOUT WAIT

ED9A AE00 2609  
MOV BX,20  
GET THE MOTOR WAIT

ED9C 6A4E 2609  
OR AH,AH  
PARAMETER

ED9D 2609  
TEST FOR NO WAIT

ED92 2609  
J12:  
TEST_WAIT_TIME

ED92 7A08 2609  
JZ J14  
EXIT WITH TIME EXPIRED

ED94 2609  
SUB CX,CX  
SET UP 1/8 SECOND LOOP TIME

ED96 FE2F 2609  
J13:  
WAIT FOR THE REQUIRED TIME

ED9D FECC 2609  
DEC AH  
DECREMENT TIME VALUE

ED9A EB56 2609  
JMP J12  
ARE WE DONE YET

ED9C 2609  
J14:  
MOTOR_RUNNING

ED9C FB 2609  
STI  
INTERUPTS BACK ON FOR BYPASS WAIT

ED9D S9 2609  
PUSH CX

ED9E 2609  
DO THE SEEK OPERATION

EDA1 5D 2609  
PUSH AX  
RECOVER COMMAND

EDA2 4AF 2609  
MOV DH,AH  
SAVE COMMAND IN BH

EDA4 6E00 2609  
MOV DH,0  
SET NO SECTORS READ IN CASE OF ERROR

EDA6 7240 2610  
JC J17  
IF ERROR, THEN EXIT AFTER MOTOR OFF

EDA8 BFE3D90  R 2611  
MOV SI,OFFSET J17  
DUMMY RETURN ON STACK FOR NEC_OUTPUT

EDAC 56 2612  
PUSH SI  
SO THAT IT WILL RETURN TO MOTOR OFF LOCATION

EDAE 2613  
SEND OUT THE PARAMETERS TO THE CONTROLLER

EDF0 00FF4D 2626  
CMP BH,040H  
IS THIS A FORMAT OPERATION

EDCH 7633 2627  
JNE J15
NO, CONTINUE WITH R/W/V

EDCA 962F 2628  
JMP J10  
IF SO, HANDLE SPECIAL

EDC7 6AE5 2629  
MOV AH,CH  
CYLINDER NUMBER

EDC9 E87000 2630  
CALL NEC_OUTPUT  
OUTPUT THE OPERATION COMMAND

EDC8 6A6601 2631  
MOV AH,1B4+11
GET THE CURRENT HEAD NUMBER

EDC5 D0E6 2631  
MOV AH,1
MOVE IT TO BIT 2

EDC0 6A66 2631  
MOV AH,1
MOVE IT TO BIT 2

EDC6 6E0404 2632  
AND AH,4  
ISOLATE THAT BIT

EDBA 6AE2 2632  
OR AH,DL  
OR IN THE DRIVE NUMBER

EDBC 605000 2633  
CALL NEC_OUTPUT

EDBF 2634  
TEST FOR FORMAT COMMAND

EDF0 E84001 2649  
CALL WAIT_INT  
WAIT FOR THE INTERRUPT

EDF3 2650  
J17:  
MOTOR_OFF

EDF5 E7265 2651  
JC J21  
LOOK FOR ERROR

EDF6 E73501 2652  
CALL RESULTS  
GET THE NEC STATUS

EDF8 723F 2653  
JC J20  
LOOK FOR ERROR

EDF9 2654  
CHECK THE RESULTS RETURNED BY THE CONTROLLER

EDFA 6C0 2655  
CID  
SET THE CORRECT DIRECTION

EDFB BE4200  R 2658  
MOV SI,OFFSET NEC_STATUS  
POINT TO STATUS FIELD

EDFC 2659  
LODS NEC_STATUS  
GET STO

EDFF 26CO 2660  
AND AL,GCOND  
TEST FOR NORMAL TERMINATION

A-35
EE01 743B 2561 JZ J22 ; OPEN_OK
EE03 3C40 2562 CMP AL,040H ; TEST FOR ABNORMAL TERMINATION
EE05 7529 2563 JNZ J18 ; NOT ABNORMAL, BAD NEC
2564
2565 ;--------- ABNORMAL TERMINATION, FIND OUT WHY
2566
EE07 AC 2567 LODS NEC_STATUS ; GET STI
EE08 D0E0 2568 SAL AL,1 ; TEST FOR EOF FOUND
EE09 8404 2569 MOV AH,RECORD_NOT_FND
EE0C 7229 2570 JC J19 ; RW_FAIL
EE0E D0E0 2571 SAL AL,1 ; TEST FOR CRC_ERROR
EE10 D0E0 2572 SAL AL,1 ; TEST FOR CRC_ERROR
EE12 8410 2573 MOV AH,BAD_CRC
EE14 721C 2574 JC J19 ; RW_FAIL
EE16 D0E0 2575 SAL AL,1 ; TEST FOR DMA OVERRUN
EE18 B408 2576 MOV AH,BAD_DMA
EE1A 7216 2577 JC J19 ; RW_FAIL
EE1C D0E0 2578 SAL AL,1 ; TEST FOR RECORD NOT FOUND
EE1E D0E0 2579 SAL AL,1 ; TEST FOR RECORD NOT FOUND
EE20 B404 2580 MOV AH,RECORD_NOT_FND
EE22 720E 2581 JC J19 ; RW_FAIL
EE24 D0E0 2582 SAL AL,1 ; TEST FOR RECORD NOT FOUND
EE26 B403 2583 MOV AH,WRITE_PROTECT ; TEST FOR WRITE_PROTECT
EE28 7208 2584 JC J19 ; RW_FAIL
EE2A D0E0 2585 SAL AL,1 ; TEST MISSING ADDRESS MARK
EE2C B402 2586 MOV AH,BAD_ADDR_MARK
EE2E 7202 2587 JC J19 ; RW_FAIL
2588
2589 ;--------- NEC MUST HAVE FAILED
2590
EE30 2591 J18: ; RW-NEC_FAIL
EE30 B420 2592 MOV AH,BAD_NEC
EE32 2593 J19: ; RW_FAIL
EE32 0026A100 2594 R OR DISKETTE_STATUS, AH
EE36 B8701 2595 J20: CALL NPL_TRANS ; HOW MANY WERE REALLY TRANSFERRED
EE39 2596 J21: ; RW_ERR
EE39 C3 2597 RET ; RETURN TO CALLER
EE3A 2598 J22: ; RW_ERR_RES
EE3A E8E01 2599 CALL RESULTS_FLUSH THE RESULTS BUFFER
EE3D C3 2600 RET
2602
2603 ;---------- OPERATION WAS SUCCESSFUL
2604
EE3E 2605 J22: ; OPN_OK
EE3E E6F01 2606 CALL NPL_TRANS ; HOW MANY GOT MOVED
EE41 32E4 2607 XOR AH, AH ; NO ERRORS
EE43 C3 2608 RET
2609 RW_OPN_ERR
2610 ;------------------------------------------------------------------------
2611 ; NEC_OUTPUT
2612 ; THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER
2613 ; AFTER TESTING FOR CORRECT DIRECTION AND CONTROLLER READY
2614 ; THIS ROUTINE WILL TIME OUT IF THE BYTE IS NOT ACCEPTED
2615 ; WITHIN A REASONABLE AMOUNT OF TIME, SETTING THE DISKETTE STATUS
2616 ; ON COMPLETION
2617 ; INPUT
2618 ; (AH) BYTE TO BE OUTPUT
2619 ; OUTPUT
2620 ; CY = 0 SUCCESS
2621 ; CY = 1 FAILURE -- DISKETTE STATUS UPDATED
2622 ; IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL
2623 ; HIGHER THAN THE CALLER OF NEC_OUTPUT
2624 ; THIS REMOVES THE REQUIREMENT OF TESTING AFTER EVERY CALL
2625 ; OF NEC_OUTPUT
2626 ; (AL) DESTROYED
2627 ;------------------------------------------------------------------------
EE44 2628 NEC_OUTPUT PROC NEAR
EE44 E2 2629 PUSH DX ; SAVE REGISTERS
EE45 51 2630 PUSH CX
EE46 8AF03 2631 MOV DX,03F4H ; STATUS PORT
EE49 33C9 2632 XOR CX,CX ; COUNT FOR TIME OUT
EE4B 2633 J23: ;
EE4B EC 2634 INT AL,DX ; GET STATUS
EE4C 840D 2635 TEST AL,040H ; TEST DIRECTION BIT
EE4E 72C 2636 JZ J25 ; IRATION OK
EE50 EF9 2637 LOOP J25 ;
EE52 2638 J24: ; TIME_ERROR

A-36
```assembly
EE52 000E510080  R  2639  OR  DISKETTE_STATUS,TIME_OUT
EE57 59      2640  POP  CX
EE56 5A      2641  POP  DX ; SET ERROR CODE AND RESTORE REGS
EE59 5B      2642  POP  AX ; DISCARD THE RETURN ADDRESS
EE6A F9      2643  STC ; INDICATE ERROR TO CALLER
EE5B C3      2644  RET
EE65C5      2645
EE6C 33C9      2646  J25:  XOR  CX,CX ; RESET THE COUNT
EE6E 56      2647  J26:  XOR  CX,CX
EE6E EC      2648  IN  AL,DX ; GET THE STATUS
EE6F 5008      2649  TEST  AL,000H ; IS IT READY
EE61 7504      2650  JNZ  J27 ; YES, GO OUTPUT
EE66 32F9      2651  LOOP  J26 ; COUNT DOWN AND TRY AGAIN
EE65 EBED      2652  JMP  J24 ; ERROR CONDITION
EE67 5354      2653  J27:  ; OUTPUT
EE67 58C4      2654  MOV  AL,AH ; GET BYTE TO OUTPUT
EE69 BAF503      2655  MOV  DX,OFFSH ; DATA PORT
EE6C EE      2656  OUT  DX,AL ; OUTPUT THE BYTE
EE6D 59      2657  POP  CX ; RECOVER REGISTERS
EE66 5A      2658  POP  DX
EE6F C3      2659  RET ; CY = 0 FROM TEST INSTRUCTION
EE7D      2660  NECD_OUTPUT ; ENDP
EE7D 1F      2661  ;__________________________________________
EE7E 52C0      2662  ; GET_PARM PROC NEAR
EE7F 6E08      2663  ;-----------------------------------------------
EE80 5C567000      2664  ; THIS ROUTINE Fetches THE Indexed pointer FROM
EE79 1DE8      2665  ; THE DISK_BASE block pointed at BY the DATA
EE7B 8A20      2666  ; VARIABLE DISK_POINTER
EE7D 1F      2667  ; A BYTE FROM THAT TABLE IS THEN MOVED INTO AH,
EE8E 5800      2668  ; THE INDEX OF THAT BYTE being the PARM in BX
EE89 5354      2669  ; ENTRY --
EE8A 59      2670  ; BX = INDEX OF BYTE to BE FETCHED * 2
EE8B 54      2671  ; IF THE LOW BIT of BX is ON, THE BYTE is IMMEDIATELY
EE90 59      2672  ; OUTPUT to THE NEC CONTROLLER
EE91 5800      2673  ; EXIT --
EE92 5354      2674  ; AH = THAT BYTE FROM BLOCK
EE93 5800      2675  ;-----------------------------------------------
EE87 1F      2676  ; GET_PARM PROC NEAR
EE88 52C0      2677  ; SAVE SEGMENT
EE89 6E08      2678  ; SUB AX,AX ; ZERO TO AX
EE8A 5C567000      2679  ; ASSUME DS:ABSO
EE8B 53567000      2680  ; LOD S1,DISK_POINTER ; POINT TO BLOCK
EE8C 1DE8      2681  ; SHR BX,1 ; DIVIDE BX BY 2, AND SET flag FOR EXIT
EE8D 8A20      2682  ; MOV AL,BX ; GET THE WORD
EE8E 1F      2683  ; POP DS ; RESTORE SEGMENT
EE8F 5800      2684  ; ASSUME DS:DATA
EE90 72C4      2685  ; JC NECD_OUTPUT ; IF flag SET, OUTPUT TO CONTROLLER
EE91 C3      2686  ; RET ; RETURN TO CALLER
EE92 1F      2687  ;__________________________________________
EE93 52C0      2688  ; SEEK PROC NEAR
EE94 8001      2689  ;-----------------------------------------------
EE95 51      2690  ; SEEK
EE96 52C0      2691  ; THIS ROUTINE WILL move THE HEAD ON THE NAMED DRIVE
EE97 6E08      2692  ; TO THE NAMED TRACK. IF THE DRIVE HAS NOT BEEN ACCESSED
EE98 5800      2693  ; SINCE THE DRIVE RESET command has ISSUED, THE DRIVE will be
EE99 5354      2694  ; RECALIBRATED.
EE9A 5800      2695  ; INPUT
EE9B 5800      2696  ; (DL) = DRIVE to SEEK ON
EE9C 5800      2697  ; (CH) = TRACK to SEEK to
EE9D 5800      2698  ; OUTPUT
EE9E 5800      2699  ; CY = 0 SUCCESS
EE9F 5800      2700  ; CY = 1 FAILURE -- DISKETTE_STATUS SET ACCORDINGLY
EEA0 5800      2701  ; (AX) DESTROYED
EEA1 2700      2702  ;-----------------------------------------------
EEA2 52C0      2703  ; SEEK PROC NEAR
EEA3 8001      2704  ; MOV AL,1 ; ESTABLISH mask for recal test
EEA4 51      2705  ; PUSH CX ; SAVE INPUT VALUES
EEA5 6A0A      2706  ; MOV CL,DL ; GET DRIVE VALUE INTO CL
EEA6 D2C0      2707  ; ROL AL,CL ; shift it by the drive value
EEA7 5800      2708  ; POP CX ; RECOVER TRACK Value
EEA8 5800      2709  ; TEST AL,SEEK_STATUS ; TEST FOR RECAL REQUIRED
EEA9 7513      2710  ; JZ J28 ; NO_REC
EEAA 000B3E00      2711  ; OR SEEK_STATUS,AL ; TURN ON THE NO RECAL BIT IN flag
EEAB B407      2712  ; MOV AH,01H ; RECALIBRATE command
EEAC EACFF      2713  ; CALL NECD_OUTPUT
EEAD 8AE2      2714  ; MOV AH,DL
EEAE 5800      2715  ; CALL NECD_OUTPUT ; OUTPUT the DRIVE NUMBER
```
EED0 E87200 2716 CALL CHK_STAT_Z ; GET THE INTERRUPT AND SENSE INT STATUS
EEA0 7229 2717 JC J32 ; SEEK_ERROR
EFA 2718 ; ----- DRIVE IS IN SYNCH WITH CONTROLLER, SEEK TO TRACK
EFA2 J2B:
EFA2 840F 2721 MOV AH,8FH ; SEEK COMMAND TO NEC
EFA4 E800FF 2722 CALL NEC_OUTPUT ; DRIVE NUMBER
EFA7 AE2 2724 MOV AH,DL ; DRIVE NUMBER
EFA9 E900FF 2725 CALL NEC_OUTPUT ; TRACK NUMBER
EFAE BA05 2726 MOV AH,CH ; TRACK NUMBER
EFAE 9E5FF 2727 CALL NEC_OUTPUT
EEB1 E65E00 2728 CALL CHK_STAT_Z ; GET ENDING INTERRUPT AND SENSE STATUS
EEB9 E87200 2729 ; ----- WAIT FOR HEAD SETTLE
EEB9 7C 2730 PUSHF ; SAVE STATUS FLAGS
EEB9 B10000 2731 MOV BX,10 ; GET HEAD SETTLE PARAMETER
EEB9 E800FF 2732 CALL GET_PARM
EEC0 51 2733 PUSH CX ; SAVE REGISTER
EEC0 9C6002 2734 MOV CX,550 ; 1 MS LOOP
EEC1 7406 2735 JZ J31 ; TEST FOR TIME EXPIRED
EEC3 8F0E 2736 JS J3B ; DELAY FOR 1 MS
EEC5 AEC 2737 DEC AH ; DECREMENT THE COUNT
EEC7 EFB3 2738 JNP J29 ; DO IT SOME MORE
EEC9 8743 2739 J31: ; ----- MAINT FOR HEAD SETTLE
EEC9 59 2740 POP CX ; RECOVER STATE
EECA 0D 2741 POPF
EECB 893FF 2742 CALL CHK_OUTPUT
EEC9 2743 ; SEEK ENOP

SEEK ENOP

SEEK ERROR

DMA_SETUP PROC NEAR
EECC 2759 ; -------------------------------
EECC 51 2760 PUSH CX ; SAVE REGISTER
EECC 6E0C 2761 OUT DMA+12,AL ; SET THE FIRST/LAST F/F
EECC E60D 2762 OUT DMA+11,AL ; OUTPUT THE MODE BYTE
EECC 6C09 2763 MOV AX,ES ; GET THE ES VALUE
EECC B104 2764 MOV CL,4 ; SHIFT COUNT
EECC D30C 2765 RL AX,CL ; ROTATE LEFT
EECC 6A08 2766 MOV CH,AL ; GET HIGHEST HIBYTE OF ES TO CH
EECC 240F 2767 AND AL,OFH ; ZERO THE LOW HIBYTE FROM SEGMENT
EECC D3C 2768 ADD AX,DX ; TEST FOR CARRY FROM ADDITION
EECC 7302 2769 JNC J33 ;
EECC F0C5 2770 INC CH ; CARRY MEANS HIGH 4 BITS MUST BE INC
EEE1 2771 J33: ; ----- DETERMINE COUNT
EEE1 50 2772 PUSH AX ; SAVE START ADDRESS
EEE2 E604 2773 OUT DMA+14,AL ; OUTPUT LOW ADDRESS
EEE4 BA04 2774 MOV AL,AX
EEE6 6E04 2775 OUT DMA+14,AL ; OUTPUT HIGH ADDRESS
EEE8 64C5 2776 MOV AL,CH ; GET HIGH 4 BITS
EEEA 240F 2777 AND AL,OFH
EEEC E661 2778 OUT 081H,AL ; OUTPUT THE HIGH 4 BITS TO PAGE REGISTER
EEF0 46 2779 PUSH AX ; SAVE COUNT VALUE
EEF0 50 2780 MOV AH,0H ; NUMBER OF SECTORS
EEF0 2AC0 2781 SUB AL,AL ; TIMES 250 INTO AX
EEF2 81E8 2782 SMRR AX,1 ; SECTORS * 125 INTO AX
EEF4 50 2783 PUSH AX
EEF5 BD0000 2784 MOV BX,6 ; GET THE BYTES/SECTOR PARAM
EEF8 E875FF 2785 CALL GET_PARM
EEF8 B4C 2786 MOV CL,AH ; USE AS SHIFT COUNT (0=128, 1=256 ETC)
EEFD 85 2787 POP AX
EEFE B3E0 2788 MOV AX,CL ; MULTIPLY BY CORRECT AMOUNT
EEFO 46 2789 DEC AX ; -1 FOR DMA VALUE
EF01 50 2790 PUSH AX ; SAVE COUNT VALUE

A-38
EF02 E605 2793 OUT DMA+5,AL ; LOW BYTE OF COUNT
EF04 8AC4 2794 MOV AL,AH
EF06 E605 2795 OUT DMA+5,AL ; HIGH BYTE OF COUNT
EF08 59 2796 POP CX ; RECOVER COUNT VALUE
EF09 58 2797 POP AX ; RECOVER ADDRESS VALUE
EF0A 03C1 2798 ADD AX,CX ; ADD, TEST FOR 64K OVERFLOW
EF0C 59 2799 POP CX ; RECOVER REGISTER
EF0D B002 2800 MOV AL,Z ; MODE FOR 0237
EF0F E60A 2801 OUT DMA+10,AL ; INITIALIZE THE DISKETTE CHANNEL
EF11 C3 2802 RET ; RETURN TO CALLER, CFL SET BY ABOVE IF ERROR

EF12 2803 DNA_SETUP END

EF12 EBIE00 3201 CALL WAIT_INT ; WAIT FOR THE INTERRUPT
EF15 7214 3202 JC J34 ; IF ERROR, RETURN IT
EF17 840B 3203 MOV AL,60H ; SENSE INTERRUPT STATUS COMMAND
EF19 E020FF 3204 CALL NEC_OUTPUT
EF1C E04C00 3205 CALL RESULTS ; READ IN THE RESULTS
EF1F 720A 3206 JC J34 ; CHK2_RETURN
EF21 A04C00 3207 MOV AL,0C0H ; SET THE FIRST STATUS BYTE
EF24 2460 3208 AND AL,00H ; ISOLATE THE BITS
EF26 3C60 3209 CMP AL,00H ; TEST FOR CORRECT VALUE
EF28 7402 320A JZ J35 ; IF ERROR, GO MARK IT
EF2A F0 320B CLC ; GOOD RETURN
EF2B 320C J34:
EF30 C3 320D RET ; RETURN TO CALLER
EF33 00E040040 320E OR DISKETTE_STATUS,BAD_SEEK
EF34 F9 320F STC ; ERROR RETURN CODE
EF36 C3 3210 RET

EF37 2803 CHK_STAT_2 END

EF38 F8 2810 STI ; TURN ON Interrupts, JUST IN CASE
EF3A 53 2811 PUSH BX
EF3B 51 2812 PUSH CX ; SAVE REGISTERS
EF3D B002 2813 MOV BL,2 ; CLEAR THE COUNTERS
EF3F 3C9F 2814 XOR CX,CX ; FOR 2 SECOND WAIT
EF3A 3215 J36:
EF3B F6063E0880 3216 TEST SEEK_STATUS,INT_FLAG ; TEST FOR INTERRUPT OCCURRING
EF3F 750C 3217 JNZ J37
EF41 2EF7 3218 LOOP J36 ; COUNT DOWN WHILE WAITING
EF43 FC9B 3219 DEC BL ; SECOND LEVEL COUNTER
EF45 75F3 321A JNZ J36
EF47 0004610000 R 321B OR DISKETTE_STATUS,TIME_OUT ; NOTHING HAPPENED
EF4C F9 321C STC ; ERROR RETURN
EF4F 9C 321D PUSHF ; SAVE CURRENT CARRY
EF50 80263E087F R 321E AND SEEK_STATUS,NOT_INT_FLAG ; TURN OFF INTERRUPT FLAG
EF53 90 321F POPF ; RECOVER CARRY
EF54 59 3220 POP CX
EF55 50 3221 POP BX ; RECOVER REGISTERS
EF56 C3 3222 RET ; GOOD RETURN CODE COMES FROM TEST INST
EF59 2804 WAIT_INT END

A-39
LOC OBJ
LINE SOURCE

2070 ;------------------------------------------
2071 ; DISK_INT
2072 ; THIS ROUTINE HANDLES THE DISKETTE INTERRUPT
2073 ; INPUT
2074 ; NONE
2075 ; OUTPUT
2076 ; THE INTERRUPT FLAG IS SET IS SEEK_STATUS
2077 ;------------------------------------------

EF57
2078 DISK_INT PROC FAR

EF57 FB
2079 STI ; RE ENABLE INTERRUPTS

EF58 1E
2080 PUSH DS

EF59 50
2081 PUSH AX

EF5A B80000 R
2082 MOV AX,DATA

EF5D 8EE0
2083 MOVD DS,AX

EF5F 80008000 R
2084 OR SEEK_STATUS.INT_FLAG

EF60 D200
2085 MOVD AL,20H ; END OF INTERRUPT MARKER

EF61 E620
2086 OUT 20H,AL ; INTERRUPT CONTROL PORT

EF62 58
2087 POP AX

EF69 1F
2088 POP DS ; RECOVER SYSTEM

EF6A CF
2089 INT 21H ; RETURN FROM INTERRUPT

EF6B
2090 DISK_INT ENDP

EF6C
2091 ; RESULTS

EF70 82
2092 ; RESULTS

EF71 B207
2093 MOV DI,OFFSET NEC_STATUS ; POINTER TO DATA AREA

EF72 51
2094 PUSH CX ; SAVE COUNTER

EF73 52
2095 PUSH DX

EF74 53
2096 PUSH BX

EF75 B307
2097 MOVD BL,7 ; MAX STATUS BYTES

EF76
2098 ----- WAIT FOR REQUEST FOR MASTER

EF77 5A:
2099 J38: ; INPUT_LOOP

EF78 33C9
2100 XOR CX,CX ; COUNTER

EF79 BAF043
2101 MOVD DX,03F0H ; STATUS PORT

EF7A 9000
2102 MOV AL,DX ; GET STATUS

EF7B 570C
2103 TEST AL,050H ; MASTER READY

EF7C E2F9
2104 LOOP J39 ; WAIT_MASTER

EF7D 8000410000 R
2105 OR DISKETTE_STATUS,TIME_OUT

EF7E 5A:
2106 J40: ; RESULTS_ERROR

EF7F 69
2107 STC

EF80 5B
2108 POP BX

EF81 5A
2109 POP DX

EF82 59
2110 POP CX

EF83 C3
2111 RET

EF84
2112 ----- TEST THE DIRECTION BIT

EF85 EC
2113 J4A: IN AL,DX ; GET STATUS REG AGAIN

EF86 A040
2114 TEST AL,040H ; TEST DIRECTION BIT

EF87 7507
2115 JNZ J42 ; OK TO READ STATUS

EF88
2116 J41: ; NEC_FAIL

EF89 8000410020 R
2117 OR DISKETTE_STATUS.BAD_NEG

EF90 EBDF
2118 JMP J40 ; RESULTS_ERROR

EF91
2119 ----- READ IN THE STATUS

EF92 67
2120 J42: ; INPUT_STAT

EF93 42
2121 INC DX ; POINT AT DATA PORT

EF94 EC
2122 IN AL,DX ; GET THE DATA

EF95 8805
2123 MOVD 1011,AL ; STORE THE BYTE

EF96 47
2124 INC DI ; INCREMENT THE POINTER

EF97 B90A00
2125 MOVD CX,10 ; LOOP TO KILL TIME FOR NEC

A-40
EF9E 02FE 2946 J43: LOOP J43
EFA0 0A 2947 DEC DX ; POINT AT STATUS PORT
EFA1 EC 2948 IN AL,DX ; GET STATUS
EFA2 A010 2949 TEST AL,010H ; TEST FOR NEC STILL BUSY
EFA4 7406 2950 JZ J44 ; RESULTS DONE
EFA6 FDCB 2951 DEC BL ; DECIPHER THE STATUS COUNTER
EFA8 75CA 2952 JNZ J38 ; GO BACK FOR MORE
EFAA EBE3 2953 JMP J41 ; CHIP HAS FAILED
2954
2955 ;----- RESULT OPERATION IS DONE
2956
EFAC 2957 J44:
EFAC 58 2958 POP BX
EFA5 5A 2959 POP DX
EFAE 59 2960 POP CX ; RECOVER REGISTERS
EFAF C3 2961 RET ; GOOD RETURN CODE FROM TEST INST
2962
2963 ; NUM_TRANS
2964 ; THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT
2965 ; WERE ACTUALLY TRANSFERRED TO/FROM THE DISKETTE
2966 ; INPUT
2967 ; (CH) = CYLINDER OF OPERATION
2968 ; (CL) = START SECTOR OF OPERATION
2969 ; OUTPUT
2970 ; (AL) = NUMBER ACTUALLY TRANSFERRED
2971 ; NO OTHER REGISTERS MODIFIED
2972
EF00 2973 NUM_TRANS PROC NEAR
EF00 A0500 R 2974 MDV AL,HEC_STATUS+3 ; GET CYLINDER END ED UP ON
EF03 3A5 2975 CMP AL,CH ; SAME AS WE STARTED
EF05 A04700 R 2976 MDV AL,HEC_STATUS+5 ; GET ENDING SECTOR
EF08 740A 2977 JZ J45 ; IF ON SAME CYL, THEN NO ADJUST
EF0B E0000 2978 MDV BX,0
EF0D E600F 2979 CALL GET_PARM ; GET EDT VALUE
EF0F 0A49 2980 MDV AL,AH ; INTO AL
EF12 FEDC 2981 INC AL ; USE EDT+1 FOR CALCULATION.
EF14 2AC1 2982 J45: SUB AL,CL ; SUBTRACT START FROM END
EF16 C3 2983 RET
2984
2985 NUM_TRANS ENDP
2986
2987 ;-----------------------------
2988 ; DISK_BASE
2989 ; THIS IS THE SET OF PARAMETERS REQUIRED FOR
2990 ; DISKETTE OPERATION. THEY ARE POINTED AT BY THE
2991 ; DATA VARIABLE DISK_POINTER. TO MODIFY THE PARAMETERS,
2992 ; BUILD ANOTHER PARAMETER BLOCK AND POINT AT IT
2993
2994
2995 ;-----------------------------
2996 ; DISK_BASE LABEL BYTE
2997 EFC7 CF 2996 DB 11001111 ; SRT=C. H0 UNLOAD=FF - 1ST SPECIFY BYTE
2998 EFC8 02 2997 DB 2 ; HD LOAD=I, MODE=DMA - 2ND SPECIFY BYTE
2999 EFC9 28 2998 DB MOTOR_WAIT ; WAIT AFTER OPEN TIL MOTOR OFF
3000 EFCB 02 2999 DB 2 ; 512 BYTES/SECTOR
3001 EFCB DB 8 ; EOT (LAST SECTOR ON TRACK)
3002 EFC2 2A 3001 DB 02AH ; GAP LENGTH
3003 EFC4 FF 3002 DB 0FFH ; DL
3004 EFC6 50 3003 DB 050H ; GAP LENGTH FOR FORMAT
3005 EFCF F6 3004 DB 0F6H ; FILL BYTE FOR FORMAT
3006 EFD0 19 3005 DB 25 ; HEAD SETTLE TIME (MILLISECONDS)
3007 EFD1 04 3006 DB 4 ; MOTOR START TIME (1/8 SECONDS)
LOC OBJ

LINE SOURCE

3007 1--- INT 17  ------------------------------
3008 1PRINTER_ID
3009 1 THIS ROUTINE PROVIDES COMMUNICATION WITH THE PRINTER
3010 1 (AH)=0 PRINT THE CHARACTER IN (AL)
3011 1 ON RETURN, AH=1 IF CHARACTER CANT NOT BE PRINTED (TIME OUT)
3012 1 OTHER Bits SET AS ON NORMAL STATUS CALL
3013 1 (AH)=1 INITIALIZE THE PRINTER PORT
3014 1 RETURNS WITH (AH) SET WITH PRINTER STATUS
3015 1 (AH)=2 READ THE PRINTER STATUS INTO (AH)
3016 1
3017 1 | 7 6 5 4 3 2 1 0 |
3018 1 _ TIME OUT
3019 1 _ 1 = UNUSED
3020 1 _ 1 = 1/0 ERROR
3021 1 _ 1 = SELECTED
3022 1 _ 1 = OUT OF PAPER
3023 1 _ 1 = ACKNOWLEDGE
3024 1 _ 1 = BUSY
3025 1
3026 1 (DX) = PRINTER TO BE USED (0,1,2) CORRESPONDING TO ACTUAL VALUES
3027 1 IN PRINTER_BASE AREA
3028 1 DATA AREA PRINTER_BASE CONTAINS THE BASE ADDRESS OF THE PRINTER CARD(S)
3029 1 AVAILABLE (LOCATED AT BEGINNING OF DATA SEGMENT, 408H ABSOLUTE, 3 WORDS)
3030 1 :REGISTERS AH IS MODIFIED
3031 1 ALL OTHERS UNCHANGED
3032 1
3033 1 ASSUME CS:CODE,DS:DATA
3034 1
3035 1-------------------------------------------------------------------------
3036 1
3037 1 EFD2 1 PRINTER_ID 1 PROC 1 FAR
3038 1 EFD2 FB 3039 1 STI 1 ; INTERRUPTS BACK ON
3040 1 EFD2 IE 3035 1 PUSH DS 1 ; SAVE SEGMENT
3041 1 EFD2 52 3036 1 PUSH DX
3042 1 EFD2 56 3037 1 PUSH SI
3043 1 EFD2 50 3038 1 PUSH CX
3044 1 EFD2 53 3039 1 PUSH BX
3045 1 EFD8 BE4000 R 3040 1 MOV SI,DATA
3046 1 EFD8 B0DE 3041 1 MOV DS,SI 1 ; ESTABLISH PRINTER SEGMENT
3047 1 EFD8 BD02 3042 1 MOV SI,DX 1 ; GET PRINT REF
3048 1 EFD8 D016 3043 1 SHL SI,1 1 ; WORD OFFSET INTO TABLE
3049 1 EFD1 00940000 R 3044 1 MOV DX,PRINTER_BASE+SI 1 ; GET BASE ADDRESS FOR PRINTER CARD
3050 1 EFES 00D2 3045 1 OR DX,DX 1 ; TEST DX FOR ZERO, INDICATING NO PRINTER
3051 1 EFES 740C 3046 1 JB BJ 1 ; RETURN
3052 1 EFES 0A64 3047 1 OR AH,AH 1 ; TEST FOR (AH)=0
3053 1 EFES 740E 3048 1 JB BJ 1 ; PRINT_AL
3054 1 EFED FECC 3049 1 DEC AH 1 ; TEST FOR (AH)=1
3055 1 EFEE 7442 3050 1 JZ BA 1 ; INIT_PRT
3056 1 EFF1 FECC 3051 1 DEC AH 1 ; TEST FOR (AH)=2
3057 1 EFF3 742A 3052 1 JB BS 1 ; PRINTER STATUS
3058 1 EFF5 3053 1 BI: 1 ; RETURN
3059 1 EFF5 SB 3054 1 POP BX
3060 1 EFF6 59 3055 1 POP CX
3061 1 EFF7 SE 3056 1 POP SI 1 ; RECOVER REGISTERS
3062 1 EFF8 5A 3057 1 POP DX 1 ; RECOVER REGISTERS
3063 1 EFF9 1F 3058 1 POP DS
3064 1 EFFA CF 3059 1 IRET
3065 1
3066 1 1------ 1 PRINT THE CHARACTER IN (AL) 1
3067 1
3068 1 EFFF 3063 1 D2: 1
3069 1 EFFF 50 3064 1 PUSH AX 1 ; SAVE VALUE TO PRINT
3070 1 EFFF B30A 3065 1 MOV BL,IO 1 ; TIME OUT VALUE
3071 1 EFFE 33C9 3066 1 XOR CX,CX 1 ; ESTABLISH SHIFT COUNT
3072 1 FF00 EE 3067 1 OUT DX,AL 1 ; OUTPUT CHAR TO PORT
3073 1 FF01 42 3068 1 INC DX 1 ; POINT TO STATUS PORT
3074 1 FF02 3069 1 BS: 1 ; WAIT_BUSY
3075 1 FF02 EC 3070 1 IN AL,DX 1 ; GET STATUS
3076 1 FF03 8E00 3071 1 MOV AH,AL 1 ; STATUS TO AH ALSO
3077 1 FF05 A600 3072 1 TEST AL,80H 1 ; IS THE PRINTER CURRENTLY BUSY
3078 1 FF07 750E 3073 1 JNZ B4 1 ; OUT_STROBE
3079 1 FF09 EFAF 3074 1 LOOP B3 1 ; DECREMENT COUNT ON TIME OUT
3080 1 FF0B FECB 3075 1 DEC BL
3081 1 FF0D 75F3 3076 1 JNZ B3 1 ; WAIT FOR NOT BUSY
3082 1 FF0F 80C01 3077 1 OR AH,1 1 ; SET ERROR FLAG
3083 1 FF12 80E4F9 3078 1 AND AH,0F9H 1 ; TURN OFF THE OTHER BITS
3084 1 FF15 EB14 3079 1 JMP SHORT B7 1 ; RETURN WITH ERROR FLAG SET
3085 1 FF17 3080 B4: 1 ; OUT_STROBE
3086 1 FF17 600D 3081 1 MOV AL,0DH 1 ; SET THE STROBE HIGH
3087 1 FF19 42 3082 1 INC DX 1 ; STROBE IS BIT 0 OF PORT C OF 0255

A-42
FOIA EE
3083 OUT DX,AL
FOIB 800C
3084 MOV AL,0CH ; SET THE STROBE LOW
FOID EE
3085 OUT DX,AL
F0EE 50
3086 POP AX ; RECOVER THE OUTPUT CHAR
3087
3088 i-------- PRINTER STATUS
3089
F0IF 50
3090 B5: PUSH AX ; SAVE AL REG
3091
F0IF 50
3092 F020 6B940000 R
3093 MOV DX,printer_base; SET THE STROBE ON
F024 42
3094 INC DX
F028 EC
3095 IN AL,DX ; GET PRINTER STATUS
F02B 6A4E88
3096 MOV AH,AL
F02B 5A
3097 POP AX ; RECOVER AL REG
F02D AE05
3098 MOV AH,Al
F028 80E4F8
3099 AND AH,0FSH ; TURN OFF UNUSED BITS
F028 87:
3100 STATUS_SET
F028 SA
3101 POP DX
F02E 80F448
3102 MOV AX,46H
F031 E8C2.
3103 JMP 81 ; RETURN FROM ROUTINE
F033
3104 i-------- INITIALIZE THE PRINTER PORT
F035
3107 PUSH AX ; SAVE AL REG
F034 86:
3108 ADD DX,2 ; POINT TO OUTPUT PORT
F037 50
3109 MOV AL,0 ; SET INIT LINE LOW
F039 EE
3110 OUT DX,Al
F03A B8E803
3111 MOV AX,1000
F03D 89:
3113 DEC AX ; LOOP FOR RESET TO TAKE
F03D 48
3114 JNZ B' ; INIT_LOOP
F040 8FDB
3115 MOV AL,0CH ; NO INTERRUPTS, NON AUTO LF, INIT HIGH
F042 EE
3116 OUT DX,AL
F043 E8DB
3117 JMP B6 ; PRINTER_STATUS_1
F044 8A:
3120 i--- VIDEO_IO
3121 i THESE ROUTINES PROVIDE THE CRT INTERFACE
3122 i THE FOLLOWING FUNCTIONS ARE PROVIDED:
3123 i (AH)=0 SET MODE (AL) CONTAINS MODE VALUE
3124 i (AH)=1 0 40X25 BW (POWER ON DEFAULT)
3125 i (AH)=2 40X25 COLOR
3126 i (AH)=3 80X25 BW
3127 i (AH)=480X25 COLOR
3128 i (AH)=5 320X200 COLOR
3129 i (AH)=6 640X200 BW
3130 i CRT MODE = 7 80X25 BW CARD (USED INTERNAL TO VIDEO ONLY)
3131 i *** VIDE BW MODES OPERATE SAME AS COLOR MODES, BUT COLOR
3132 i A-43
3133 i BURST IS NOT ENABLED
3134 i (AH)=1 SET CURSOR TYPE
3135 i (AH)=2 CURSOR POSITION
3136 i (AH)=3 READ CURSOR POSITION
3137 i (AH)=4 READ LIGHT PEN POSITION
3138 i (AH)=5 SELECT ACTIVE DISPLAY PAGE (VALID ONLY FOR ALPHA MODES)
3139 i (AH)=6 NEW PAGE VALUE (0-7 FOR MODES 0&1, 0-3 FOR MODES 2&3)
(AH)=6  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

(AH)=7  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
(BH) = ATTRIBUTE TO BE USED ON BLANK LINE

CHARACTER HANDLING ROUTINES

(AH)=8  READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION
(BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)

ON EXIT:
(AL) = CHAR READ

(AH)=9  WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION
(BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)

(cx) = COUNT OF CHARACTERS TO WRITE

BL = ATTRIBUTE OF CHARACTER (ALPHA/CHAR (GRAPHICS)

SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.

(AH)=10 WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION
(BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)

(cx) = COUNT OF CHARACTERS TO WRITE

(AL) = CHAR 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 CHAR,
THE USER MUST INITIALIZE THE POINTER AT INTERRUPT IFH
LOCATION 00H/02H TO POINT TO THE 1K BYTE TABLE CONTAINING

FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION FACTOR
CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID RESULTS ONLY
FOR CHARACTERS CONTAINED ON THE SAME ROW. CONTINUATION TO
SUCCEEDING LINES WILL NOT PRODUCE CORRECTLY.

GRAPHICS INTERFACE

(AH)=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(1)/RED(2)/YELLOW(3)
1 = CYAN(1)/MAGENTA(2)/WHITE(3)

IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET FOR
PALETTE COLOR 0 INDICATES THE BORDER COLOR
TO BE USED (VALUES 0-31, WHERE 16-31 SELECT THE
HIGH INTENSITY BACKGROUND SET.

(AH)=12 WRITE DOT

(dx) = ROW 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

(AH)=13 READ DOT

(dx) = ROW NUMBER

(cx) = COLUMN NUMBER

(al) RETURNS THE DOT READ

ASCII TELETYPING ROUTINE FOR OUTPUT

(AH)=14 WRITE TELETYPE

(al) = CHAR TO WRITE

(bh) = FOREGROUND COLOR IN GRAPHICS MODE

(bh) = DISPLAY PAGE IN ALPHA MODE

NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET

A-44
3232  \( (AH) = 15 \) CURRENT VIDEO STATE
3233  \( (AL) \) = CURRENT VIDEO STATE (see AH=O for explanation)
3234  \( (AH) \) = NUMBER OF CHARACTER COLUMNS ON SCREEN
3235  \( (BH) \) = CURRENT ACTIVE DISPLAY PAGE
3236  \( (AL) \) :: NUMBER OF CHARACTER COLUMNS ON SCREEN
3237  \( (AH) \) :: CURRENT ACTIVE DISPLAY PAGE
3238  CS, SS, DS, ES, BX, CX, DX PRESERVED DURING CALL
3239  ALL OTHERS DESTROYED

3240  ----------------------------------------
3241  
3242  ;---------------------------------------
3243  
3244  ;---------------------------------------
3245  
3246  ;---------------------------------------
3247  
3248  ;---------------------------------------
3249  
3250  ;---------------------------------------
3251  
3252  ;---------------------------------------
3253  
3254  ;---------------------------------------
3255  
3256  ;---------------------------------------
3257  
3258  ;---------------------------------------
3259  
3260  ;---------------------------------------
3261  
3262  ;---------------------------------------
3263  
3264  ;---------------------------------------
3265  
3266  ;---------------------------------------
3267  
3268  ;---------------------------------------
3269  
3270  ;---------------------------------------
3271  
3272  ;---------------------------------------
3273  
3274  ;---------------------------------------
3275  
3276  ;---------------------------------------
3277  
3278  ;---------------------------------------
3279  
3280  ;---------------------------------------
3281  
3282  ;---------------------------------------
3283  
3284  ;---------------------------------------
3285  
3286  ;---------------------------------------
3287  
3288  ;---------------------------------------
3289  
3290  ;---------------------------------------
3291  
3292  ;---------------------------------------
3293  
3294  ;---------------------------------------
3295  
3296  ;---------------------------------------
3297  
3298  ;---------------------------------------
3299  
3300  
3301  
3302  
3303  
3304  
3305 

F044  
3306  VIDEOParms LABEL BYTE
3307  ;---- INIT_TABLE
LOC OBJ | LINE | SOURCE
--- | --- | ---
F04A 30282DD0A1F0619 | 3308 | DB 30H,2AH,2DH,0AH,1FH,6,19H ; SET UP FOR 40X25
F04B 1C02070607 | 3309 | DB 1CH,2,7,6,7
F04B 00000000 | 3310 | DB 0,0,0,0
0010 | 3311 | M4 EQU $-VIDEO_PARS
F04B 71505A01F0619 | 3312 | DB 71H,50H,5AH,0AH,1FH,6,19H ; SET UP FOR 60X25
F04B 1C02070607 | 3313 | DB 1CH,2,7,6,7
F04C 00000000 | 3314 | DB 0,0,0,0
F04A 30282DD0F0664 | 3315 | DB 30H,2AH,2DH,0AH,1FH,6,64H ; SET UP FOR GRAPHICS
F04B 7000010607 | 3316 | DB 70H,2,1,6,7
F04D 00000000 | 3317 | DB 0,0,0,0
F04B 61505201F0619 | 3318 | DB 61H,50H,52H,0FH,6,19H ; SET UP FOR 80X25 BW CARD
F04D 19020DDC0C | 3319 | DB 19H,2,0DH,0DH,0CH
F04C 00000000 | 3320 | DB 0,0,0,0
F04C | 3321 | M5 LABEL WORD ; TABLE OF REGEN LENGTHS
F04C 00000000 | 3322 | DB 40,40,00,00,40,40,00,00
F04C 20205052B285505 | 3323 | DB 40,40,00,00,40,40,00,00
F04D 1440 | 3324 | M6 LABEL BYTE ; TABLE OF MODE SETS
F04B 02C20D292A2E1E29 | 3325 | DB 2CH,20H,2DH,2AH,2AH,2AH,2AH,2AH
F04C | 3326 | SET_MODE PROC NEAR
F04C B4A003 | 3327 | MOV DX,03H4H ; ADDRESS OF COLOR CARD
F04F B300 | 3328 | MOV DL,0 ; MODE SET FOR COLOR CARD
F101 83FF30 | 3329 | CMP DL,30H ; IS BW CARD INSTALLED
F106 7507 | 3330 | JNE #9B ; IS NOT COLOR
F106 B007 | 3331 | MOV AL,7 ; INDICATE BW CARD MODE
F10B A6H03 | 3332 | MOV DX,03B4H ; ADDRESS OF BW CARD
F10B FECC | 3333 | INC DL ; MODE SET FOR BW CARD
F10A 0490 | 3334 | MOV M8, DL ; SAVE MODE IN AL
F110 249000 | 3335 | MOV M9, AL ; SAVE IN GLOBAL VARIABLE
F112 09166300 | 3336 | MOV ADDR_6045,DX ; SAVE ADDRESS OF BASE
F116 1000 | 3337 | PUSH DS ; SAVE POINTED TO DATA SEGMENT
F117 50 | 3338 | PUSH AX ; SAVE MODE
F11A 550D | 3339 | PUSH DX ; SAVE OUTPUT PORT VALUE
F119 03C20C4 | 3340 | ADD DX,4 ; POINT TO CONTROL REGISTER
F11C 6A3C | 3341 | MOV AX,AL ; GET MODE SET FOR CARD
F11E EE | 3342 | OUT DX,AL ; RESET VIDEO
F11F 5A | 3343 | POP AX ; BACK TO BASE REGISTER
F120 2BC0 | 3344 | SUB AL,AX ; SET UP FOR AB5 SEGMENT
F122 80ED | 3345 | MOV DS,AX ; ESTABLISH VECTOR TABLE ADDRESSING
F123 9000 | 3346 | ASSUME DS:AB50
F124 C51E7400 | 3347 | LDS DX,PARM_PTR ; GET Pointer TO VIDEO PARMs
F128 5B | 3348 | POP AX ; RECOVER PARMs
F129 B90000 | 3349 | ASSUME DS:CODE
F12C 80F0C0 | 3350 | MOV CX,94H ; LENGTH OF EACH ROW OF TABLE
F12F 80FC02 | 3351 | CMP AH,2 ; DETERMINE WHICH ONE TO USE
F132 7C10 | 3352 | JC #M ; MODE IS 0 OR 1
F131 03D9 | 3353 | ADD BX,CX ; MOVE TO NEXT ROW OF INIT TABLE
F133 00FC04 | 3354 | MOV CX,4H ; DETERMINE WHICH ONE TO USE
F135 7209 | 3355 | JC #M ; MODE IS 2 OR 3
F136 03D0 | 3356 | ADD BX,CX ; MOVE TO GRAPICS ROW OF INIT_TABLE
F13A 80FC07 | 3357 | CMP AH,7
F13D 7202 | 3358 | JC #M ; MODE IS 4.5, OR 6
F13F 03D9 | 3359 | ADD BX,CX ; MOVE TO BW CARD ROW OF INIT_TABLE
F142 32E4 | 3360 | XOR AH,4H ; AH WILL SERVE AS REGISTER NUMBER DURING LOOP
F141 32E4 | 3361 | XOR AH,4H ; AH WILL SERVE AS REGISTER NUMBER DURING LOOP
F145 | 3362 | M10 : ; INIT LOOP
F146 84A4 | 3363 | MOV AL,4H ; GET 6045 REGISTER NUMBER
A-46
DETERMINE FIC4 AL600D

FIC7 42
FIC6 FE04
FIC5 8A07
FIC4 E8C4
FIC3 44
FIC2 FF3
FIC1 58
FIC0 1F
FIC7 35FF
FIC6 09B4E00
FIC5 C660620000
FIC4 B0D20
FIC3 80FC04
FIC2 720C
FIC1 80FC07
FIC0 7404
FIC7 33C0
FIC6 ED04
FIC5 F6
FIC4 B0DD08
FIC3 72
FIC2 0207
FIC1 75
FIC0 3F
FIC7 20460036700
FIC6 A04900
FIC5 32E4
FIC4 080F
FIC3 01663300
FIC2 83C204
FIC1 000F
FIC0 16500
FIC7 2E8A04EF0
FIC6 32E4
FIC5 A34A00
FIC4 81E60E00
FIC3 8E0BCE4AF0
FIC2 800E6C00
FIC1 A80000
FIC0 18
FIC7 33C0
FIC6 F3
FIC5 AB

LOC OBJ | LINE SOURCE
--------|--------|---------
F146 EE | 3304   | OUT DX,AL  |
F167 42  | 3355   | INC DX    |
F148 FE04 | 3356   | INC AH    |
F144 8A07 | 3357   | MOV AL,[BX]  |
F14C EE | 3358   | OUT DX,AL  |
F14D 45  | 3359   | INC BX    |
F144 4A  | 3390   | DEC BX    |
F14F 2F3 | 3391   | LOOP M10  |
F151 58  | 3392   | POP AX    |
F152 1F  | 3393   | POP DS    |
F155 33FF| 3398   | XOR DI,DI  |
F159 C660620000 | 3400 | MOV ACTIVE_PAGE,0  |
F156 B0D20 | 3401 | MOV CX,[SI]  |
F161 80FC04 | 3402 | CMP AH,4  |
F164 720C | 3403 | JC M12  |
F166 80FC07 | 3404 | CMP AH,7  |
F169 7404 | 3405 | JE M11  |
F168 33C0 | 3406 | XOR AX,AX  |
F16D ED04 | 3407 | JMP SHORT M13  |
F16F 3308 | 3408 | M11:  |
F16F B0DD08 | 3409 | MOV CX,2048  |
F172 3410 | M12:  |
F172 0207 | 3411 | MOV AX,*'*+7*256  |
F175 3412 | M13:  |
F176 F3 | 3413 | REP STOSW  |
F176 AB | 3414 |  
F177 20460036700 | 3415 | MOV CURSOR_MODE,67H  |
F170 A04900 | 3416 | MOV AL,CRT_MODE  |
F180 32E4 | 3417 | XOR AH,AH  |
F182 080F | 3418 | MOV SI,AX  |
F184 01663300 | 3419 | MOV DX,OFFSET_6045  |
F186 83C204 | 3420 | ADD DX,4  |
F188 2E8A04EF0 | 3421 | MOV AX,OFFSET_M1  |
F190 EE | 3422 | OUT DX,AL  |
F191 A6500 | 3423 | MOV CRT_MODE_SET,AL  |
F194 2E8A04ECF0 | 3424 | MOV AL,AX  |
F199 32E4 | 3425 | XOR AH,AH  |
F19B A34A00 | 3426 | MOV CRT_COLS,AX  |
F19E 81E60E00 | 3427 | MOV AL,CS:[SI]+OFFSET M1  |
F1A2 8E0BCE4AF0 | 3428 | MOV CX,CS:[SI]  |
F1A7 800E6C00 | 3429 | MOV CRT_LEN,CX  |
F1AB B0DD00 | 3430 | MOV CX,8  |
F1AD BF5D00 | 3431 | MOV QI,OFFSET_CURSOR_POS  |
F1B0 1E | 3432 | FUSH DS  |
F1B2 07 | 3433 | POP ES  |
F1B3 33C0 | 3434 | XOR AX,AX  |
F1BB F3 | 3435 | REP STOSW  |
F1BD AB | 3436 |  
F1B7 42 | 3446 | INC DX  |
F1BB B030 | 3447 | MOV AL,3OH  |
F1BA 83E4900006 | 3448 | CMP CRT_MODE,6 |
F1BF 7502 | 3449 | JNZ M14  |
F1C1 B03F | 3450 | MOV AL,3FH  |
F1C3 EE | 3451 | OUT DX,AL  |
F1C4 A66000 | 3452 | MOV CRT_PALETTE,AL  |

A-47
LOC OBJ LINE SOURCE

FIC7 3458 VIDEO_RETURN:
FIC7 SF 3459 POP DI
FIC8 SE 3460 POP SI
FIC9 SB 3461 POP DX
FICA 3462 M15: ; VIDEO_RETURN_C
FICA 3463 POP CX
FICA 3464 POP DX
FICC IF 3465 POP DS
FICO 67 3466 POP ES ; RECOVER SEGMENTS
FICE CF 3467 IRET ; ALL DONE
3468 SET_MODE ENDP
3469 ;---------------------------------------------------------------------
3470 ; SET_CTYPE
3471 ; THIS ROUTINE SETS THE CURSOR VALUE
3472 ; INPUT
3473 ; (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE
3474 ; OUTPUT
3475 ; NONE
3476 ;---------------------------------------------------------------------
FICF 3477 SET_CTYPE PROC NEAR
FICF 840A 3478 MOV AH,10 ; 6845 REGISTER FOR CURSOR SET
FICF 80060000 3479 MOV CURSOR_MODE,CX ; SAVE IN DATA AREA
FICO 500200 3480 CALL M16 ; OUTPUT CX REG
FICO EBED 3481 JMP VIDEO_RETURN
3482
3483 ;----- THIS ROUTINE OUTPUTS THE CX REGISTER TO THE 6845 REGS NAMED IN AH
3484 3485
FICF 3485 M16:
FICF 8D166300 3486 MOV DX,ADDR_6845 ; ADDRESS REGISTER
FIDE 8AC4 3487 MOV AL,AH ; GET VALUE
FIE0 EE 3488 OUT DX,AL ; REGISTER SET
FIE1 42 3489 INC DX ; DATA REGISTER
FIEE 5AC5 3490 MOV AL,CH ; DATA
FIE4 EE 3491 OUT DX,AL
FIE5 4A 3492 DEC DX
FIE6 8AC4 3493 MOV AL,AH
FIE8 FECC 3494 INC AL ; POINT TO OTHER DATA REGISTER
FIEA EE 3495 OUT DX,AL ; SET FOR SECOND REGISTER
FIEB 42 3496 INC DX
FIEC 8AC1 3497 MOV AL,CL ; SECOND DATA VALUE
FIEE EE 3498 OUT DX,AL
FIEF C3 3499 RET ; ALL DONE
3500 SET_CTYPE ENDP
3501 ;---------------------------------------------------------------------
3502 ; SET_CPOS
3503 ; THIS ROUTINE SETS THE CURRENT CURSOR POSITION TO THE
3504 ; NEW X-Y VALUES PASSED
3505 ; INPUT
3506 ; DX - ROW,COLUMN OF NEW CURSOR
3507 ; BH - DISPLAY PAGE OF CURSOR
3508 ; OUTPUT
3509 ; CURSOR IS SET AT 6845 IF DISPLAY PAGE IS CURRENT DISPLAY
3510 ;---------------------------------------------------------------------
FIF0 3511 SET_CPOS PROC NEAR
FIF0 8AC4 3512 MOV CL,BH
FIF2 32ED 3513 XOR CH,CH ; ESTABLISH LOOP COUNT
FIF4 D1E1 3514 SAL CX,1 ; WORD OFFSET
FIF6 60F1 3515 MOV SI,CX ; USE INDEX REGISTER
FIF8 89945000 3516 MOV [SI+OFFSET CURSOR_POSN1,DX] ; SAVE THE POINTER
FIFC 303E6200 3517 CMP ACTIVE_PAGE,BH
F200 7505 3518 JNZ M17 ; SET_CPOS_RETURN
F202 60E2 3519 MOV AX,DX ; SET ROW/COLUMN TO AX
F204 EB0200 3520 CALL M18 ; CURSOR_SET
F207 3521 M17: ; SET_CPOS_RETURN
F207 EBBE 3522 JMP VIDEO_RETURN
3523 SET_CPOS ENDP
3524
3525 ;----- SET CURSOR POSITION, AX HAS ROW/COLUMN FOR CURSOR
3526 3527
F209 3527 M18 PROC NEAR
F209 E07F00 3528 CALL POSITION ; DETERMINE LOCATION IN REGEN BUFFER
F20C 88C8 3529 MOV CX,AX
F20E 03004E00 3530 ADD CX,CRT_START ; ADD IN THE START ADDRESS FOR THIS PAGE
F212 D1F9 3531 SAR CX,1 ; DIVIDE BY 2 FOR CHAR ONLY COUNT
F214 B40E 3532 MOV AH,14 ; REGISTER NUMBER FOR CURSOR

A-48
```
F21A 3546 READ_CURSOR ENDP
F21B 3547 MOV BL,BH ;  READ_CURSOR
F21C 3548 XOR BH,BH
F21D 3549 SAL BX,1  ;  WORD OFFSET
F21E 3550 MOV DX,(BX+OFFSET CURSOR_POSN)
F21F 3551 MOV CX,CURSOR_MODE
F220 3552 POP DI
F221 3553 POP SI
F222 3554 POP BX
F223 3555 POP AX  ;  DISCARD SAVED CX AND DX
F224 3556 POP AX
F225 3557 POP DS
F226 3558 POP ES
F227 3559 JEBT
F228 3560 READ_CURSOR ENDP
F229 3561 ;  ACT_DISP_PAGE
F22A 3562 ;  THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING
F22B 3563 ;  THE FULL USE OF THE RAM SET ASIDE FOR THE VIDEO ATTACHMENT
F22C 3564 ;  INPUT
F22D 3565 ;  AL HAS THE NEW ACTIVE DISPLAY PAGE
F22E 3566 ;  OUTPUT
F22F 3567 ;  THE 6845 IS RESET TO DISPLAY THAT PAGE
F230 3568 ;  --------------------------------------------
F231 3569
F232 3570 ACT_DISP_PAGE ENDP
F233 3571 MOV ACTIVE_PAGE,AL ;  SAVE ACTIVE PAGE VALUE
F234 3572 MOV CX,CRT_LEN ;  GET SAVED LENGTH OF REGEN BUFFER
F235 3573 CEN ;  CONVERT AL TO WORD
F236 3574 PUSH AX ;  SAVE PAGE VALUE
F237 3575 MUL CX ;  DISPLAY PAGE TIMES REGEN LENGTH
F238 3576 MOV CRT_START,AX ;  SAVE START ADDRESS FOR LATER REQUIREMENTS
F239 3577 MOV CX,AX ;  START ADDRESS TO CX
F23A 3578 SAR CX,1 ;  DIVIDE BY 2 FOR 6845 HANDLING
F23B 3579 MOV AH,12 ;  6845 REGISTER FOR START ADDRESS
F23C 3580 CALL Mi6
F23D 3581 POP BX ;  RECOVER PAGE VALUE
F23E 3582 SAL BX,1 ;  *2 FOR WORD OFFSET
F23F 3583 MOV AX,(BX+OFFSET CURSOR_POSN) ;  GET CURSOR FOR THIS PAGE
F240 EBD0FF 3584 CALL Mi6 ;  SET THE CURSOR POSITION
F241 E973FF 3585 JMP VIDEO_RETURN
F242 3586 ACT_DISP_PAGE ENDP
F243 3587 ;  --------------------------------------------
F244 3588 ;  SET COLOR
F245 3589 ;  THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN COLOR,
F246 3590 ;  AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION GRAPHICS
F247 3591 ;  INPUT
F248 3592 ;  (BH) HAS COLOR ID
F249 3593 ;  IF BH=0, THE BACKGROUND COLOR VALUE IS SET
F24A 3594 ;  FROM THE LOW BITS OF BH (0-31)
F24B 3595 ;  IF BH=1, THE PALLETTE SELECTION IS MADE
F24C 3596 ;  BASED ON THE LOW BIT OF BH:
F24D 3597 ;  0 = GREEN, RED, YELLOW FOR COLORS 1,2,3
F24E 3598 ;  1 = BLUE, CYAN, MAGENTA FOR COLORS 1,2,3
F24F 3599 ;  IF (BL) HAS THE COLOR VALUE TO BE USED
F250 3599 ;  OUTPUT
F251 3600 ;  THE COLOR SELECTION IS UPDATED
F252 3601 ;  --------------------------------------------
F253 3602
F254 3603 SET_COLOR ENDP
F255 3604 MOV DX,ADDR_6845 ;  I/O PORT FOR PALLETTE
F256 3605 ADD DX,5 ;  OVERSCAN PORT
F257 3606 MOV AL,CRT_PALLETTE ;  GET THE CURRENT PALLETTE VALUE
F258 3607 OR BH,BH ;  IS THIS COLOR 0?
```
LOC OBJ  

LINE SOURCE

F260 750E  3608  JNZ  M20  ; OUTPUT COLOR 1
3609
3610 ---- HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
3611
F262 24E0  3612  AND  AL, 0EOH  ; TURN OFF LOW 5 BITS OF CURRENT
3613
F264 60631F  3614  AND  BL, 01FH  ; TURN OFF HIGH 3 BITS OF INPUT VALUE
3615
F267 04C3  3616  OR  AL, BL  ; PUT VALUE INTO REGISTER
3617
F269  3618  M19:  ; OUTPUT THE PALETTE
F269 EE  3619  OUT  DX, AL  ; output color selection to 3d9 port
F26A A26600  R  3620  MOV  CRT_PALETTE, AL  ; SAVE THE COLOR VALUE
F26D E957FF  3621  JMP  VIDEO_RETURN
3622
F270  3623  M20:  
F270 24DF  3624  AND  AL, 0DFH  ; TURN OFF PALETTE SELECT BIT
F272 00ED  3625  SHR  BL, 1  ; TEST THE LOW ORDER BIT OF BL
F274 73F3  3626  JNC  M19  ; ALREADY DONE
F276 02C0  3627  OR  AL, 20H  ; TURN ON PALETTE SELECT BIT
F278 BBFF  3628  JMP  M19  ; GO DO IT
3629
F27A  3630  VIDEO_STATE  PROC  NEAR
F27A 8A2CA000  R  3631  MOV  AH, BYTE PTR CRT_COLS  ; GET NUMBER OF COLUMNS
F27A A04900  R  3632  MOV  AL, CRT_MODE  ; CURRENT MODE
F281 8A3E6000  R  3633  MOV  BH, ACTIVE_PAGE  ; GET CURRENT ACTIVE PAGE
F285 5F  3634  POP  DI  ; RECOVER REGISTERS
F286 5E  3635  POP  SI  ;
F287 59  3636  POP  CX  ; DISCARD SAVED BX
F288 E93FF  3637  JMP  M15  ; RETURN TO CALLER
3638
F28A  3639  VIDEO_STATE  ENDP

F28B  3640  POSITION  PROC  NEAR
F28B 53  3641  PUSH  BX  ; SAVE REGISTER
F28C 00D0  3642  MOV  BX, AX
F28E 8AC4  3643  MOV  AL, AH  ; ROWS TO AL
F290 F064A00  R  3644  MUL  BYTE PTR CRT_COLS  ; DETERMINE BIT8E TO ROW
F294 33FF  3645  XOR  BL, BH  ;
F296 03C3  3646  ADD  AX, BX  ; ADD IN COLUMN VALUE
F298 01E0  3647  SAL  AX, 1  ; *= 2 FOR ATTRIBUTE BYTES
F29A 5B  3648  POP  BX
F29B C3  3649  RET
3650
F29D  3651  POSITION  ENDP

F29E  3652  SCROLL_UP  PROC  NEAR
F29E 8A00  3653  MOV  BL, AL  ; SAVE LINE COUNT IN BL
F29F 8AFC04  3654  CMP  AH, 4  ; TEST FOR GRAPHICS MODE

A-50
LOC OBJ  
LINE SOURCE

F2A1 7200 3664 JC N1 : HANDLE SEPARATELY
F2A3 00FC07 3665 CMP AH,7 : TEST FOR DW CARD
F2A6 74E3 3666 JE N1
F2AB 9E301 3667 JMP GRAPhICS_UP
F2AB 3668 N1:  : UP_CONTINUE
F2A2 53 3669 PUSH BX  : SAVE FILL ATTRIBUTE IN BH
F2AC 8DC1 3670 MOV AX,CX  : UPPER LEFT POSITION
F2AE E5900 3671 CALL SCROLL_POSITION  : DO SETUP FOR SCROLL
F2B1 7435 3672 JE N7  : BLANK_FIELD
F2B5 03F8 3673 ADD SI,AX  : FROM ADDRESS
F2BA 8A86 3674 MOV AH,DH  : 8 ROWS IN BLOCK
F2BA AE3 3675 SUB AH,BL  : 8 ROWS TO BE MOVED
F2B9 3676 N2:  : ROW_LOOP
F2B9 E6700 3677 CALL N10  : MOVE ONE ROW
F2B8 03F5 3678 ADD DI,BP  : POINT TO NEXT LINE IN BLOCK
F2C0 FECC 3679 DEC AH  : COUNT OF LINES TO MOVE
F2C2 75F5 3680 JNZ N4  : CLEAR_LOOP
F2C4 3681 N5:  : CLEAR_ENTRY
F2C4 50 3682 POP AX  : RECOVER ATTRIBUTE IN AH
F2C5 0020 3683 MOV AL,1  : FILL WITH BLANKS
F2C7 3684 JMP HANDLE
F2C7 E6700 3685 CALL N11  : CLEAR THE ROW
F2CA 03F0 3686 ADD DI,BP  : POINT TO NEXT LINE
F2CC FECD 3687 DEC BL  : COUNTER OF LINES TO SCROLL
F2C7 75F9 3688 JNZ N4  : CLEAR_LOOP
F2D0 3689 NS:  : SCROLL_END
F2D0 B0000 R 3690 MOV AX,DATA  : GET LOCATION
F2D3 8E88 3691 MOV DS,AX
F2D5 836490007 R 3692 JMP CRT_MODE,7  : IS THIS THE BLACK AND WHITE CARD
F2DA 7407 3693 JE N6  : IF SO, SKIP THE MODE RESET
F2DC A06500 3694 MOV AL,CRT_MODE_SET  : GET THE VALUE OF THE MODE SET
F2DF BADD03 3695 MOV DX,0300H  : ALWAYS SET COLOR CARD PORT
F2EE EE 3696 OUT DX,AL  : VIDEO_MODE
F2E3 3697 N6:  : VIDEO_RESET_HERE
F2E3 E9E1FE 3698 JMP VIDEO_RETURN  : BLANK_FIELD
F2E6 3699 N7:  : BLANK_FIELD
F2EB 6A0E 369A MOV BL,DH  : GET ROW COUNT
F2EB EBOA 369B JMP N3  : GO CLEAR THAT AREA
F2EB 7712 369C SCROLL_UP : END
F2EB 7724 369D
F2EB 7725 369E ;----- HANDLE COMMON SCROLL SET UP HERE
F2EB 7726
F2EA 369F SCROLL_POSITION : PROC NEAR
F2EA 836440002 R 3700 CMP CRT_MODE,2  : TEST FOR SPECIAL CASE HERE
F2EF 7219 3701 JB N9  : HAVE TO HANDLE BOX25 SEPARATELY
F2F1 836440003 R 3702 CMP CRT_MODE,3
F2F6 7712 3703 JA N9
F2F6 7724 3704
F2F6 7725 3705 ;----- BOX25 COLOR CARD SCROLL
F2F6 7734
F2F8 51 3706 PUSH DX
F2F9 BAD403 3707 MOV DX,3AH  : GUARANTEED TO BE COLOR CARD HERE
F2FC 50 3708 PUSh AX
F2FD 3709 NB:  : WAIT_DISP_ENABLE
F2FD EC 3710 IN AL,DX  : GET PORT
F2FE A0A8 3711 TEST AL,0  : WAIT FOR VERTICAL RETRACE
F300 74FB 3712 JZ N8  : WAIT_DISP_ENABLE
F302 B025 3713 MOV AL,SH
F304 BADD03 3714 MOV DX,0300H
F307 EE 3715 OUT DX,AL  : TURN OFF VIDEO
F306 50 3716 POP AX  : DURING VERTICAL RETRACE
F309 5A 3717 POP DX
F30A E7EFF 3718 CALL POSITION  : CONVERT TO REGN POINTER
F30D 0364E000 R 3719 ADD AX,CRT_START  : OFFSET OF ACTIVE PAGE
F311 00F8 371A MOV DI,AX  : TO ADDRESS FOR SCROLL
F313 00F0 371B MOV SI,AX  : FROM ADDRESS FOR SCROLL
F319 2D01 371C SUB DX,AX  : DX = #ROWS, COLS IN BLOCK
F317 F6CA 371D INC DH
F319 FEC2 371E DL  : INCREMENT FOR 0 ORIGIN
F31B 32ED 371F XOR CH,CH  : SET HIGH BYTE OF COUNT TO ZERO
F31D 0B26A000 R 3720 MOV BP,CRT_COLS  : GET NUMBER OF COLUMNS IN DISPLAY
F31E 03ED 3721 ADD BP,BP  : TIMES 2 FOR ATTRIBUTE BYTE
F321 0BAC3 3722 MOV AL,BL  : GET LINE COUNT
F325 FA26A400 R 3723 PUSH PTR CRT_COLS  : DETERMINE OFFSET TO FROM ADDRESS
F329 03CO 3724 ADD AX,AX  : *2 FOR ATTRIBUTE BYTE
F326 06 3760 PUSH ES ; ESTABLISH ADDRESSING TO REGEN BUFFER
F32C 0F 3761 POP DS ; FOR BOTH PointERS
F330 80F0003762 CMP BL,0 ; 0 SCROLL MEANS BLANK FIELD
F33C 80 3763 RET ; RETURN WITH FLAGS SET

F334 SCROLL_POSITION ENDP

F336 N10 PROC NEAR
F33A 0ACA 3766 ----- MOVE_ROW
F33F 80FC04 3804 CMP AH,4 ; TEST FOR GRAPHICS
F340 80FC03 3806 MOV AH,DX ; LINE COUNT TO BL
F342 80FC07 3808 MOV AH,7 ; TEST FOR BW CARD
F344 7200 3809 JC N12 ; LOWER RIGHT CORNER OF REGION
F346 7420 3811 JZ H16 ; REGEN END LOCATION
F348 2BFD 3813 SUB SI,AX SI IS FROM ADDRESS
F34A 8A64 3815 MOV AH,DH ; GET TOTAL # ROWS
F34C 2AE3 3816 SUB AH,BL ; COUNT TO MOVE IN SCROLL
F350 N12: CONTINUE_DOWN
F355 80DC 3817 JMP GRAPHICS_DOWN
F355 80DC 3819 MOV AX,DX ; LOWER RIGHT CORNER
F355 80D0 3820 SUB SI,AX
F355 8A56 3821 INC SI
F355 2F5F 3823 JMP H13 ; MOVE ONE ROW
F358 0600 3825 MOV AL,0
F35A 0605 3827 CALL H11 ; CLEAR ONE ROW
F35C 2FDF 3829 SUB DI,DP ; GO TO NEXT ROW
F35F 8ECC 382A DEC AH
F362 75F5 382B JNZ N13
F364 N13: MOVE_ROW
F36A 50 382D POP AX ; RECOVER ATTRIBUTE IN AH
F36C B020 382F MOV AL,0
F36C 0600 3831 CALL H11 ; CLEAR ONE ROW
F36F 2FDF 3833 SUB DI,DP ; GO TO NEXT ROW
F372 8FEC 3835 DEC BL
F374 79F7 3837 JNZ N15
F376 8E57FF 3839 JMP H5 ; SCROLL_END
F379 N16: SCROLL_END
F379 8ADE 383B MOV BL,DH
The document contains assembly code for a computer program, specifically for handling attribute and character values at the current cursor position. The code includes procedures for reading and writing attribute and character values, as well as handling page retrace events. The comments in the code explain the purpose of each section and the context in which it operates. The assembly language instructions are used to manipulate data and control the flow of the program, typically involving memory access, register operations, and conditional jumps.
WRITE_AC_CURRENT

; IS THIS GRAPHICS
CMP AH, 4
JC P6

; IS THIS BW CARD
CMP AH, 7
JE P6

; WRITE_AC_CONTINUE
JMP GRAPHICS_WRITE

F3C0 0AE3
F3C0 7403
F3C0 7403
F3C0 E98101
F3C0 P6:

F3C0 50
F3C0 51
F3C0 E5OFF
F3D7 60FB
F3D9 59
F3D9 58
P7:

F3D9 58
F3D9 50
P7:

F3E0 59
F3E0 A801
F3E0 74FB
F3ED 74FF
F3ED 80FC04
F3ED 80FC07
F3ED 7403
F3ED E97E01
F3E8 50
F3E8 51
F3E8 58
F3E8 59
F3E8 58
P8:

F3E8 58
F3E8 50
P9:

F3E8 59
F3E8 58
P10:

F3E8 58
F3E8 58
F3E8 59
F3E8 58
P11:

F3E9 59
F3E9 58
P12:

F3E9 58
F3E9 58
F3E9 58
F3E9 58
P13:

F3E9 58
F3E9 58
F3E9 59
F3E9 58
P14:

F3FA 59
F3FA 58
P9:

F3FA 58
F3FA 50
P8:

F3FA 50
F3FA 59
P7:

F3FB 58
F3FB 50
P6:

; INPUT
; (AH) = CURRENT CRT MODE
; (BH) = DISPLAY PAGE
; (CX) = COUNT OF CHARACTERS TO WRITE
; (AL) = CHAR TO WRITE
; (DS) = DATA SEGMENT
; (ES) = REGSEG SEGMENT

; OUTPUT
; NONE

F3F6 00FC04
F3F6 7403
F3F6 E97E01
F3F8 50
F404 0E69FF
F408 586B
F408 586B
F40C 59
F40C 58
F40C

; WRITE_LOOP
; AS MANY TIMES AS REQUESTED

; ------ WAIT FOR HORIZONTAL RETRACE

; ------ WAIT FOR HORIZONTAL RETRACE

; ------ WAIT FOR HORIZONTAL RETRACE

; ------ WAIT FOR HORIZONTAL RETRACE

; ------ WAIT FOR HORIZONTAL RETRACE

; ------ WAIT FOR HORIZONTAL RETRACE

; ------ WAIT FOR HORIZONTAL RETRACE

; ------ WAIT FOR HORIZONTAL RETRACE

; ------ WAIT FOR HORIZONTAL RETRACE

; ------ WAIT FOR HORIZONTAL RETRACE
F41C 74FB 3906 JZ P13 ; WAIT UNTIL IT IS
F41E BAC3 3907 MOV AL,BL ; RECOVER CHAR
F420 AA 3908 STOSB ; PUT THE CHAR/ATTR
F421 07 3909 INC DI ; BUMP PAST ATTRIBUTE
F422 E260 3910 LOOP P11 ; AS MANY TIMES AS REQUESTED
F424 8F40FD 3991 JMP VIDEO_RETURN
F424 8F458 3992 WRITE.C_CURRENT ENDP
3993 ---------------
3994 ; READ DOT -- WRITE DOT
3995 ; THESE ROUTINES WILL WRITE A DOT, OR READ THE
3996 ; DOT AT THE INDICATED LOCATION
3997 ; ENTRY ---
3998 ; DX = ROW (0-199) (THE ACTUAL VALUE DEPENDS ON THE MODE)
3999 ; CX = COLUMN (0-639) (THE VALUES ARE NOT RANGE CHECKED)
4000 ; AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE).
4001 ; REQ'D FOR WRITE DOT ONLY, RIGHT JUSTIFIED.
4002 ; BIT 7 OF AL = 1 INDICATES XOR THE VALUE INTO THE LOCATION
4003 ; DS = DATA SEGMENT
4004 ; ES = REGEN SEGMENT
4005 ;
4006 ; EXIT
4007 ; AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY
4008 ;---------------------------------------------------------------
4009 ; ASSUME CS:CODE,DS:DATA,ES:DATA
F427 0B3100 4010 READ_DOT PROC NEAR
F428 26BA04 4011 CALL R3 ; DETERMINE BYTE POSITION OF DOT
F429 0B2C6 4012 MOV AL,ES:[SI] ; GET THE BYTE
F42A 22C4 4013 AND AL,AX ; MASK OFF THE OTHER BITS IN THE BYTE
F42B 0210 4014 SHR AL,CL ; LEFT JUSTIFY THE VALUE
F430 8A2C 4015 MOV CL,DL ; GET NUMBER OF BITS IN RESULT
F431 320C 4016 ROL AL,CL ; RIGHT JUSTIFY THE RESULT
F432 E900FD 4017 JMP VIDEO_RETURN ; RETURN FROM VIDEO ID
F433 7500 4018 READ_DOT ENDP
F435 8D50 4020 WRITE_DOT PROC NEAR
F436 50 4021 PUSH AX ; SAVE DOT VALUE
F439 50 4022 PUSH AX ; TWICE
F43A E0100 4023 CALL R3 ; DETERMINE BYTE POSITION OF THE DOT
F43B D20E 4024 SHR AL,CL ; SHIFT TO SET UP THE BITS FOR OUTPUT
F43C 22C4 4025 AND AL,AX ; STRIP OFF THE OTHER BITS
F43D 66A0C 4026 MOV CL,ES:[SI] ; GET THE CURRENT BYTE
F43E 50 4027 POP AX ; RECOVER XOR FLAG
F440 6C30 4028 TEST BL,00H ; IS IT ON
F441 7500 4029 JNZ R2 ; YES, XOR THE DOT
F442 F604 4030 NOT AH ; SET THE MASK TO REMOVE THE INDICATED BITS
F443 22CC 4031 AND CL,AN ; AL AM
F444 8A1C 4032 OR CL,AL ; OR IN THE NEW VALUE OF THOSE BITS
F445 0B11 4033 R1: ; FINISH_DOT
F446 26B004 4034 MOV ES:[SI],AL ; RESTORE THE BYTE IN MEMORY
F447 50 4035 POP AX
F448 E970FD 4036 JMP VIDEO_RETURN ; RETURN FROM VIDEO ID
F449 7500 4037 R2: ; XOR_DOT
F44A 32C1 4038 XOR AL,CL ; EXCLUSIVE OR THE DOTS
F44B 8B50 4039 JNP R1 ; FINISH UP THE WRITING
4040 WRITE_DOT ENDP
4041 -------------------------------
4042 ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION OF THE
4043 ; INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.
4044 ; ENTRY ---
4045 ; DX = ROW VALUE (0-199)
4046 ; CX = COLUMN VALUE (0-639)
4047 ; EXIT ---
4048 ; SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST
4049 ; AH = MASK TO STRIP OFF THE BITS OF INTEREST
4050 ; CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH
4051 ; DH = # BITS IN RESULT
4052 -------------------------------
F450 0B2D 4053 R3 PROC NEAR
F450 53 4054 PUSH BX ; SAVE BX DURING OPERATION
F45C 50 4055 PUSH AX ; WILL SAVE AL DURING OPERATION
4056 0B2E 4057 ------ DETERMINE 1ST BYTE IN INDICATED ROW BY MULTIPLYING ROW VALUE BY 40
4058 ------ ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW
4059 0B2F 4060 MOV AL,40
F45F 52 4061 PUSH DX ; SAVE ROW VALUE
LOC OBJ

F450 00E2FE 4062 AND DL.0FEH ; STRIP OFF ODD/EVEN BIT
F453 F6E2 4063 MUL DL ; AX HAS ADDRESS OF 1ST BYTE OF INDICATED ROW
F465 5A 4064 POP DX ; RECOVER IT
F466 F6C01 4065 TEST DL.1 ; TEST FOR EVEN/ODD
F469 7403 4066 JZ R4 ; JUMP IF EVEN ROW
F46B 05020 4067 ADD AX.2000H ; OFFSET TO LOCATION OF ODD ROWS
F46E 4068 R4: ; EVEN_ROW
F46F 88F0 4069 MOV SI,AX ; MOVE POINTER TO SI
F470 5B 4070 POP AX ; RECOVER AL VALUE
F471 00D1 4071 MOV DX,DX ; COLUMN VALUE TO DX
F472 4072 ;-----------------------------------
F473 4073 i----- DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
F474 4074 | SET UP THE REGISTERS ACCORDING TO THE MODE
F475 4075 | CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES)
F476 4076 | CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M)
F477 4077 | BL = MASK TO SELECT BITS FROM POINTED BYTE (80H/0FH FOR H/M)
F478 4078 | BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M)
F479 4079 |
F47A 4080 |
F47B 4081 |
F47C 4082 MOV BX.2OCH
F47D 4083 MOV CX.302H ; SET PARMS FOR MED RES
F47E 4084 CMP CRT_MODE.6
F47F 4085 JC R5 ; HANDLE IF MED ARE
F480 4086 MOV BX.180H
F481 4087 MOV CX.703H ; SET PARMS FOR HIGH RES
F482 4088 |
F483 4089 i----- DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
F484 4090 AND CH,DL ; ADDRESS OF PEL WITHIN BYTE TO CH
F485 4091 |
F486 4092 |
F487 4093 |
F488 4094 SHR DX,CL ; SHIFIT BY CORRECT AMOUNT
F489 4095 ADD SI,DX ; INCREMENT THE POINTER
F48A 4096 MOV DH,BH ; GET THE # OF BITS IN RESULT TO DH
F48B 4097 |
F48C 4098 i----- MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
F48D 4099 |
F48E 409A | SUB CL,CL ; ZERO INTO STORAGE LOCATION
F48F 409B |
F490 409C |
F491 409D |
F492 409E |
F493 409F |
F494 40A0 ADD CL,CH ; ADD IN THE BIT OFFSET VALUE
F495 40A1 DEC BH ; LOOP CONTROL
F496 40A2 MOV AH,DL ; SET MASK TO AH
F497 40A3 |
F498 40A4 SHR AH,CL ; MOVE THE MASK TO CORRECT LOCATION
F499 40A5 PDP BX ; RECOVER REG
F49A 40A6 RET ; RETURN WITH EVERYTHING SET UP
F49B 40A7 |
F49C 40A8 |
F49D 40A9 |
F49E 40A0 |
F49F 40A1 |
F4A0 40A2 |
F4A1 40A3 |
F4A2 40A4 |
F4A3 40A5 |
F4A4 40A6 |
F4A5 40A7 |
F4A6 40A8 |
F4A7 40A9 |
F4A8 40AA |
F4A9 40AB |
F4AA 40AC |
F4AB 40AD |
F4AC 40AE |
F4AD 40AF |
F4AE 40B0 |
F4AF 40B1 |
F4B0 40B2 |
F4B1 40B3 |
F4B2 40B4 |
F4B3 40B5 |
F4B4 40B6 |
F4B5 40B7 |
F4B6 40B8 |
F4B7 40B9 |
F4B8 40BA |
F4B9 40BB |
F4BA 40BC |
F4BB 40BD |
F4BC 40BE |
F4BD 40BF |
F4BE 40C0 |
F4BF 40C1 |
F4C0 40C2 |
F4C1 40C3 |
F4C2 40C4 |
F4C3 40C5 |
F4C4 40C6 |
F4C5 40C7 |
F4C6 40C8 |
F4C7 40C9 |
F4C8 40CA |
F4C9 40CB |
F4CA 40CC |
F4CB 40CD |
F4CC 40CE |
F4CD 40CF |
F4CE 40D0 |
F4CF 40D1 |
F4D0 40D2 |
F4D1 40D3 |
F4D2 40D4 |
F4D3 40D5 |
F4D4 40D6 |
F4D5 40D7 |
F4D6 40D8 |
F4D7 40D9 |
F4D8 40DA |
F4D9 40DB |
F4DA 40DC |
F4DB 40DD |
F4DC 40DE |
F4DD 40DF |
F4DE 40E0 |
F4DF 40E1 |
F4E0 40E2 |
F4E1 40E3 |
F4E2 40E4 |
F4E3 40E5 |
F4E4 40E6 |
F4E5 40E7 |
F4E6 40E8 |
F4E7 40E9 |
F4E8 40EA |
F4E9 40EB |
F4EA 40EC |
F4EB 40ED |
F4EC 40EE |
F4ED 40EF |
F4EE 40F0 |
F4EF 40F1 |
F4F0 40F2 |
F4F1 40F3 |
F4F2 40F4 |
F4F3 40F5 |
F4F4 40F6 |
F4F5 40F7 |
F4F6 40F8 |
F4F7 40F9 |
F4F8 40FA |
F4F9 40FB |
F4FA 40FC |
F4FB 40FD |
F4FC 40FE |
F4FD 40FF |
F4FE |
F4FF |

-------------------------------------------------------------------
F49E 4125 GRAPHICS_UP PROC NEAR
F49F 4126 MOV BL,AL ; SAVE LINE COUNT IN BL
F4A0 4127 MOV AX,CX ; GET UPPER LEFT POSITION INTO AX REG
F4A1 4128 |
F4A2 4129 i----- USE CHARACTER SUBROUTINE FOR POSITIONING
F4A3 412A |
F4A4 412B i----- ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
F4A5 412C |
F4A6 412D |
F4A7 412E |
F4A8 412F |
F4A9 4130 |
F4AA 4131 |
F4AB 4132 |
F4AC 4133 |
F4AD 4134 |
F4AE 4135 |
F4AF 4136 i----- DETERMINE SIZE OF WINDOW

A-56
LOOP THROUGH, MOVING ONE ROW

FA47 0001 4137 SUB DX.CH
FA49 81020101 4138 ADD DX,101H ; ADJUST VALUES
FA4D 0006 4139 SAL DH,1 ; MULTIPLY # ROWS BY 4 SINCE 8 VERT DOTS/CHAR
FA4F 0006 4140 SAL DH,1 ; AND EVEN/ODD ROWS

FA48 ; ------ DETERMINE CRT MODE
FA4B 8014900006 R 4141 CMP CRT_MODE,6 ; TEST FOR MEDIUM RES
FA46 7304 4142 JNC R7 ; FIND_SOURCE
FA46 4143

FA48 ; ------ MEDIUM RES UP
FA4B 0002 4144 SAL DL,1 ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
FA4A 8107 4145 SAL DI,1 ; OFFSET #2 SINCE 2 BYTES/CHAR

FA49 ; ------ DETERMINE THE SOURCE ADDRESS IN THE BUFFER
FA4C 0012 4146 R7: ; FIND_SOURCE
FA4C 06 4147 PUSH ES ; GET SEGMENTS BOTH POINTING TO REGEN
FA4D 1F 4148 POP DS
FA4E 2A0D 4149 SUB CH.CH ; ZERO TO HIGH OF COUNT REG
FA4C 0003 4150 SAL BL,1 ; MULTIPLY NUMBER OF LINES BY 4
FA4C 0003 4151 SAL BL,1

FA4C 740D 4152 JZ R11 ; IF ZERO, THEN BLANK ENTIRE FIELD
FA4C 8A05 4153 MOV AL,80 ; GET NUMBER OF LINES IN AL
FA4C 8A00 4154 MOV AH,00 ; #0 BYTES/ROW
FA4C F66 4155 MUL AH ; DETERMINE OFFSET TO SOURCE
FA4C 0007 4156 MOV SI,DI ; SET UP SOURCE
FA4C 3F0 4157 ADD SI,AX ; ADD IN OFFSET TO IT
FA4D 8A06 4158 MOV AH,DX ; NUMBER OF ROWS IN FIELD
FA4D 0A0E 4159 MOV DX,SI ; DETERMINE NUMBER TO MOVE
FA4D 2A05 4160 SUB AH,DX ; RETURNED

FA4D ; ------ LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
FA4D 0004 4161 RB: ; ROW_LOOP
FA4D E80000 4162 CALL R17 ; MOVE ONE ROW
FA4D B1EBD01F 4163 SUB SI,2000H-00 ; MOVE TO NEXT ROW
FA4D B1EF01F 4164 SUB DI,2000H-00
FA4D F80F 4165 DEC AH ; NUMBER OF ROWS TO MOVE
FA4D 7F1 4166 JNZ R8 ; CONTINUE TILL ALL MOVED
FA4D 7F1 4167 ; ------ FILL IN THE VACATED LINE(S)
FA4E 0012 4168 R9: ; CLEAR_ENTRY
FA4E 8A07 4169 MOV AL,BH ; ATTRIBUTE TO FILL WITH
FA4E 0010 4170 R10: ; CLEAR THAT ROW
FA4E E80000 4171 CALL R18 ; CLEAR THAT ROW
FA4E B1EBD01F 4172 SUB SI,2000H-00 ; POINT TO NEXT LINE
FA4E B1EF01F 4173 SUB DI,2000H-00
FA4E 000F 4174 DEC BL ; NUMBER OF LINES TO FILL
FA4E 7F5F 4175 JNZ R10 ; CLEAR_LOOP
FA4F 0904FC 4176 JMP VIDEO_RETURN ; EVERYTHING DONE
FA4F 000B 4177
FA4F 8010 4178 R11: ; BLANK_FIELD
FA4F 800E 4179 MOV BL,0DH ; SET BLANK COUNT TO EVERYTHING IN FIELD
FA4F 8E0C 4180 JMP R9 ; CLEAR THE FIELD
FA4F 8E0C 4181 GRAPHICS_UP ENDP
FA4F 8E0C 4182
FA4F 8E0C 4183 ; SCROLL DOWN
FA4F 8E0C 4184 ; THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT
FA4F 8E0C 4185 ; ENTRY --
FA4F 8E0C 4186 ; CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
FA4F 8E0C 4187 ; DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
FA4F 8E0C 4188 ; BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
FA4F 8E0C 4189 ; BL = # LINES TO SCROLL (ALWAYS BLANK THE ENTIRE FIELD)
FA4F 8E0C 418A ; DS = DATA SEGMENT
FA4F 8E0C 418B ; ES = REGEN SEGMENT
FA4F 8E0C 418C ; EXIT --
FA4F 8E0C 418D ; NOTHING, THE SCREEN IS SCROLLED
FA4F 8E0C 418E
FA4F 8E0C 418F ;-------------------------------
FA4F 8E0C 4190
FA4F 8E0C 4191 ;-------------------------------
FA4F 8E0C 4192
FA4F 8E0C 4193 ;-------------------------------
FA4F 8E0C 4194
FA4F 8E0C 4195 ;-------------------------------
FA4F 8E0C 4196
FA4F 8E0C 4197 ;-------------------------------
FA4F 8E0C 4198
FA4F 8E0C 4199 ;-------------------------------
FA4F 8E0C 419A
FA4F 8E0C 419B
FA4F 8E0C 419C
FA4F 8E0C 419D
FA4F 8E0C 419E
FA4F 8E0C 419F
FA4F 8E0C 4200
FA4F 8E0C 4201
FA4F 8E0C 4202
FA4F 8E0C 4203
FA4F 8E0C 4204 ;-------------------------------
FA4F 8E0C 4205 ;-------------------------------
FA4F 8E0C 4206 ;-------------------------------
FA4F 8E0C 4207 ;-------------------------------
FA4F 8E0C 4208
FA4F 8E0C 4209 ;-------------------------------
FA4F 8E0C 420A
LOC OBJ       LINE       SOURCE

F4FF 80F0 4213  MOV DI,AX ; SAVE RESULT AS DESTINATION ADDRESS
F4F4 4214
F4F5 4215 ;------ DETERMINE SIZE OF WINDOW
F501 20D1 4216
F50B 8036490006 4217  SUB DX,CX
F50F 80C01010 4218  ADD CX,DL
F507 00E6 4219  SAL DH,1 ; MULTIPLY # ROWS BY 4 SINCE 8 VERT DOTS/CHAR
F509 00E6 4220  SAL DH,1 ; AND EVEN/ODD ROWS
F502 4221
F501 4222 ;------ DETERMINE CRT MODE
F504 4223
F500 4224  CMP CRT_MODE,6 ; TEST FOR MEDIUM RES
F510 7305 4225  JNC R12 ; FIND_SOURCE_DOWN
F514 4226
F508 4227 ;------ MEDIUM RES DOWN
F512 00E2 4228  SAL DL,1 ; 8 COLUMNS = 2, SINCE 2 BYTES/CHAR (OFFSET OK)
F514 01E7 4229  SAL DL,1 ; OFFSET "2 SINCE 2 BYTES/CHAR
F516 47 4230  INC DI ; POINT TO LAST BYTE
F51C 4231
F517 4232 ;------ DETERMINE THE SOURCE ADDRESS IN THE BUFFER
F517 06 4233  PUSH ES ; BOTH SEGMENTS TO REGEN
F51B 1F 4234  POP DS
F519 2AED 4235  SUB CH,CH ; ZERO TO HIGH OF COUNT REG
F518 81C70000 4236  ADD DI,240 ; POINT TO LAST ROW OF PIXELS
F51F 00E3 4237  SAL BL,1 ; MULTIPLY NUMBER OF LINES BY 4
F521 00E3 4238  SAL BL,1
F523 74E2 4239  JZ R16 ; IF ZERO, THEN BLANK ENTIRE FIELD
F525 8A33 4240  MOV AL,BL ; GET NUMBER OF LINES IN AL
F527 B450 4241  MOV AH,60 ; 60 BYTES/ROW
F529 F644 4242  MOV AL,AH ; DETERMINE OFFSET TO SOURCE
F53B 08F7 4243  MOV SI,SI,DI ; SET UP SOURCE
F53B 20F0 4244  MOV DI,SI,AX ; SUBTRACT THE OFFSET
F52F 8A66 4245  MOV AH,DI ; NUMBER OF ROWS IN FIELD
F531 2A33 4246  SUB SI,AX ; DETERMINE NUMBER TO MOVE
F538 4247
F533 4248 ;------ LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
F533 002100 4249  MOV SI,2000H ; BOTH SEGMENTS TO REGEN
F536 81EE5020 4250  CALL R17 ; MOVE ONE ROW
F53A 01E5020 4251  SUB SI,2000H+60 ; MOVE TO NEXT ROW
F53E FECC 4252  INC SI,2000H+60 ; MOVE ONE ROW
F540 75F1 4253  DEC AH ; NUMBER OF ROWS TO MOVE
F545 01E7 4254  DEC SI,DI ; CONTINUE TILL ALL MOVED
F546 4255
F542 4256 ;------ FILL IN THE VACATED LINE(S)
F542 00C7 4257  MOV BL,DLH ; ATTRIBUTE TO FILL WITH
F544 4258
F544 4259 ;------ CLEAR_LOOP_DOWN
F544 402900 4260  CALL R18 ; CLEAR A ROW
F547 81EBF020 4261  SUB DI,2000H+60 ; POINT TO NEXT LINE
F548 FEFB 4262  DEC DI ; NUMBER OF LINES TO FILL
F54D 75F5 4263  MOV SI,R15 ; CLEAR_LOOP_DOWN
F54F FC 4264  CLI ; RESET THE DIRECTION FLAG
F550 E974FC 4265  JMP VIDEO_RETURN ; EVERYTHING DONE
F551 4266
F551 4267 ;------ BLANK_FIELD_DOWN
F551 00DE 4268  MOV BL,DLOH ; SET BLANK COUNT TO EVERYTHING IN FIELD
F555 EBEB 4269  JMP R14 ; CLEAR THE FIELD
F557 4270
F557 4271 GRAPHICS_DOWN ENDP
F558 A2D 4272
F559 4273 ;------ ROUTINE TO MOVE ONE ROW OF INFORMATION
F55A 44 4274
F55B F3 4275  REP MOVSB ; MOVE THE EVEN FIELD
F55C 44
F55D 5F 4276  POP SI
F55E 5E 4277
F55F 40C60020 4278  ADD SI,2000H ; POINT TO THE ODD FIELD
F563 81C70020 4279  ADD DI,2000H
F566 56 4280  PUSH SI
F568 57 4281  PUSH DI ; SAVE THE POINTERS
F569 8ACA 4282  MOV CL,DL ; COUNT BACK

A-58
GRAPHICS_WRITE PROC NEAR

; THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
; POSITION ON THE SCREEN.
;
; ENTRY --

AL = CHARACTER TO WRITE
BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR
:

IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN BUFFER
: (0 IS USED FOR THE BACKGROUND COLOR)

CX = NUMBER OF CHAR'S TO WRITE
DS = DATA SEGMENT
ES = REGEN SEGMENT

EXIT --

NOTHING IS RETURNED

(--)

ASSUME CS:CODE.DS:DATA,ES:DATA

GRAPHICS_WRITE PROC NEAR

F501 B400
; ZERO TO HIGH OF CODE POINT
F503 50
; SAVE CODE POINT VALUE
F504
determine_position_in_regen_buffer_to_put_code_points
F504 E85501
CALL 526
; FIND LOCATION IN REGEN BUFFER
F507 80F8
; REGEN POINTER IN DI

------ DETERMINE REGION TO GET CODE POINTS FROM
F509 50
; RECOVER CODE POINT
F50A SC80
; IS IT IN SECOND HALF
F50C 7306
; YES

image_is_in_first_half, contained_in_rom

------ IMAGE IS IN FIRST HALF, CONTAINED IN ROM
F50E B6E0
; OFFSET CRT_CHAP_GEN-OFFSET OF IMAGES
F501 0E
; SAVE SEGMENT ON STACK
F502 E00F
; DETERMINE_MODE

------ DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
F480 F3
REP MOVSB ; MOVE THE ODD FIELD
F56C A4
F56D SF
POP DI ; POINTERS BACK
F56F C3
RET ; RETURN TO CALLER
F579 R17 ENDP

------ CLEAR A SINGLE ROW

-----

F571 8ACA
MOV CL,DL ; NUMBER OF BYTES IN FIELD
F572 57
PUSH DI ; SAVE POINTER
F573 F3
REP STOSB ; STORE THE NEW VALUE
F574 AA
F575 SF
POP DI ; POINTER BACK
F576 81C70020
ADD DI,2000H ; POINT TO ODD FIELD
F577 57
PUSH DI
F578 8ACA
MOV CL,DL
F579 F3
REP STOSB ; FILL THE ODD FIELD
F57A AA
F57E SF
POP DI
F580 C3
RET ; RETURN TO CALLER
F581 R10 ENDP

-----

GRAPHICS_READ PROC NEAR

; THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT
; POSITION ON THE SCREEN BY MATCHING THE DOTS ON THE SCREEN TO THE
; CHARACTER GENERATOR CODE POINTS
;

ENTRY --

NOTHING IS RETURNED

(--)

------

ASSUME CS:CODE.DS:DATA,ES:DATA

GRAPHICS_READ PROC NEAR

F501 B400
; ZERO TO HIGH OF CODE POINT
F503 50
; SAVE CODE POINT VALUE
F504
determine_position_in_regen_buffer_to_put_code_points
F504 E85501
CALL 526
; FIND LOCATION IN REGEN BUFFER
F507 80F8
; REGEN POINTER IN DI

------ DETERMINE REGION TO GET CODE POINTS FROM
F509 50
; RECOVER CODE POINT
F50A SC80
; IS IT IN SECOND HALF
F50C 7306
; YES

image_is_in_first_half, contained_in_rom

------ IMAGE IS IN FIRST HALF, CONTAINED IN ROM
F50E B6E0
; OFFSET CRT_CHAP_GEN-OFFSET OF IMAGES
F501 0E
; SAVE SEGMENT ON STACK
F502 E00F
; DETERMINE_MODE

------

GRAPHICS_WRITE PROC NEAR

; THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
; POSITION ON THE SCREEN.
;
; ENTRY --

AL = CHARACTER TO WRITE
BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR
:

IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN BUFFER
: (0 IS USED FOR THE BACKGROUND COLOR)

CX = NUMBER OF CHAR'S TO WRITE
DS = DATA SEGMENT
ES = REGEN SEGMENT

EXIT --

NOTHING IS RETURNED

(--)

ASSUME CS:CODE.DS:DATA,ES:DATA

GRAPHICS_WRITE PROC NEAR

F501 B400
; ZERO TO HIGH OF CODE POINT
F503 50
; SAVE CODE POINT VALUE
F504
determine_position_in_regen_buffer_to_put_code_points
F504 E85501
CALL 526
; FIND LOCATION IN REGEN BUFFER
F507 80F8
; REGEN POINTER IN DI

------ DETERMINE REGION TO GET CODE POINTS FROM
F509 50
; RECOVER CODE POINT
F50A SC80
; IS IT IN SECOND HALF
F50C 7306
; YES

image_is_in_first_half, contained_in_rom

------ IMAGE IS IN FIRST HALF, CONTAINED IN ROM
F50E B6E0
; OFFSET CRT_CHAP_GEN-OFFSET OF IMAGES
F501 0E
; SAVE SEGMENT ON STACK
F502 E00F
; DETERMINE_MODE

------

GRAPHICS_READ PROC NEAR

; THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT
; POSITION ON THE SCREEN BY MATCHING THE DOTS ON THE SCREEN TO THE
; CHARACTER GENERATOR CODE POINTS
;

ENTRY --

NOTHING IS RETURNED

(--)

------

ASSUE CS:CODE.DS:DATA,ES:DATA

GRAPHICS_READ PROC NEAR

F501 B400
; ZERO TO HIGH OF CODE POINT
F503 50
; SAVE CODE POINT VALUE
F504
determine_position_in_regen_buffer_to_put_code_points
F504 E85501
CALL 526
; FIND LOCATION IN REGEN BUFFER
F507 80F8
; REGEN POINTER IN DI

------ DETERMINE REGION TO GET CODE POINTS FROM
F509 50
; RECOVER CODE POINT
F50A SC80
; IS IT IN SECOND HALF
F50C 7306
; YES

image_is_in_first_half, contained_in_rom

------ IMAGE IS IN FIRST HALF, CONTAINED IN ROM
F50E B6E0
; OFFSET CRT_CHAP_GEN-OFFSET OF IMAGES
F501 0E
; SAVE SEGMENT ON STACK
F502 E00F
; DETERMINE_MODE

------
; EXTEND_CHAR
; ZERO ORIGIN FOR SECOND HALF
; DATA POINTER
; ESTABLISH VECTOR ADDRESSING
; GET OFFSET OF THE TABLE
; SEGMENT OF THE TABLE
; DATA
; RECOVER DATA SEGMENT
; TABLE SEGMENT ON STACK
; DETERMINE GRAPHICS MODE IN OPERATION
; MULTIPLY CODE POINT
; VALUE BY 8
; SI HAS OFFSET OF DESIRED CODES
; CCH RUN MODE
; NUMBER OF TIMES THROUGH LOOP
; READ BYTE FROM CODE POINTS
; SHOULD WE USE THE FUNCTION
; TO PUT CHAR IN
; STORE IN REGEN BUFFER
; READ BYTE FROM CODE POINTS
; MOVE TO NEXT ROW IN REGEN
; DONE WITH LOOP
; AX,1
; SAVE CODE POINTER
; RETURN TO
; STORE SECOND HALF
; EXCLUSIVE OR WITH CURRENT
; STORE THE CODE POINT
; AGAIN FOR ODD FIELD
; XOR AL,ES:[ID]+2000H-11 ; STORE SECOND HALF
; MOVE TO NEXT ROW IN REGEN
; ADD DX,79
; DCT DH
; JNZ 54
; XOR DI
; TO POINT TO CHAR POSITION
; MORE CHARS TO WRITE
; JMP VIDEO_RETURN
; STORE
; XOR AL,ES:[ID]
; STORE THE CODE POINT
; XOR AL,ES:[ID]+2000H-11
; BACK TO MAINSTREAM
; MED_RES_WRITE
; MED_RESWRITE
; DIRIM COLOR BIT
; OFFSET=2 TIMES 2 BYTES/CHAR
; EXPAND BL TO FULL WORD/CHAR
; MED_CHAR
; SAVE REGEN POINTER
; SAVE THE CODE POINTER
; NUMBER OF LOOPS
; GET CODE POINT
; CALL 521
; STORE SECOND BYTE
; GET CODE POINT
; STORE FIRST BYTE
F609 S7 4511  PUSH DI  ; SAVE CODE POINTER
F608 B00000 4512  MOV CX,0  ; NUMBER OF BYTES TO MATCH
F608 F3 4513  REP REPE CMPSB  ; COMPARE THE 0 BYTES
F608 A6 4514  POP DI  ; RECOVER THE POINTERS
F608 SF 4515  POP SI
F609 741E 4516  JL 518  ; IF ZERO FLAG SET, THEN MATCH OCCURRED
F609 FEC0 4517  JNC AL  ; NO MATCH, MOVE ON TO NEXT
F609 83C700 4518  ADD DI,0  ; NEXT CODE POINT
F609 4A 4519  DEC DX  ; LOOP CONTROL
F609 75ED 4520  JNZ 517  ; DO ALL OF THEM
F621 4521  ;------ CHAR NOT MATCHED, MIGHT BE IN USER SUPPLIED SECOND HALF
F609 3C00 4524  CMP AL,0  ; AL<>0 IF ONLY 1ST HALF SCANNED
F609 7412 4525  JE 518  ; IF = 0, THEN ALL HAS BEEN SCANNED
F609 2B00 4526  SUB AX,AX
F609 8608 4527  MOV DS,AX  ; ESTABLISH ADDRESSING TO VECTOR
F608 A6 4528  ASSUME DS:DATA
F609 C3E7C00 4529  LES DI,EXT_PTR  ; GET POINTER
F608 A6CF0 4530  MOV AX,ES  ; SEE IF THE POINTER REALLY EXISTS
F609 0B07 4531  OR AX,DI  ; IF ALL 0, THEN DOESN'T EXIST
F609 4B04 4532  JZ 518  ; NO SENSE LOOKING
F609 B000 4533  MOV AL,128  ; ORIGIN FOR SECOND HALF
F608 0B02 4534  JMP SI6  ; GO BACK AND TRY FOR IT
F609 75ED 4535  ASSUME DS:DATA
F636 4536
F601 4537  ;------ CHARACTER IS FOUND ( AL<>0 IF NOT FOUND )
F608 83C408 4538  ADD SP,0  ; READJUST THE STACK, THROW AWAY SAVE
F608 E910FB 4539  JMP VIDEO_RETURN  ; ALL DONE
F640 4540  GRAPHICS_READ ENDP

F608 4541  ;--------------------------------------------------------------------------------
F608 4542  ; EXPAND_COLOR
F608 4543  ; THIS ROUTINE EXPANDS THE LOW 2 BITS IN BL TO
F608 4544  ; FILL THE ENTIRE BX REGISTER
F608 4545  ; ENTRY --
F608 4546  ; BL = COLOR TO BE USED ( LOW 2 BITS )
F608 4547  ; EXIT --
F608 4548  ; BX = COLOR TO BE USED ( 8 REPlications OF THE 2 COLOR BITS )
F650 4549  ;----------------------------------------------------------------------
F607 4550

F607 4551  S19 PROC NEAR
F608 4552  AND BL,3  ; ISOLATE THE COLOR BITS
F608 4553  MOV AL,BL  ; COPY TO AL
F608 4AC3 4554  PUSH CX  ; SAVE REGISTER
F608 4555  MOV CX,3  ; NUMBER OF TIMES TO DO THIS
F608 4556  S20:
F608 4557  SAL AL,1  ; LEFT SHIFT BY 2
F608 4558  SAL AL,1  ; LEFT SHIFT BY 2
F608 4559  OR BL,AL  ; ANOTHER COLOR VERSION INTO BL
F608 455A  ESF8  ; LOOP S20  ; FILL ALL OF BL
F608 455B  MOV BH,BL  ; FILL UPPER PORTION
F608 455C  S9  ; POP CX  ; REGISTER BACK
F608 455D  RET  ; ALL DONE
F654 455E  S19 ENDP
F655 455F  ;----------------------------------------------------------------------
F656 4560  ; EXPANDBYTE
F657 4561  ; THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ALL
F658 4562  ; OF THE BITS, TURNING THE 8 BITS INTO 16 BITS.
F659 4563  ; THE RESULT IS LEFT IN AX
F670 4564  ;----------------------------------------------------------------------
F66C 4565
F66C 4566  S21 PROC NEAR
F66C 4567  PUSH DX  ; SAVE REGISTERS
F66C 4568  PUSH CX  ; SAVE REGISTERS
F66C 4569  PUSH BX  ; SAVE REGISTERS
F66C 456A  BAO000 4570  MOV DX,0  ; RESULT REGISTER
F66C 4571  MOV CX,1  ; MASK REGISTER
F605 4572  S22:
F605 4573  MOV BX,AX  ; BASE INTO TEMP
F605 4574  AND DX,CX  ; USE MASK TO EXTRACT A BIT
F605 4575  OR DX,BX  ; PUT INTO RESULT REGISTER
F605 4576  SHL AX,1  ; SHIFTBASE AND MASK BY 1
F605 4577  MOV BX,AX  ; BASE TO TEMP
F605 4578  MOV DX,CX  ; EXTRACT THE SAME BIT
F605 4579  OR DX,BX  ; PUT INTO RESULT

A-62
F6E5 D1E1 4506  SHL  CX,1 ; SHIFT ONLY MASK NOW, MOVING TO NEXT BASE
F6E7 73EC 4507  JNC  S2Z ; USE MASK BIT COMING OUT TO TERMINATE
F6E9 80C2 4508  MOV  AX,DX ; RESULT TO PARM REGISTER
F6EB 50 4509  POP  BX
F6EC 59 4509  POP  CX ; RECOVER REGISTERS
F6ED 5A 4509  POP  DX
F6EE C3 4510  RET ; ALL DONE
4512 ;-----------------------------------
4514 1MED_READ_BYTE
4516 ; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN BUFFER.
4518 ; COMPARE AGAINST THE CURRENT FOREGROUND COLOR, AND PLACE
4520 ; THE CORRESPONDING ON/OFF BIT PATTERN INTO THE CURRENT
4522 ; POSITION IN THE SAVE AREA
4523 ; ENTRY --
4525 ; S1.05 = POINTER TO REGEN AREA OF INTEREST
4527 ; BX = EXPANDED FOREGROUND COLOR
4529 ; BP = POINTER TO SAVE AREA
452B ; BP IS INCREMENT AFTER SAVE
452D ;-----------------------------------
4531 S23 PROC NEAR
4533 MOV  AH,[SI] ; GET FIRST BYTE
4535 MOV  AL,[SI+1] ; GET SECOND BYTE
4537 MOV  CX,0C000H ; 2 BIT MASK TO TEST THE ENTRIES
4539 MOV  DL,0 ; RESULT REGISTER
453B MOV  CX,1 ; IS THIS SECTION BACKGROUND?
453D CLC ; CLEAR CARRY IN HOPES THAT IT IS
453F JZ  S25 ; IF ZERO, IT IS BACKGROUND
4541 STC ; WASN'T, SO SET CARRY
4543 MOV  CX,1 ; MOVE THAT BIT INTO THE RESULT
4545 MOV  DX,AX ; IS THIS SECTION BACKGROUND?
4547 MOV  CX,1 ; MOVE THE MASK TO THE RIGHT BY 2 BITS
4549 JNC  S24 ; DO IT AGAIN IF MASK DIDN'T FALL OUT
454B MOV  [BP].DL, ; STORE RESULT IN SAVE AREA
454D INC  BP ; ADJUST POINTER
454F MOV  CX,1 ; RESULT TO PARM REGISTER
4551 MOV  BX,AX ; GET CURRENT CURSOR
4553 MOV  AX,CURSOR_POSH ; CURSOR POSITION CONTAINED IN
4555 MOV  AL,CURSOR_POSH ; THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET
4557 MOV  CX,0 ; INTO THE REGEN BUFFER, ASSUMING ONE BYTE/CHAR.
4559 MOV  DL,0 ; FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST
455B MOV  CX,0 ; BE DOUBLED.
455D MOV  CX,0 ; ENTRY -- NO REGISTERS, MEMORY LOCATION CURSOR_POSH IS USED
455F ; EXIT--
4561 MOV  AX,0 ; AX CONTAINS OFFSET INTO REGEN BUFFER
4563 ;-----------------------------------
4565 S26 PROC NEAR
4567 MOV  AX,CURSOR_POSH ; GET CURRENT CURSOR
4569 MOV  [EH], ; CURRENT CURSOR
456B MOV  AL,CURSOR_POSH ; CURSOR POSITION ON SCREEN IS SCROLLED UP ONE ROW.
456D MOV  AL,CURSOR_POSH ; THE NUMBER MUST BE DETERMINED FROM THE BUFFER.
456F MOV  AL,CURSOR_POSH ; COLUMN VALUE IS INCREMENTED.
4571 MOV  AL,CURSOR_POSH ; BIT VALUE INCREASED. IF THE ROW
4573 MOV  AL,CURSOR_POSH ; POSH VALUE LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW.
4575 MOV  AL,CURSOR_POSH ; FIRST COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE.
4577 MOV  AL,CURSOR_POSH ; WHEN THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE
4579 MOV  AL,CURSOR_POSH ; BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS
457B MOV  AL,CURSOR_POSH ; LINE BEFORE THE SCROLL, IN CHARACTER MODE. IN GRAPHICS MODE.
THE 0 COLOR IS USED.

ENTRY --

(C) = CURRENT CRT MODE

(AL) = CHARACTER TO BE WRITTEN

NOTE THAT BACK SPACE, CAR RET, BELL AND LINE FEED ARE HANDLED AS COMMANDS RATHER THAN AS DISPLAYABLE GRAPHICS

(BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A GRAPHICS MODE

EXIT --

ALL REGISTERS SAVED

ASSUME CS:CODE,DS:DATA

WRITE_TTY PROC NEAR

PUSH AX

MOV AH,3

INT 10H

SAVE REGISTERS

READ THE CURRENT CURSOR POSITION

MOV BH,ACTIVE_PAGE

INT 10H

DETERMINE VALUE TO FILL WITH DURING SCROLL

READ CURSOR

READ CHAR/ATTR AT CURRENT CURSOR

STORE IN BH

SCROLL-UP

SCROLL-UP

UPPER LEFT CORNER

LOWER RIGHT COLUMN

STORE IN BH

Determine value to fill with during scroll

GET THE CURRENT MODE

READ-CURSOR

READ-CURSOR

READ CHAR/ATTR AT CURRENT CURSOR

STORE IN BH

Dec.

INT 9H

SCROLL UP THE SCREEN

TTY_RETURN

RESTORE THE CHARACTER
LOC OBJ  
LINE  SOURCE  

F770 E947FA  4738  JMP  VIDEO_RETURN  ; RETURN TO CALLER  
4739  
F780  4740  U6:  ; SET-CURSOR-INC  
F780  FEC6  4741  INC  DH  ; NEXT ROW  
F782  4742  U7:  ; SET-CURSOR  
F782  0402  4743  MOV  AH,2  
F784  4744  JMP  U4  ; ESTABLISH THE NEW CURSOR  
4745  
F786  4746  ;------ BACK SPACE FOUND  
4747  
F786  607A00  4749  CMP  DL,0  ; ALREADY AT END OF LINE  
F787  74FF  4750  JE  U7  ; SET-CURSOR  
F787  F6C  4751  DEC  DL  ; NO -- JUST MOVE IT BACK  
F780  4752  JMP  U7  ; SET-CURSOR  
4753  
F791  4754  ;------ CARRIAGE RETURN FOUND  
4755  
F791  4756  U9:  
F791  B200  4757  MOV  DL,0  ; MOVE TO FIRST COLUMN  
F791  4758  JMP  U7  ; SET-CURSOR  
4759  
F791  4760  U10:  
F791  80FE10  4763  CMP  DH,24  ; BOTTOM OF SCREEN  
F796  75E0  4764  JNE  U6  ; YES, SCROLL THE SCREEN  
F796  4765  JMP  U1  ; NO, JUST SET THE CURSOR  
4766  
F796  4767  ;------ BELL FOUND  
4768  
F796  4769  U11:  
F796  B302  4770  MOV  DL,2  ; SET UP COUNT FOR BEEP  
F79C  4771  CALL  BEEP  ; SOUND THE POD BELL  
F79F  4772  JMP  U5  ; TTY_RETURN  
4773  
F79F  4774  WRITE_TTY  ENDP  
4775  ;--------------------- 
F79F  4776  ; LIGHT PEN  
F79F  4777  ; THIS ROUTINE TESTS THE LIGHT PEN SWITCH AND THE LIGHT  
F79F  4778  ; PEN TRIGGER. IF BOTH ARE SET, THE LOCATION OF THE LIGHT  
F79F  4779  ; PEN IS DETERMINED. OTHERWISE, A RETURN WITH NO INFORMATION  
F79F  4780  ; IS MADE.  
F79F  4781  ; ON EXIT:  
F79F  4782  ; (AH) = 0 IF NO LIGHT PEN INFORMATION IS AVAILABLE  
F79F  4783  ; BX,CX,DX ARE DESTROYED  
F79F  4784  ; (AH) = 1 IF LIGHT PEN IS AVAILABLE  
F79F  4785  ; (DH,DL) = RD4,COLUMN OF CURRENT LIGHT PEN POSITION  
F79F  4786  ; (CH) = RASTER POSITION  
F79F  4787  ; (BX) = BEST GUESS AT PIXEL HORIZONTAL POSITION  
4788  
4788  ;------------------------------- 
4788  ; ASSUME CS:CODE,DS:DATA  
F7A1  4789  ;------ SUBTRACT_TABLE  
F7A1  4790  VI  LABEL  BYTE  
F7A1  03005050303030304  4791  DB  3,3,5,5,3,3,3,4  ;  
F7A9  4792  READ_LIPEN  PROC  NEAR  
F7A9  4793  
F7A9  4794  ;------ WAIT FOR LIGHT PEN TO BE DEPRESSED  
4795  
F7A9  B400  4796  MOV  AH,0  ; SET NO LIGHT PEN RETURN CODE  
F7A9  B8166300  R  4797  MOV  DX,ADDR_6845  ; GET BASE ADDRESS OF 6845  
F7A9  B8166300  R  4798  ADD  DX,6  ; POINT TO STATUS REGISTER  
F782  EC  4799  IN  AL,DX  ; GET STATUS REGISTER  
F783  A804  4800  TEST  AL,4  ; TEST LIGHT PEN SWITCH  
F785  757B  4801  JNZ  V6  ; NOT SET, RETURN  
4802  
F7A9  4803  ;------ NOW TEST FOR LIGHT PEN TRIGGER  
4804  
F7A7  A802  4805  TEST  AL,2  ; TEST LIGHT PEN TRIGGER  
F7B9  747E  4806  JZ  V7  ; RETURN WITHOUT RESETTING TRIGGER  
4807  
F7B0  4A10  4808  MOV  AH,16  ; LIGHT PEN REGISTERS ON 6845  
F7B1  4809  ;------ TRIGGER HAS BEEN SET, READ THE VALUE IN  
4810  
F7B0  B410  4811  MOV  AX,16  ; LIGHT PEN REGISTERS ON 6845  
F7B1  4812  ;------ INPUT REGS POINTED TO BY AX, AND CONVERT TO ROW COLUMN IN DX  
F7B1  4813  
F7B0  B8166300  R  4814  MOV  DX,ADDR_6845  ; ADDRESS REGISTER FOR 6845  
4815  
A-65
FXC1 8AC4  4815  MOV  AL, AH  ; REGISTER TO READ
FXC3  EE   4816  OUT  DX, AL  ; SET IT UP
FXC4  42   4817  INC  DX  ; DATA REGISTER
FXC5  EC   4818  IN  AL, DX  ; GET THE VALUE
FXC6  BAE8  4819  MOV  CH, AL  ; SAVE IN CX
FXC8  4A    4820  DEC  DX  ; ADDRESS REGISTER
FXC9  FE4C  4821  INC  AM  
FXCB  BAC4  4822  MOV  AL, AH  ; SECOND DATA REGISTER
FXCD  EE   4823  OUT  DX, AL  
FXCE  42   4824  INC  DX  ; POINT TO DATA REGISTER
FXCF  EC   4825  IN  AL, DX  ; GET SECOND DATA VALUE
FXD0  BAE6  4826  MOV  AH, CH  ; AX HAS INPUT VALUE
4827
4828  ;----- AX HAS THE VALUE READ IN FROM THE 6845
4829
482A
FXD2  BA1E4900  R  4830  MOV  BL, CRT_MODE
FXD6  ZAFF  4831  SUB  BH, DH  ; MODE VALUE TO BX
FXDB  ZEA49A1F7  R  4832  MOV  BL, CS: VI[BX]  ; DETERMINE AMOUNT TO SUBTRACT
FXD2  28C3  4833  SUB  AX, DX  ; TAKE IT AWAY
FXDF  ZB046E00  R  4834  SUB  AX, CRT_START  ; CONVERT TO CORRECT PAGE ORIGIN
FXE3  7903  4835  JNS  V2  ; IF POSITIVE, DETERMINE MODE
FXE5  B80000  4836  MOV  AX, 0  ; <0 PLAYS AS 0
4837
4838
4839
FXE8  440  4840  V2:  ; DETERMINE_MODE
FXEB  B103  4841  MOV  CL, 5  ; SET BS SHIFT COUNT
FXEA  8034990004  R  4842  CMP  CRT_MODE, 4  ; DETERMINE IF GRAPHICS OR ALPHA
FXEF  722A  4843  JB  V4  ; ALPHA_PEN
FF01  863690007  R  4844  CMP  CRT_MODE, 7  ; DETERMINE MODE VALUE TO
FF06  7423  4845  JE  V4  ; ALPHA_PEN
4846
4847
4848
FF08  B228  4849  MOV  DL, 40  ; DIVISOR FOR GRAPHICS
FF0A  F6F2  4850  DIV  DL  ; DETERMINE ROW(AL) AND COLUMN(AH)
4851  AL RANGE 0-99, AH RANGE 0-39
4852
4853
4854
FF0C  BAE8  4855  MOV  CH, AL  ; SAVE ROW VALUE IN CH
FF0E  B2ED  4856  ADD  CH, CH  ; +2 FOR EVEN-ODD FIELD
FAMO  BADC  4857  MOV  BL, AH  ; COLUMN VALUE TO BX
FAM2  ZAFF  4858  SUB  BH, DH  ; MULTIPLY BY 8 FOR MEDIUM RES
FAM0  8034990006  R  4859  CMP  CRT_MODE, 6  ; DETERMINE MEDIUM OR HIGH RES
FAM9  7504  485A  JNE  V3  ; NOT_HIGH_RES
FAM0  B104  485B  MOV  CL, 4  ; SHIFT VALUE FOR HIGH RES
FAMD  D0E4  485C  SAL  AH, 1  ; COLUMN VALUE TIMES 2 FOR HIGH RES
FAMF  440  485D  V3:  ; NOT_HIGH_RES
FAMF  D3E3  485E  SML  BX, CL  ; MULTIPLY #16 FOR HIGH RES
485F
4860
4861
4862
4863
4864
4865
4866
FF11  B04  4867  MOV  DL, AH  ; COLUMN VALUE FOR RETURN
FF13  B0F  4868  MOV  DH, AL  ; ROW VALUE
FF15  D0EE  4869  SHR  DH, 1  ; DIVIDE BY 4
FF17  D0EE  4870  SHR  DH, 1  ; FOR VALUE IN 0-25 RANGE
FF19  ED12  4871  JHP  SHORT V5  ; LIGHT_PEN_RETURN_SET
4872
4873
4874
FF1B  4875  V4:  ; ALPHA_PEN
FF1B  F6364A00  R  4876  DIV  BYTE PTR CRT_COLS  ; DETERMINE ROW,COLUMN VALUE
FFFF  80F  4877  MOV  DH, AL  ; ROWS TO DH
F831  B0A  4878  MOV  DL, AH  ; COLS TO DL
F833  B0E0  4879  SAL  AL, CL  ; MULTIPLY ROWS X 8
F835  BAE0  4880  MOV  CH, AL  ; GET RASTER VALUE TO RETURN REG
F827  BADC  4881  MOV  BL, AH  ; COLUMN VALUE
F829  32F  4882  XCR  BH, DH  ; TO BX
F82B  D3E3  4883  SAL  BX, CL  ; LIGHT_PEN_RETURN_SET
F82D  4004  V5:  ; LIGHT_PEN_RETURN_SET
F82D  D401  4885  MOV  AH, 1  ; INDICATE EVERYTHING SET
F82F  4886  V6:  ; LIGHT_PEN_RETURN
F82F  S2  4887  PUSH  DX  ; SAVE RETURN VALUE (IN CASE)
F830  D2166300  R  4888  MOV  DX, ADDR_64X5  ; GET BASE ADDRESS
F834  83C207  4889  ADD  DX, 7  ; POINT TO RESET PARM
F837  EE   4890  OUT  DX, AL  ; ADDRESS, NOT DATA, IS IMPORTANT

A-66
MEMORY_SIZE_DETERMINE PROC FAR

F041 6F  ; STI ; INTERRUPTS BACK ON
F041 1E  ; PUSH DS ; SAVE SEGMENT
F043 B04000  R  ; MOV AX,DATA ; ESTABLISH ADDRESSING
F046 8E0B  ; MOV DS,AX
F046 A13000  R  ; MOV AX,MEMORY_SIZE ; SET VALUE
F049 IF  ; POP DS ; RECOVER SEGMENT
F04C CF  ; IRET ; RETURN TO CALLER

MEMORY_SIZE_DETERMINE ENDP

; --------------------------

ASSUME CS:CODE,DS:DATA

MEMORY_SIZE_DETERMINE PROC

F03B 5A  ; POP DX ; RECOVER VALUE
F039  ; V7: ; RETURN_NO_RESET
F039 5F  ; POP DI
F03A 5E  ; POP SI
F03B 1F  ; POP DS ; DISCARD SAVED BX,CX,DX
F03C 1F  ; POP DS
F03D 1F  ; POP DS
F03E 1F  ; POP DS
F03F 07  ; POP ES
F040 CF  ; IRET

READ_LPNI ENDP

; --------------------------

; IN3 12 ------------------------

; MEMORY_SIZE_DETERMINE

; THIS ROUTINE DETERMINES THE AMOUNT OF MEMORY IN THE SYSTEM

; AS REPRESENTED BY THE SWITCHES ON THE PLANAR. NOTE THAT

; THE SYSTEM MAY NOT BE ABLE TO USE I/O MEMORY UNLESS THERE

; IS A FULL COMPLEMENT OF 64K BYTES ON THE PLANAR.

; INPUT

; NO REGISTERS

; THE MEMORY_SIZE VARIABLE IS SET DURING POWER ON DIAGNOSTICS

; ACCORDING TO THE FOLLOWING HARDWARE ASSUMPTIONS:

; PORT 60 BITS 3.2 = 00 - 16K BASE RAM
; 01 - 32K BASE RAM
; 10 - 48K BASE RAM
; 11 - 64K BASE RAM

; PORT 62 BITS 3-0 INDICATE AMOUNT OF I/O RAM IN 32K INCREMENTS

; E.G., 0000 = NO RAM IN I/O CHANNEL
; 0010 = 64K RAM IN I/O CHANNEL, ETC.

; OUTPUT

; (AX) = NUMBER OF CONTIGUOUS 1K BLOCKS OF MEMORY

; THIS ROUTINE ATTEMPTS TO DETERMINE WHAT OPTIONAL

; DEVICES ARE ATTACHED TO THE SYSTEM.

; NO REGISTERS

; THE EQUIP_FLAG VARIABLE IS SET DURING THE POWER ON DIAGNOSTICS

; USING THE FOLLOWING HARDWARE ASSUMPTIONS:

; PORT 60 LOW ORDER BYTE OF EQUIPMENT

; PORT 3FA INTERRUPT 10 REGISTER OF 8250

; BITS 7-3 ARE ALWAYS 0

; PORT 378 = OUTPUT PORT OF PRINTER -- 8255 PORT THAT

; CAN BE READ AS WELL AS WRITTEN

; BIT 0 = IPL FROM DISKETTE -- THIS BIT INDICATES THAT THERE ARE DISKETTE

; DRIVES ON THE SYSTEM

; NO OTHER REGISTERS AFFECTED

; --------------------------
FR40 EQUIPMENT  PROC  FAR
FR40 FB  4967  STI ; INTERRUPTS BACK ON
FR4E 1E  4969  PUSH DS ; SAVE SEGMENT REGISTER
FR4F B04000  R  4970  MOV AX,DATA ; ESTABLISH ADDRESSING
FR52 BED8  4971  MOV DS,AX
FR54 A10000  R  4972  MOV AX,EQUIP_FLAG ; GET THE CURRENT SETTINGS
FR57 IF  4973  POP DS ; RECOVER SEGMENT
FR58 CF  4974  IRET ; RETURN TO CALLER
FR49 EQUIPMENT  ENDP
FR49 ;--- INT 15 -------------------
FR55 ; CASSETTE I/O
FR56 ; (AH) = 0  TURN CASSETTE MOTOR ON
FR57 ; (AH) = 1  TURN CASSETTE MOTOR OFF
FR58 ; (AH) = 2  READ 1 OR MORE 256 BYTE BLOCKS FROM CASSETTE
FR59 ; (ES,BX) = POINTER TO DATA BUFFER
FR60 ; (CX) = COUNT OF BLOCKS TO READ
FR61 ; ON EXIT:
FR62 ; (AX) = COUNT OF BYTES ACTUALLY READ
FR63 ; (CY) = 0  IF NO ERROR OCCURRED
FR64 ; = 1  IF ERROR OCCURRED
FR65 ; = 3  WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE
FR66 ; (ES,BX) = POINTER TO DATA BUFFER
FR67 ; (CX) = COUNT OF BLOCKS TO WRITE
FR68 ; ON EXIT:
FR69 ; (AX) = COUNT OF BYTES WRITTEN
FR6A ; (AH) = 0  IF NO DATA WAS FOUND
FR6B ; (AH) = 2  IF CRC ERROR WAS DETECTED
FR6C ; (AH) = 3  IF DATA TRANSITIONS ARE LOST
FR6D ; (AH) = 4  IF NO DATA WAS FOUND
FR6E ; (AH) = 5  IF ERROR OCCURRED
FR6F ; (AH) = 6  IF CRC ERROR WAS DETECTED
FR70 ; (AH) = 7  IF DATA TRANSITIONS ARE LOST
FR71 ; (AH) = 8  IF NO DATA WAS FOUND
FR72 ; (AH) = 9  IF ERROR OCCURRED
FR73 ; (AH) = 10  IF CRC ERROR WAS DETECTED
FR74 ; (AH) = 11  IF DATA TRANSITIONS ARE LOST
FR75 ; (AH) = 12  IF NO DATA WAS FOUND
FR76 ; (AH) = 13  IF ERROR OCCURRED
FR77 ; (AH) = 14  IF CRC ERROR WAS DETECTED
FR78 ; (AH) = 15  IF DATA TRANSITIONS ARE LOST
FR79 ; (AH) = 16  IF NO DATA WAS FOUND
FR7A ; (AH) = 17  IF ERROR OCCURRED
FR7B ; (AH) = 18  IF CRC ERROR WAS DETECTED
FR7C ; (AH) = 19  IF DATA TRANSITIONS ARE LOST
FR7D ; (AH) = 20  IF NO DATA WAS FOUND
FR7E ; (AH) = 21  IF ERROR OCCURRED
FR7F ; (AH) = 22  IF CRC ERROR WAS DETECTED
FR80 ; (AH) = 23  IF DATA TRANSITIONS ARE LOST
FR81 ; (AH) = 24  IF NO DATA WAS FOUND
FR82 ; (AH) = 25  IF ERROR OCCURRED
FR83 ; (AH) = 26  IF CRC ERROR WAS DETECTED
FR84 ; (AH) = 27  IF DATA TRANSITIONS ARE LOST
FR85 ; (AH) = 28  IF NO DATA WAS FOUND
FR86 ; (AH) = 29  IF ERROR OCCURRED
FR87 ; (AH) = 30  IF CRC ERROR WAS DETECTED
FR88 ; (AH) = 31  IF DATA TRANSITIONS ARE LOST
FR89 ; (AH) = 32  IF NO DATA WAS FOUND
FR8A ; (AH) = 33  IF ERROR OCCURRED
FR8B ; (AH) = 34  IF CRC ERROR WAS DETECTED
FR8C ; (AH) = 35  IF DATA TRANSITIONS ARE LOST
FR8D ; (AH) = 36  IF NO DATA WAS FOUND
FR8E ; (AH) = 37  IF ERROR OCCURRED
FR8F ; (AH) = 38  IF CRC ERROR WAS DETECTED
FR90 ; (AH) = 39  IF DATA TRANSITIONS ARE LOST
FR91 ; (AH) = 40  IF NO DATA WAS FOUND
FR92 ; (AH) = 41  IF ERROR OCCURRED
FR93 ; (AH) = 42  IF CRC ERROR WAS DETECTED
FR94 ; (AH) = 43  IF DATA TRANSITIONS ARE LOST
FR95 ; (AH) = 44  IF NO DATA WAS FOUND
FR96 ; (AH) = 45  IF ERROR OCCURRED
FR97 ; (AH) = 46  IF CRC ERROR WAS DETECTED
FR98 ; (AH) = 47  IF DATA TRANSITIONS ARE LOST
FR99 ; (AH) = 48  IF NO DATA WAS FOUND
FR9A ; (AH) = 49  IF ERROR OCCURRED
FR9B ; (AH) = 50  IF CRC ERROR WAS DETECTED
FR9C ; (AH) = 51  IF DATA TRANSITIONS ARE LOST
FR9D ; (AH) = 52  IF NO DATA WAS FOUND
FR9E ; (AH) = 53  IF ERROR OCCURRED
FR9F ; (AH) = 54  IF CRC ERROR WAS DETECTED
FR90 ; (AH) = 55  IF DATA TRANSITIONS ARE LOST
FR91 ; (AH) = 56  IF NO DATA WAS FOUND
FR92 ; (AH) = 57  IF ERROR OCCURRED
FR93 ; (AH) = 58  IF CRC ERROR WAS DETECTED
FR94 ; (AH) = 59  IF DATA TRANSITIONS ARE LOST
FR95 ; (AH) = 60  IF NO DATA WAS FOUND
FR96 ; (AH) = 61  IF ERROR OCCURRED
FR97 ; (AH) = 62  IF CRC ERROR WAS DETECTED
FR98 ; (AH) = 63  IF DATA TRANSITIONS ARE LOST
FR99 ; (AH) = 64  IF NO DATA WAS FOUND
FR9A ; (AH) = 65  IF ERROR OCCURRED
FR9B ; (AH) = 66  IF CRC ERROR WAS DETECTED
FR9C ; (AH) = 67  IF DATA TRANSITIONS ARE LOST
FR9D ; (AH) = 68  IF NO DATA WAS FOUND
FR9E ; (AH) = 69  IF ERROR OCCURRED
FR9F ; (AH) = 70  IF CRC ERROR WAS DETECTED
FR90 ; (AH) = 71  IF DATA TRANSITIONS ARE LOST
FR91 ; (AH) = 72  IF NO DATA WAS FOUND
FR92 ; (AH) = 73  IF ERROR OCCURRED
FR93 ; (AH) = 74  IF CRC ERROR WAS DETECTED
FR94 ; (AH) = 75  IF DATA TRANSITIONS ARE LOST
FR95 ; (AH) = 76  IF NO DATA WAS FOUND
FR96 ; (AH) = 77  IF ERROR OCCURRED
FR97 ; (AH) = 78  IF CRC ERROR WAS DETECTED
FR98 ; (AH) = 79  IF DATA TRANSITIONS ARE LOST
FR99 ; (AH) = 80  TO BE RETURNED (INVALID COMMAND).
FR9A ; --- INT 15 -------------------
FR9B ; PURPOSE:
FR9C ; TO CALL APPROPRIATE ROUTINE DEPENDING ON REG AH
FR9D ; ---AH ROUTINE---
FR9E ; ---DAE4---
FR9F ; OR AH,AH ;TURN ON MOTOR?
FR90 ; JZ MOTOR_ON ;YES, DO IT
FR91 ; DEC AH ;TURN OFF MOTOR?
FR92 ; JZ MOTOR_OFF ;YES, DO IT
FR93 ; DEC AH ;READ CASSETTE BLOCK?
FR94 ; JZ READ_BLOCK ;YES, DO IT
FR95 ; DEC AH ;WRITE CASSETTE BLOCK?
FR96 ; JNZ H2 ;NOT_DEFINED
FR97 ; JMP WRITE_BLOCK ;YES, DO IT
FR98 ; ---DAE6---
FR99 ; W2:  ;COMMAND NOT DEFINED
FR9A ; MOV AH,80H ;ERROR, UNDEFINED OPERATION
FR9B ; STC ;ERROR FLAG
FR9C ; ---C---
FR9D ; RET
FR9E ; W1  ;ENDP
F805
5043  MOTOR_ON  PROC  NEAR
5044 ;---------------------------------------------
5045 ; PURPOSE:
5046 ; TO TURN ON CASSETTE MOTOR
5047 ;---------------------------------------------
F805  E641
5048  IN  AL,PORT_B  ;READ CASSETTE OUTPUT
F807  24F7
5049  AND  AL,00H  ; CLEAR BIT TO TURN ON MOTOR
F809  E661
5050  W3:  OUT  PORT_D,AL  ;WRITE IT OUT
F882  2AE4
5051  SUB  AH,AH  ;CLEAR AH
F800  CA
5052  RET
5053  MOTOR_ON  ENDP
5054
F80E  MOTOR_OFF  PROC  NEAR
5055 ;---------------------------------------------
5056 ; PURPOSE:
5057 ; TO TURN CASSETTE MOTOR OFF
5058 ;---------------------------------------------
F80E  E641
5059  IN  AL,PORT_B  ;READ CASSETTE OUTPUT
F900  DC08
5060  OR  AL,00H  ; SET BIT TO TURN OFF
F902  E6F5
5061  JMP  W3  ;WRITE IT, CLEAR ERROR, RETURN
5062
F804  MOTOR_OFF  ENDP
5063
F804  READ_BLOCK  PROC  NEAR
5064 ;---------------------------------------------
5065 ; PURPOSE:
5066 ; TO READ 1 OR MORE 256 BYTE BLOCKS FROM CASSETTE
5067 ;---------------------------------------------
F804  SS
5068  PUSH  BX  ;SAVE BX
F899  51
5069  PUSH  CX  ;SAVE CX
F899  56
5070  PUSH  SI  ;SAVE SI
F897  BE0700
5071  MOV  SI,7  ;SET UP RETRY COUNT FOR LEADER
F904  EC201
5072  CALL  BEGIN_DP  ;BEGIN BY STARTING MOTOR
F909
5073  W4:  ;SEARCH FOR LEADER
F806  E662
5074  IN  AL,PORT_C  ;GET INITIAL VALUE
F909  2410
5075  AND  AL,010H  ;MASK OFF EXTRANEOUS BITS
F8A1  A80000  R
5076  MOV  AL,01H  ;MUST HAVE AT LEAST THIS MANY ONE SIZE
F8A4  BA7A3F
5077  MOV  DX,16250  ;B OF TRANSITIONS TO LOOK FOR
F8A4
5078  J53
5079  TEST  BIOS_BREAK, 80H  ;CHECK FOR BREAK KEY
F8A4  7C03
5080  JZ  W6  ;JUMP IF NO BREAK KEY
F8AE  E9A000
5081  JMP  W17  ;JUMP IF BREAK KEY HIT
F8A0  4A
5082  DEC  DX  ;--WAIT FOR EDGE
F801  7503
5083  JNZ  W7  ;JUMP IF BEGINNING OF LEADER
F804  E90400
5084  JMP  W17  ;JUMP IF NO LEADER FOUND
F807  EC600
5085  W7:  CALL  READ_HALF_BIT  ;IGNORE FIRST EDGE
F8A3  E3EB
5086  JCXZ  W5  ;JUMP IF NO EDGE DETECTED
F80C  BA7003
5087  MOV  DX,0370H  ;CHECK FOR HALF BITS
F8BF  B90002
5088  MOV  CX,200H  ;MUST HAVE AT LEAST THIS MANY ONE SIZE
F8C2  E621
5089  IN  AL,021H  ;INTERRUPT MASK REGISTER
F8C4  GCO
5090  OR  AL,1  ;DISABLE TIMER INTERRUPTS
F8C6  E621
5091  OUT  021H,AL  ;------SEARCH-LDR
F8C0  9W:
5092  TEST  BIOS_BREAK, 80H  ;CHECK FOR BREAK KEY
F8CD  756C
5093  JNZ  W17  ;JUMP IF BREAK KEY HIT
F8CF  51
5094  PUSH  CX  ;SAVE REG CX
F8DD  E6AD00
5095  CALL  READ_HALF_BIT  ;GET PULSE WIDTH
F8D3  08C9
5096  OR  CX, CX  ;CHECK FOR TRANSITION
F8DB  59
5097  POP  CX  ;RESTORE ONE BIT COUNTER
F8DE  74C5
5098  JZ  W4  ;JUMP IF NO TRANSITION
F8DF  3B03
5099  CMP  DX,BX  ;CHECK PULSE WIDTH
; A SYNCH BIT HAS BEEN FOUND. READ SYN CHARACTER:
.F808 266000 R 5151 MDV CRC_REG,0FFFFH ;INIT CRC REG
.F806 BAA001 R 5152 MDV DX,256 ;SET DX TO DATA BLOCK SIZE
.F809 F606710000 R 5154 TEST BIOS_BREAK, 00H ; CHECK FOR BREAK KEY
.F80F 7523 R 5155 JNZ W13 ; JUMP IF BREAK KEY HIT
.F900 E64F00 R 5156 CALL READ_BYTE ; READ BYTE FROM CASSETTE
.F903 721E R 5157 JC W13 ; CY SET INDICATES NO DATA TRANSITIONS
.F905 E305 R 5158 JCXZ W12 ; IF WE'VE ALREADY REACHED
.F910 7506 R 5159 POP DX ; END OF MEMORY BUFFER
.F914 E1F9 R 5160 MOV AH,OlH ; SET AH;:;02 TO INDICATE
.F915 24E9 R 5161 MOV ES: [BX,AL] ; STORE DATA BYTE AT BYTE PTR
.F90A 33 R 5162 INC BX ; INC BUFFER PTR
.F90B 49 R 5163 INC CX ; INC BYTE COUNTER
.F90C 44 R 5164 JCXZ W12 ; LOOP UNTIL DATA BLOCK HAS BEEN READ FROM CASSETTE.
.F90D 7E19 R 5165 DEC DX ; DEC BLOCK CNT
.F90E 7EFA R 5166 JG W11 ; RD_BLK
.F90F E6400 R 5167 CALL READ_BYTE ; NOW READ TWO CRC BYTES
.F912 E83D00 R 5168 CALL READ_BYTE
.F915 24E9 R 5169 SUB AH, AH ; CLEAR AH
.F917 83169000F1D R 5170 CMP CRC_REG,100FH ; IS THE CRC CORRECT
.F91D 7E06 R 5171 JNE W14 ; IF NOT EQUAL CRC IS BAD
.F91F E306 R 5172 JCXZ W15 ; IF BYTE COUNT IS ZERO
.F921 EBCD R 5173 POP DX ; PREPARE FOR SYNCHRONIZATION
.F922 7E13 R 5174 JMP W10 ; STILL MORE, SO READ ANOTHER BLOCK
.F923 0401 R 5175 POP AH,0IH ; SET AH;:02 TO INDICATE
.F925 80 R 5176 MOV AH,0IH ; CLEAR AH
.F926 F9C4 R 5177 INC AH ; EXIT EARLY ON ERROR
.F927 05 R 5178 POP DX ; CALCULATE COUNT OF
.F928 2D01 R 5179 INC AX ; DATA BYTES ACTUALLY READ
.F92A 30 R 5179 POP AX ; SAVE AX (RET CODE)
.F92B F9C03 R 517A TEST AH, 03H ; CHECK FOR ERRORS
.F92E 7513 R 517B JNZ W18 ; JUMP IF ERROR DETECTED
.F93D E1F00 R 5179 CALL READ_BYTE ; READ TRAILER
.F93E 8E0E R 517A JMP SHORT W15 ; Skip to turn off motor
.F93F 35 R 517B POP AH;:;01 TO INDICATE CRC ERROR
.F93F 35 R 517C INC AH ; ERROR DETECTED
.F93F 35 R 517D INC AX ; BAD-LEADER
LOC OBJ  
LINE  
SOURCE  

F936 7403  5194  JZ  W17  ; JUMP IF TOO MANY RETRIEVS  
F938 E92FF  5195  JMP  W4  ; JUMP IF NOT TOO MANY RETRIEVS  
F938 5E  5196  W17:  ; NO VALID DATA FOUND  
5197  --- NO DATA FROM CASSETTE ERROR, I.E. TIMEOUT  
5198  
F93B 5E  5199  POP  SI  ; RESTORE REGS  
F93C 59  5200  POP  CX  ; RESTORE REGS  
F93D 58  5201  POP  BX  
F93E 2802  5202  SUB  DX,DX  ; ZERO NUMBER OF BYTES READ  
F940 B004  5203  MOV  AH,04H  ; TIME OUT ERROR (NO LEADER)  
F942 50  5204  PUSH  AX  
F943 50F5  5205  W10:  ; MOT-OFF  
F945 E621  5206  IN  AL,021H  ; RE_ENABLE INTERRUPTS  
F945 24FE  5207  AND  AL,OFFH-1  
F947 E621  5208  OUT  021H,AL  
F949 E92FF  5209  CALL  MOTOR_OFF  ; TURN OFF MOTOR  
F94C 5B  5210  POP  AX  ; RESTORE RETURN CODE  
F94D 80FC01  5211  CMP  AH,01H  ; SET CARRY IF ERROR (AH=0)  
F950 F5  5212  CRC  
F951 C3  5213  RET  ; FINISHED  
5214  READ_BLOCK  ENDP  
5215  
F952  
5216  READ_BYTE  PROC  NEAR  
5217  ; PURPOSE:  
5218  ; TO READ A BYTE FROM CASSETTE  
5219  ;  
5220  ; ON EXIT REG AL CONTAINS READ DATA BYTE  
5221  ; -----------------------------  
5222  F952 53  5222  PUSH  BX  ; SAVE REGS BX,CX  
F953 51  5223  PUSH  CX  
F954 B108  5224  MOV  CL,8H  ; SET BIT COUNTER FOR 8 BITS  
F956 5225  W19:  ; BYTE-ASM  
F956 51  5226  PUSH  CX  ; SAVE CX  
5227  ; -----------------------------  
5228  F957 62600  5228  CALL  READ_HALF_BIT  ; READ ONE PULSE  
F95A E320  5229  JCXZ  W21  ; IF CX=0 THEN TIMEOUT  
5230  F95C 53  5231  PUSH  BX  ; BECAUSE OF NO DATA TRANSITIONS  
5232  F95D 58  5233  POP  AX  ; SAVE 1ST HALF BIT'S  
5234  F95E 802000  5235  CALL  READ_HALF_BIT  ; READ COMPLEMENTARY PULSE  
F960 50  5236  POP  AX  ; COMPUTE DATA BIT  
F961 E319  5237  JCXZ  W21  ; IF CX=0 THEN TIMEOUT DUE TO  
5238  F963 0300  5239  ADD  BX,AX  ; NO DATA TRANSITIONS  
F965 81FB006  5240  CMP  BX,06FH  ; CHECK FOR ZERO BIT  
F969 F5  5241  CMP  AH  ; CARRY IS SET IF ONE BIT  
F96A 9F  5242  LAHF  ; SAVE CARRY IN AH  
F96B 59  5243  POP  CK  ; RESTORE CK  
5244  ; NOTE:  
5245  ; MS BIT OF BYTE IS READ FIRST.  
5246  ; REG CH IS SHIFTED LEFT WITH  
5247  ; CARRY BEING INSERTED INTO LS  
5248  ; BIT OF CH.  
5249  ; AFTER ALL 8 BITS HAVE BEEN  
5250  ; READ, THE MS BIT OF THE DATA BYTE  
5251  ; WILL BE IN THE MS BIT OF REG CH  
F96C D005  5252  RCL  CH,1  ; ROTATE REG CH LEFT 1 WITH CARRY TO  
5253  ; LS BIT OF REG CH  
F96E 9E  5254  SAHF  ; RESTORE CARRY FOR CRC ROUTINE  
F96F 8000  5255  CALL  CRC_GEN  ; GENERATE CRC FOR BIT  
F972 FEC9  5256  DEC  CL  ; LOOP TILL ALL 6 BITS OF DATA  
5257  F974 75E0  5258  JNZ  W19  ; BYTE-ASM  
F976 8AC5  5259  MOV  AL,CH  ; RETURN DATA BYTE IN REG AL  
F978 58  5260  CLC  
F979 5261  W20:  ; RD-BY-EX  
F979 59  5262  POP  CX  ; RESTORE REGS CX,BX  
F97A 5B  5263  POP  BX  
F97B C3  5264  RET  ; FINISHED  
F97C 5265  W21:  ; NO-DATA  
F97C 59  5266  POP  CX  ; RESTORE CX  
F97D F9  5267  STC  ; INDICATE ERROR
LOC OBJ

F97E EBF9 5268 JMP H20 ; RD_BYTE_EX
F979 5269 READ_BYTE ENDP
F970 5270 ;---------------------------------------------
F96D 5271 READ_HALF_BIT PROC NEAR
F972 5272 ; PURPOSE:
F973 5273 ; TO COMPUTE TIME TILL NEXT DATA
F974 5274 ; TRANSITION (EDGE)
F975 5275 ;
F976 5276 ; ON ENTRY:
F977 5277 ; EDGE_CNT CONTAINS LAST EDGE COUNT
F978 5278 ;
F979 5279 ; ON EXIT:
F980 5280 ; AX CONTAINS OLD LAST EDGE COUNT
F981 5281 ; BX CONTAINS PULSE WIDTH (HALF BIT)
F982 5282 ;---------------------------------------------
F980 5283 MOV CX, 100 ; SET TIME TO WAIT FOR BIT
F993 5284 MOV AL, LAST_VAL ; GET PRESENT INPUT VALUE
F987 5285 MOV W23 ; RD-H-BIT
F987 5286 IN AL, PORT_C ; INPUT DATA BIT
F989 5287 AND AL, 010H ; MASK OFF EXTRANEOUS BITS
F98B 5288 CMP AL, AH ; SAME AS BEFORE?
F98D 5289 LOOP MW2 ; LOOP TILL IT CHANGES
F98F 5290 MOV LAST_VAL, AL ; UPDATE LAST_VAL WITH NEW VALUE
F992 5291 MOV AL, 0 ; READ TIMER'S COUNTER COMMAND
F994 5292 OUT TIM_CTL, AL ; LATCH COUNTER
F996 5293 IN AL, TIMERO ; GET LS BYTE
F998 5294 MOV AH, AL ; SAVE IN AH
F99A 5295 IN AL, TIMERO ; GET MS BYTE
F99C 5296 XCHG AL, AH ; XCHG AL, AH
F99E 5297 MOV BX, EDGE_CNT ; BX GETS LAST EDGE COUNT
F9A2 5298 SUB BX, AX ; SET BX EQUAL TO HALF BIT PERIOD
F9A4 5299 MOV EDGE_CNT, AX ; UPDATE EDGE COUNT;
F9A7 C3 5300 RET ;---------------------------------------------
F98B 5301 READ_HALF_BIT ENDP
F9A8 5302 ;---------------------------------------------
F9A8 5303 WRITE_BLOCK PROC NEAR
F9A9 5304 ;
F9A9 5305 ; WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE.
F9AB 5306 ; THE DATA IS PADDED TO FILL OUT THE LAST 256 BYTE BLOCK.
F9AC 5307 ;
F9AD 5308 ; ON ENTRY:
F9AE 5309 ; BX POINTS TO MEMORY BUFFER ADDRESS
F9AF 5310 ; CX CONTAINS NUMBER OF BYTES TO WRITE
F9B0 5311 ;
F9B1 5312 ; ON EXIT:
F9B2 5313 ; BX POINTS 1 BYTE PAST LAST BYTE WRITTEN TO CASSETTE
F9B3 5314 ; CX IS ZERO
F9B4 5315 ;
F9B5 5316 PUSH BX
F9B9 5317 PUSH CX
F9B4 5318 IN AL, PORT_B ; DISABLE SPEAKER
F9B4 5319 AND AL, NOT 02H
F9B4 5320 OR AL, 01H ; ENABLE TIMER
F9B0 5321 OUT PORT_B, AL
F9B2 5322 MOV AL, 086H ; SET UP TIMER -- MODE 3 SQUARE WAVE
F9B4 5323 OUT TIM_CTL, AL
F9B6 5324 CALL BEGIN_OP ; START MOTOR AND DELAY
F9B8 5325 MOV AX, 11045 ; SET NORMAL BIT SIZE
F9B0 5326 CALL WS1 ; SET_TIMER
F9B8 5327 MOV CX, 0800H ; SET CX FOR LEADER BYTE COUNT
F9C2 5328 5329 MOV H23 ; WRITE LEADER
F9C2 5329 STC ; WRITE ONE BITS
F9C3 5330 CALL WRITE_BIT
F9C6 5331 LOOP H23 ; LOOP 'TIL LEADER IS WRITTEN
F9C8 5332 CLC ; WRITE SYNC BIT (0)
F9C9 5333 CALL WRITE_BIT
F9CC 59 5334 POP CX ; RESTORE REGS CX,BX
F9C0 5335 POP BX
F9CE 5336 MOV AL, 16H ; WRITE SYN CHARACTER
F9CF 5337 MOV AX, 16H ; WRITE SYN CHARACTER

A-72
<table>
<thead>
<tr>
<th>LOC</th>
<th>OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5330</td>
<td></td>
<td>----</td>
<td>------</td>
</tr>
<tr>
<td>5338</td>
<td></td>
<td>WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE</td>
<td></td>
</tr>
<tr>
<td>5339</td>
<td></td>
<td>; WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE</td>
<td></td>
</tr>
<tr>
<td>5340</td>
<td></td>
<td>; CREATE CRC REG, OFFSET</td>
<td></td>
</tr>
<tr>
<td>5341</td>
<td></td>
<td>R INIT CRC</td>
<td></td>
</tr>
<tr>
<td>5342</td>
<td></td>
<td>FOR 256 BYTES</td>
<td></td>
</tr>
<tr>
<td>5343</td>
<td></td>
<td>; WRITE CRC TO OFFSET</td>
<td></td>
</tr>
<tr>
<td>5344</td>
<td></td>
<td>; BX POINTS TO MEMORY BUFFER ADDRESS</td>
<td></td>
</tr>
<tr>
<td>5345</td>
<td></td>
<td>; CX CONTAINS NUMBER OF BYTES TO WRITE</td>
<td></td>
</tr>
<tr>
<td>5346</td>
<td></td>
<td>; ON EXIT:</td>
<td></td>
</tr>
<tr>
<td>5347</td>
<td></td>
<td>; BX POINTS 1 BYTE LAST BYTE WRITTEN TO CASSETTE</td>
<td></td>
</tr>
<tr>
<td>5348</td>
<td></td>
<td>; CX IS ZERO</td>
<td></td>
</tr>
<tr>
<td>5349</td>
<td></td>
<td>--------------------------</td>
<td></td>
</tr>
</tbody>
</table>

FA03 5349 WR_BLOCK:

FA03 C7066900FFFF R 5350 MOV CRC_REG, OFFFH ;INIT CRC
FA03 BA0001 5351 MOV DX, 256 ; FOR 256 BYTES
FA03 5352 M24: ; WR-BLK
FA03 26A007 5353 MOV AL, 0E9 |BX| ; READ BYTE FROM MEM
FA03 F83500 5354 CALL WRITE_BYTE ; WRITE IT TO CASSETTE
FA03 E302 5355 JCXZ W25 ; UNLESS CX=0, ADVANCE PTRS & DEC COUNT
FA03 43 5356 INC BX ; INC BUFFER POINTER
FA03 49 5357 DEC CX ; DEC BYTE COUNTER
FA03 5358 M25: ; SKIP-ADV
FA03 4A 5359 DEC DX ; DEC BLOCK_CNT
FA0E 7FF3 5360 POP HG24 ; LOOP TILL 256 BYTE BLOCK
FA07 5361 ; IS WRITTEN TO TAPE

FA09 A16900 R 5362 ------------------- WRITE CRC -------------------
FA09 A16900 5363 ; WRITE 1'S COMPLEMENT OF CRC REG TO CASSETTE
FA09 5364 ; WHICH IS CHECKED FOR CORRECTNESS WHEN THE BLOCK IS READ
FA09 5365 1
FA09 5366 ; REG AX IS MODIFIED
FA09 5367 ---------------------------
FA09 5368 MOV AX, CRC_REG ; WRITE THE ONE'S COMPLEMENT OF THE
FA09 5369 ; TWO BYTE CRC TO TAPE
FA0C F700 5370 NOT AX ; FOR 1'S COMPLEMENT
FA0E 5371 PUSH AX ; SAVE IT
FA0F 8600 5372 XOR AH, AL ; WRITE MS BYTE FIRST
FA10 E83300 5373 CALL WRITE_BYTE ; WRITE IT
FA10 5B 5374 POP AX ; GET IT BACK
FA11 E6F00 5375 CALL WRITE_BYTE ; HOW WRITE IS BYTE
FA11 8EC9 5376 OR CX, CX ; IS BYTE COUNT EXHAUSTED?
FA11 7507 5377 JNZ HR_BLOCK ; JMP IF NOT DONE YET
FA12 5C1 5378 PUSH CX ; SAVE REG CX
FA12 5D90 5379 MOV CX, 32 ; WRITE OUT TRAILER BITS
FA13 5380 FA00 5381 MZ6: ; TRAIL-LOOP
FA13 5382 FA00 F9 5383 SC 
FA13 5384 FA01 E82A00 5385 CALL WRITE_BIT
FA13 5386 FA04 E2FA 5387 LOOP MZ6 ; WRITE UNTIL TRAILER WRITTEN
FA13 5388 FA06 S9 5389 POP CX ; RESTORE REG CX
FA13 5390 FA07 8080 5391 MOV AL, 00H ; TURN TIMER2 OFF
FA13 5392 FA09 6E43 5393 OUT TIM_CTL, AL
FA13 5394 FA0B 80100 5395 MOV AX, 1
FA13 5396 FA0E E83300 5397 CALL W31 ; SET_TIMER
FA13 5398 FA11 E7AFE 5399 CALL MOTOR_OFF ; TURN MOTOR OFF
FA13 539A FA14 2BC0 539B SUB AX, AX ; NO ERRORS REPORTED ON WRITE OP
FA13 539C FA16 539D RET ; FINISHED
FA17 539E ------------------------------- WRITE_BLOCK ENDP
FA17 539F |----------------------------- |
FA17 539G FA17 539H WRITE_BYTE PROC NEAR
FA17 539J ; WRITE A BYTE TO CASSETTE.
FA17 539K ; BYE TO WRITE IS IN REG AL.
FA17 539L |----------------------------- |
FA17 539M FA17 539N PUSH CX ; SAVE REGS CX, AX
FA17 539O FA19 539P PUSH AX
FA19 5AEB 539Q MOV CH, AL ; A=BYTE TO WRITE.
FA19 5400 539R ; MS BIT WRITTEN FIRST?
FA19 5401 FA1B 539S MOV CL, 0 ; FOR 0 DATA BITS IN BYTE.
FA1B 5402 539T ; NOTE: TWO EDGES PER BIT
FA1B 5403 FA1D 539U ; DISASSEMBLE THE DATA BIT
FA1D 5404 FA1D 5405 RCL CH, 1 ; ROTATE MS BIT INTO CARRY
FA1F 5C 5406 PUSHF ; SAVE FLAGS
FA1F 5407 FA20 5408 CALL WRITE_BIT ; WRITE DATA BIT
FA20 5409 FA23 540A POPF ; RESTORE CARRY FOR CRC CALC
FA23 540B FA24 540C CALL CRC_GEN ; COMPUTE CRC ON DATA BIT
FA27 5FEC 540D DEC CL ; LOOP TILL ALL 0 BITS DONE
FA29 7FBE 540E JNZ MZ7 ; JUMP IF NOT DONE YET
FA2B 5410 FA2B 5412 POP AX ; RESTORE REGS AX, CX
FA2C 5414 FA2C 5415 POP CX

A-73
FA2D C3 5415  RET  ; THE ARE FINISHED
5416  WRITE_BYTE  ENDP
5417  

FA2E 5418  WRITE_BIT  PROC NEAR
5419  ; PURPOSE:
5420  ;
5421  ; TO WRITE A DATA BIT TO CASSETTE
5422  ; CARRY FLAG CONTAINS DATA BIT
5423  ; I.E. IF SET DATA BIT IS A ONE
5424  ; IF CLEAR DATA BIT IS A ZERO
5425  ;
5426  ; NOTE: TWO EDGES ARE WRITTEN PER BIT
5427  ; ONE BIT HAS 500 USEC BETWEEN EDGES
5428  ; FOR A 1000 USEC PERIOD (.5 MILLISEC)
5429  ;
5430  ; ZERO BIT HAS 250 USEC BETWEEN EDGES
5431  ; FOR A 500 USEC PERIOD (.5 MILLISEC)
5432  ; CARRY FLAG IS DATA BIT
5433  
5434  ;
5435  FA2E B04094 5436  ; ASsume IT'S A '1'
5437  FA31 7203 5438  ; SET AX TO NOMINAL ONE SIZE
5439  ;
5440  ;
5441  FA33 B50002 5442  ; MOV AX, 592 ; NO, SET TO NOMINAL ZERO SIZE
5443  ;
5444  FA36 5445  ; WRITE-BIT AX
5446  ;
5447  FA36 50 5448  ; PUSH AX
5449  ; WRITE BIT WITH PERIOD EQ TO VALUE AX
5450  ;
5451  FA37 5452  ; E642
5453  ; IN AL, PORT_C
5454  ; INPUT TIMER_0 OUTPUT
5455  ;
5456  FA39 2420 5457  ; AND AL, 020H
5458  ;
5459  FA3B 760A 5460  ; JZ W29
5461  ; LOOP TILL HIGH
5462  ;
5463  FA3D 5464  ; E642
5465  ; IN AL, PORT_C
5466  ; NOW WAIT TILL TIMER'S OUTPUT IS LOW
5467  ;
5468  FA3F 2420 5469  ; AND AL, 020H
5470  ;
5471  FA41 750A 5472  ; JNZ W30
5473  ;
5474  FA43 5475  ; 50
5476  ; POP AX
5477  ; RESTORE PERIOD COUNT
5478  ;
5479  FA44 5480  ; W31:
5481  ; SET TIMER
5482  ;
5483  FA44 5484  ; E642
5485  ; OUT 042H, AL
5486  ; SET LOW BYTE OF TIMER 2
5487  ;
5488  FA46 4004 5489  ; MOV AL, AH
5490  ;
5491  FA4A 5492  ; E642
5493  ; OUT 042H, AL
5494  ; SET HIGH BYTE OF TIMER 2
5495  
5496  FA4A C3 5497  ; RET
5498  
5499  FA4B 5499  ; CRC_GEN PROC NEAR
5500  ; UPDATE CRC REGISTER WITH NEXT DATA BIT
5501  ;
5502  ; CRC IS USED TO DETECT READ ERRORS
5503  ;
5504  ; ASSUMES DATA BIT IS IN CARRY
5505  ;
5506  ; REG AX IS MODIFIED
5507  ;
5508  ; FLAGS ARE MODIFIED
5509  ;
5510  FA4B A16900 5511  ; R
5512  ; THE FOLLOWING INSTRUCTIONS
5513  ; WILL SET THE OVERFLOW FLAG
5514  ; IF CARRY AND MS BIT OF CRC
5515  ; ARE UNEQUAL
5516  ;
5517  FA4E D108 5518  ; RCR AX, 1
5519  ;
5520  FA50 D100 5521  ; RCL AX, 1
5522  ;
5523  FA52 F0 5524  ; CJC ; CLEAR CARRY
5525  ;
5526  FA53 7104 5527  ; JND W32
5528  ; SKIP IF NO OVERFLOW
5529  ;
5530  FA55 351008 5531  ; XOR AX, 0810H
5532  ; THEN XOR CRC REG WITH
5533  ;
5534  FA56 5480  ; B O810H
5535  ;
5536  FA58 F9 5537  ; STC ; SET CARRY
5538  ;
5539  FA59 5482 5540  ; W32:
5541  ;
5542  FA59 D100 5543  ; RCL AX, 1 ; ROTATE CARRY (DATA BIT)
5544  ;
5545  FA5B A36900 5546  ; R
5547  ; MOV CRC_REG, AX ; UPDATE CRC_REG
5548  ;
5549  FA5E C3 5550  ; RET ; FINISHED
5551  
5552  FA5F 5553  ; BEGIN_OP PROC NEAR ; START TAPE AND DELAY
5554  ;
5555  ;
This routine allows the clock to be set/read

INPUT

READ THE CURRENT CLOCK SETTING

RETURNS AX = HIGH PORTION OF COUNT

DX = LOW PORTION OF COUNT

AL = 0 IF TIMER HAS NOT PASSED 24 HOURS SINCE LAST READ

Z:0 IF ON ANOTHER DAY

SET THE CURRENT CLOCK

NOTE: COUNTS OCCUR AT THE RATE OF 1193180/65536 COUNTS/SEC

(OR ABOUT 18.2 PER SECOND -- SEE EQUATES BELOW)

ASSUME CS:CODE, DS:DATA

STI ; INTERRUPTS BACK ON

PUSH DS ; SAVE SEGMENT

PUSH AX ; SAVE PARM

MOV AX,DATA

STI

POP AX

GET BACK INPUT PARM

INTERRUPTS BACK ON

SAVE SEGMENT

RETURN TO CALLER

TIME_OF_DAY

PROC FAR

AH=0

READ_TIME

AH;

SET_TIME

TOO RETURN

INTERRUPTS BACK ON

RECOVER SEGMENT

READ_TIME

NO INTERRUPTS WHILE READING

TIMER_OFL.O

GET OVERFLOW, AND RESET THE FLAG

TIMER_LOW

RESET OVERFLOW

TIMER_HIGH

INTERRUPTS BACK ON

TIME_OF_DAY

ENDP

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

TIMER_INT

PROC FAR

STI

PUSH DS

PUSH AX

PUSH OX ; SAVE MACHINE STATE

MOV AL,TIMER_OFL

INC TIMER_LOW

JNZ T4

TEST_DAY

INC TIMER_HIGH

JMP T1 ; TOD_RETURN

TIME_OF_DAY

ENDP

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

TIMER_INT

PROC FAR

STI

PUSH DS

PUSH AX

PUSH OX ; SAVE MACHINE STATE

MOV AX,DATA

MOV DS,AX

MOV AX,DATA

STI

MOV AX,DATA

MOV AX,DATA

JMP T1 ; TOD_RETURN

TIME_OF_DAY

ENDP

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

; THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM

; CHANNEL 0 OF THE 8253 TIMER. INPUT FREQUENCY IS 1.19318 MHZ

; AND THE DIVIDER IS 65536 RESULTING IN APPROX. 18.2 INTERRUPTS

; EVERY SECOND.

; THE INTERRUPT HANDLER MAINTAINS A COUNT OF INTERRUPTS SINCE POWER

; ON TIME, WHICH MAY BE USED TO ESTABLISH TIME OF DAY.

; THE INTERRUPT HANDLER ALSO DECREMENTS THE MOTOR CONTROL COUNT

; OF THE DISKETTE, AND WHEN IT EXPIRES, WILL TURN OFF THE DISKETTE

; MOTOR, AND RESET THE MOTOR RUNNING FLAGS

; THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE THROUGH INTERRUPT

; AT EVERY TIME TICK. THE USER MUST CODE A ROUTINE AND PLACE THE

; CORRECT ADDRESS IN THE VECTOR TABLE.

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

; INTERRUPTS BACK ON

; SAVE MACHINE STATE

; ESTABLISH ADDRESSABILITY

; INCREMENT TIME

; TEST_DAY

; TEST_DAY

; INT "1A" -------------------------------

; TIME_OF_DAY

; THIS ROUTINE ALLOWS THE CLOCK TO BE SET/READ

; INPUT

; READ THE CURRENT CLOCK SETTING

; RETURNS AX = HIGH PORTION OF COUNT

; DX = LOW PORTION OF COUNT

; AL = 0 IF TIMER HAS NOT PASSED 24 HOURS SINCE LAST READ

; Z:0 IF ON ANOTHER DAY

; SET THE CURRENT CLOCK

; NOTE: COUNTS OCCUR AT THE RATE OF 1193180/65536 COUNTS/SEC

; (OR ABOUT 18.2 PER SECOND -- SEE EQUATES BELOW)

; ASSUME CS:CODE, DS:DATA

; INTERRUPTS BACK ON

; SAVE SEGMENT

; GET BACK INPUT PARM

; INTERRUPTS BACK ON

; RECOVER SEGMENT

; RETURN TO CALLER

; READ_TIME

; NO INTERRUPTS WHILE READING

; TIMER_OFL.O

; GET OVERFLOW, AND RESET THE FLAG

; TIMER_LOW

; RESET OVERFLOW

; TIMER_HIGH

; INTERRUPTS BACK ON

; SAVE MACHINE STATE

; MOV AX,DATA

; STI

; MOV AX,DATA

; JMP T1 ; TOD_RETURN

; TIME_OF_DAY

; ENDP

; INTERRUPTS BACK ON

; SAVE SEGMENT

; GET BACK INPUT PARM

; INTERRUPTS BACK ON

; RECOVER SEGMENT

; RETURN TO CALLER

; READ_TIME

; NO INTERRUPTS WHILE READING

; TIMER_OFL.O

; GET OVERFLOW, AND RESET THE FLAG

; TIMER_LOW

; RESET OVERFLOW

; TIMER_HIGH

; INTERRUPTS BACK ON

; SAVE MACHINE STATE

; MOV AX,DATA
TIMER_HAS_GONE_24_HOURS

----- TEST FOR DISKETTE TIME OUT

------ TEST FOR COUNT EQUALLING 24 HOURS

------ DISKETTE_CTL

------ MOVE TO INTERRUPTS

------ RETURN FROM INTERRUPT
LOC OBJ LINE SOURCE

FF33 0000 5791  DW 00000H ; INTERRUPT 10H
FF35 00F6 5792  DW 0F600H ; ROM BASIC ENTRY POINT
FF37 FF26 5793  DW OFFSET BOOT Strap ; INTERRUPT 19H
FF39 00F0 5794  DW CODE
FF3B 00FE R 5795  DW TIME_OF_DAY ; INTERRUPT 1AH -- TIME OF DAY
FF3D 00F0 R 5796  DW CODE
FF3F 00FF R 5797  DW CODE
FF40 00F0 R 5798  DW CODE
FF41 00F0 R 5799  DW CODE
FF43 00FF R 5800  DW DUMMY_RETURN ; INTERRUPT 1BH -- KEYBOARD BREAK ADDR
FF44 00F0 R 5801  DW CODE
FF45 00F0 R 5802  DW CODE
FF47 00F0 R 5803  DW DUMMY RETURN ; INTERRUPT 1C -- TIMER BREAK ADDR
FF48 00F0 R 5804  DW CODE
FF49 00F0 R 5805  DW CODE
FF4A 00F0 R 5806  DW CODE
FF4B 00F0 R 5807  DW CODE
FF4C 00F0 R 5808  DW CODE
FF4D 00F0 R 5809  DW CODE
FF4E 00000000 5810  DD 0 ; INTERRUPT IF -- POINTER TO VIDEO EXT
FF50 00F0 5811
FF53 00F0 5812  DD 0 ; INTERRUPT IF -- POINTER TO VIDEO EXT

FF54 00F0 5813

FF55 FF26 5814  DW 00000H ; INTERRUPT FOR RETURN FROM KEYBOARD
FF57 FF26 5815  IRET ; DUMMY RETURN FOR RETURN FROM KEYBOARD

 FF58 FF26 5816  ; --- INT 9 ---
 FF59 FF26 5817  ; THIS LOGIC WILL BE INVOKED BY INTERRUPT 9H TO PRINT
 FF5A FF26 5818  ; THE SCREEN. THE CURSOR POSITION AT THE TIME THIS ROUTINE
 FF5B FF26 5819  ; IS INVOKED WILL BE SAVED AND RESTORED UPON COMPLETION. THE
 FF5C FF26 5820  ; ROUTINE IS INTENDED TO RUN WITH INTERRUPTS ENABLED.
 FF5D FF26 5821  ; IF A SUBSEQUENT "PRINT SCREEN KEY IS DEPRESSED DURING THE
 FF5E FF26 5822  ; TIME THIS ROUTINE IS PRINTING IT WILL BE IGNORED.
 FF5F FF26 5823  ; ADDRESS 50:0 CONTAINS THE STATUS OF THE PRINT SCREEN:
 FF60 FF26 5824  ;攀登
 FF61 FF26 5825  ; 50:0 =0 EITHER PRINT SCREEN HAS NOT BEEN CALLED
 FF62 FF26 5826  ; OR UPON RETURN FROM A CALL THIS INDICATES
 FF63 FF26 5827  ; A SUCCESSFUL OPERATION.
 FF64 FF26 5828  ;攀登
 FF65 FF26 5829  ; =1 PRINT SCREEN IS IN PROGRESS
 FF66 FF26 5830  ;攀登
 FF67 FF26 5831  ; =377 ERROR ENCOUNTERED DURING PRINTING
 FF68 FF26 5832  ; ---
 FF69 FF26 5833  ; ASSUME CS:CODE,DS:XXDATA
 FF70 FF26 5834

FF54 8055 5835  PRINT_SCREEN PROC FAR
FF55 8056 5836  STI ; MUST RUN WITH INTERRUPTS ENABLED
FF56 8057 5837  PUSH DS ; MUST USE 50:0 FOR DATA AREA STORAGE
FF57 8058 5838  PUSH AX
FF58 8059 5839  PUSH BX
FF59 805A 5840  PUSH CX ; WILL USE THIS LATER FOR CURSOR LIMITS
FF60 805B 5841  PUSH DX ; WILL HOLD CURRENT CURSOR POSITION
FF61 805C 5842  MOV AX,XXDATA ; MAX 50
FF62 805D 5843  MOV DS,AX
FF63 805E 5844  CMP STATUS_BYTE,1 ; SEE IF PRINT ALREADY IN PROGRESS
FF64 805F 5845  JZ EXIT ; JUMP IF PRINT ALREADY IN PROGRESS
FF65 8060 5846  MOV STATUS_BYTE,1 ; INDICATE PRINT NOW IN PROGRESS
FF66 8061 5847  MOV AH,15 ; WILL REQUEST THE CURRENT SCREEN MODE
FF67 8062 5848  INT 10H ; [AH]=MODE
FF68 8063 5849  IN [AH],NUMBER_COLUMNS/LINE
FF69 8064 5850  [BH]=VISUAL PAGE
FF6A 8065 5851
FF6B 8066 5852  ; AT THIS POINT WE KNOW THE COLUMNS/LINE ARE IN
FF6C 8067 5853  ; [AX] AND THE PAGE IF APPLICABLE IS IN [BH]. THE STACK
FF6D 8068 5854  ; HAS DS,AX,BX,CX,DX PUSHED. [AL] HAS VIDEO MODE
FF6E 8069 5855  ;攀登
FF6F 806A 5856  ; ---

FF70 806B 5857  MOV CL,AX ; WILL MAKE USE OF [AX] REGISTER TO
FF71 806C 5858  MOV CH,CS ; CONTROL ROW & COLUMNS
FF72 806D 5859  CALL CLRF ; CARRIAGE RETURN LINE FEED ROUTINE
FF73 806E 585A  MOV CX ; SAVE SCREEN BOUNDS
FF74 806F 585B  PUSH CX
FF75 8070 585C  MOV AH,3 ; WILL NOW READ THE CURSOR.
FF76 8071 585D  INT 10H ; AND PRESERVE THE POSITION
FF77 8072 585E  POP CX ; RECALL SCREEN BOUNDS
FF78 8073 585F  MOV CX ; RECALL SCREEN BOUNDS
FF79 8074 5860  MOV DX ; RECALL SCREEN BOUNDS
FF7A 8075 5861  XOR DX,DX ; WILL SET CURSOR POSITION TO [0,0]
THE LOOP FROM PRIIO TO THE INSTRUCTION PRIIO IS THE LOOP TO READ EACH CURSOR POSITION FROM THE SCREEN AND PRINT.
Notes For The BIOS Listing

1. The wait loop for the printer times out on form feed of > 51 lines. - line ref (3069)

2. Mode controls for the 320 x 200 video have Color/BW reversed. - line ref (3338)

3. The RS232 Timeout is 80 decimal, not 80 hexadecimal. - line ref (1566)

4. The Base Pointer register is destroyed by some video calls.

5. D_04 character in the character generator has 08 as it’s last value, S/80. - line ref (5511)

6. If you hit print screen in the Color/Graphics 80x25 Character Mode, the screen may not display during the print cycle.
Appendix B. Assembly Instruction Set Reference
### 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>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SP</th>
<th>STACK POINTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>BP</td>
<td>BASE POINTER</td>
</tr>
<tr>
<td>SI</td>
<td>SOURCE INDEX</td>
</tr>
<tr>
<td>DI</td>
<td>DESTINATION INDEX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IP</th>
<th>INSTRUCTION POINTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLAGS</td>
<td>STATUS FLAGS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CS</th>
<th>CODE SEGMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS</td>
<td>DATA SEGMENT</td>
</tr>
<tr>
<td>SS</td>
<td>STACK SEGMENT</td>
</tr>
<tr>
<td>ES</td>
<td>EXTRA SEGMENT</td>
</tr>
</tbody>
</table>

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

```
15 7 0
| X | X | X | X | OF | DF | IF | TF | SF | ZF | X | AF | X | PF | X | CF |
```

X = Don’t Care

<table>
<thead>
<tr>
<th>AF:</th>
<th>AUXILIARY CARRY – BCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>CF:</td>
<td>CARRY FLAG</td>
</tr>
<tr>
<td>PF:</td>
<td>PARITY FLAG</td>
</tr>
<tr>
<td>SF:</td>
<td>SIGN FLAG</td>
</tr>
<tr>
<td>ZF:</td>
<td>ZERO FLAG</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DF:</th>
<th>DIRECTION FLAG (STRINGS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF:</td>
<td>INTERRUPT ENABLE FLAG</td>
</tr>
<tr>
<td>OF:</td>
<td>OVERFLOW FLAG (CF ⊕ SF)</td>
</tr>
<tr>
<td>TF:</td>
<td>TRAP – SINGLE STEP FLAG</td>
</tr>
</tbody>
</table>

---

8080 FLAGS

8088 FLAGS
## OPERAND SUMMARY

"reg" field Bit Assignments:

<table>
<thead>
<tr>
<th></th>
<th>16-Bit (w=1)</th>
<th>8-Bit (w=0)</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>AX</td>
<td>000 AL</td>
<td>00 ES</td>
</tr>
<tr>
<td>001</td>
<td>CX</td>
<td>001 CL</td>
<td>01 CS</td>
</tr>
<tr>
<td>010</td>
<td>DX</td>
<td>010 DL</td>
<td>10 SS</td>
</tr>
<tr>
<td>011</td>
<td>BX</td>
<td>011 BL</td>
<td>11 DS</td>
</tr>
<tr>
<td>100</td>
<td>SP</td>
<td>100 AH</td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>BP</td>
<td>101 CH</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>SI</td>
<td>110 DH</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>DI</td>
<td>111 BH</td>
<td></td>
</tr>
</tbody>
</table>

## SECOND INSTRUCTION BYTE SUMMARY

<table>
<thead>
<tr>
<th>mod</th>
<th>xxxx</th>
<th>r/m</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>disp = 0*, disp-low and disp-high are absent</td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>disp = disp-low sign-extended to 16-bits, disp-high is absent</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>disp = disp-high: disp-low</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>r/m is treated as a &quot;reg&quot; field</td>
<td></td>
</tr>
</tbody>
</table>

### Displacement

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

### Operand Address

<table>
<thead>
<tr>
<th>r/m</th>
<th>Operand Address</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

### Logical Memory Space

- **Offset Address**
- **Selected Segment Register**
  - CS, SS, DS, ES or None for I/O, INT
- **Displacement**
- **Adder**
- **Physical Address Latch**

### Segment Override Prefix

![Segment Override Prefix Diagram]

**0 0 1 reg 1 1 0**

### 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 incl. strings)</td>
<td>DS</td>
<td>ES, SS, or CS</td>
</tr>
<tr>
<td>SI (implicit source addr for strings)</td>
<td>DS</td>
<td>ES, SS, or CS</td>
</tr>
<tr>
<td>DI (implicit dest addr for strings)</td>
<td>ES</td>
<td>Never</td>
</tr>
</tbody>
</table>
DATA TRANSFER

**MOV** = Move
Register/memory to/from register

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

Immediate to register/memory

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

Immediate to register

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

Memory to accumulator

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

Accumulator to memory

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

Register/memory to segment register

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

Segment register to register/memory

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

**PUSH** = Push
Register/memory

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

Register

| 0 1 0 1 0 reg |

Segment register

| 0 0 0 reg 1 1 0 |

**POP** = Pop
Register/memory

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

Register

| 0 1 0 1 1 reg |

Segment register

| 0 0 0 reg 1 1 1 |
**XCHG** = Exchange  
Register/memory with register  
\[
\begin{array}{c|c}
10000011 & \text{mod reg r/m} \\
\end{array}
\]

Register with accumulator  
\[
\begin{array}{c}
10010 \text{ reg} \\
\end{array}
\]

**IN** = Input to AL/AX from  
Fixed port  
\[
\begin{array}{c|c}
1110010 & \text{w port} \\
\end{array}
\]

Variable port (DX)  
\[
\begin{array}{c}
1110110 \text{ w} \\
\end{array}
\]

**OUT** = Output from AL/AX to  
Fixed port  
\[
\begin{array}{c|c}
1110011 & \text{w port} \\
\end{array}
\]

Variable port (DX)  
\[
\begin{array}{c}
1110111 \text{ w} \\
\end{array}
\]

**XLAT** = Translate byte to AL  
\[
\begin{array}{c}
11010111 \\
\end{array}
\]

**LEA** = Load EA to register  
\[
\begin{array}{c|c}
10001101 & \text{mod reg r/m} \\
\end{array}
\]

**LDS** = Load pointer to DS  
\[
\begin{array}{c|c}
11000101 & \text{mod reg r/m} \\
\end{array}
\]

**LES** = Load pointer to ES  
\[
\begin{array}{c|c}
11000100 & \text{mod reg r/m} \\
\end{array}
\]

**LAHF** = Load AH with flags  
\[
\begin{array}{c}
10011111 \\
\end{array}
\]

**SAHF** = Store AH into flags  
\[
\begin{array}{c}
10011110 \\
\end{array}
\]

**PUSHF** = Push flags  
\[
\begin{array}{c}
10011100 \\
\end{array}
\]

**POPF** = Pop flags  
\[
\begin{array}{c}
10011101 \\
\end{array}
\]
### ARITHMETIC

**ADD** = Add

Reg./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

Reg./memory with register to either

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

Immediate to register/memory

| 1 0 0 0 0 0 s w | mod  0 1 0 r/m  | data  | data if s:w=01 |

Immediate to accumulator

| 0 0 0 1 0 1 0 w | data  | data if w=1 |

**INC** = Increment

Register/memory

| 1 1 1 1 1 1 1 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

Reg./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

<p>| 0 0 1 0 1 1 0 w | data  | data if w=1 |</p>
<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Opcode</th>
<th>Mod</th>
<th>Reg</th>
<th>R/M</th>
<th>Data</th>
<th>Data if s:w=01</th>
<th>Data if w=1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SBB</strong></td>
<td>Subtract with borrow Reg./memory and register to either</td>
<td><strong>0 0 0 1 1 0 d w</strong></td>
<td>mod</td>
<td>reg</td>
<td>r/m</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Immediate from register/memory</strong></td>
<td></td>
<td><strong>1 0 0 0 0 0 s w</strong></td>
<td>mod</td>
<td>0 1 1</td>
<td>r/m</td>
<td>data</td>
<td>data if s:w=01</td>
<td></td>
</tr>
<tr>
<td><strong>Immediate from accumulator</strong></td>
<td></td>
<td><strong>0 0 0 1 1 1 0 w</strong></td>
<td>data</td>
<td></td>
<td></td>
<td>data if w=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DEC</strong></td>
<td>Decrement Register/memory</td>
<td><strong>1 1 1 1 1 1 w</strong></td>
<td>mod</td>
<td>0 0 1</td>
<td>r/m</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>NEG</strong></td>
<td>Change sign</td>
<td><strong>1 1 1 1 0 1 1 w</strong></td>
<td>mod</td>
<td>0 1 1</td>
<td>r/m</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CMP</strong></td>
<td>Compare Register/memory and register</td>
<td><strong>0 0 1 1 1 0 d w</strong></td>
<td>mod</td>
<td>reg</td>
<td>r/m</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Immediate with register/memory</strong></td>
<td></td>
<td><strong>1 0 0 0 0 0 s w</strong></td>
<td>mod</td>
<td>1 1 1</td>
<td>r/m</td>
<td>data</td>
<td>data if s:w=01</td>
<td></td>
</tr>
<tr>
<td><strong>Immediate with accumulator</strong></td>
<td></td>
<td><strong>0 0 1 1 1 1 0 w</strong></td>
<td>data</td>
<td></td>
<td></td>
<td>data if w=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>AAS</strong></td>
<td>ASCII adjust for subtract</td>
<td><strong>0 0 1 1 1 1 1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DAS</strong></td>
<td>Decimal adjust for subtract</td>
<td><strong>0 0 1 0 1 1 1 1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>MUL</strong></td>
<td>Multiply (unsigned)</td>
<td><strong>1 1 1 1 0 1 1 w</strong></td>
<td>mod</td>
<td>1 0 0</td>
<td>r/m</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>IMUL</strong></td>
<td>Integer multiply (signed)</td>
<td><strong>1 1 1 1 0 1 1 w</strong></td>
<td>mod</td>
<td>1 0 1</td>
<td>r/m</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
AAM = ASCII adjust for multiply

```
0 0 0 0 1 0 1 0
```

DIV = Divide (unsigned)

```
1 1 1 1 0 1 1 w  mod 1 1 0 r/m
```

IDIV = Integer divide (signed)

```
1 1 1 1 0 1 1 w  mod 1 1 1 r/m
```

AAD = ASCII adjust for divide

```
0 0 0 0 1 0 1 0
```

DIY = Divide (unsigned)

```
1 1 1 1 0 1 1 w  mod 1 1 0 r/m
```

IDLY = Integer divide (signed)

```
1 1 1 1 0 1 1 w  mod 1 1 1 r/m
```

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
Reg./memory and register to either

\[
\begin{array}{cccc}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & d \\
\end{array}
\mod \begin{array}{cccc}
reg & r/m \\
\end{array}
\]

Immediate to register/memory

\[
\begin{array}{cccc}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & w \\
\end{array}
\mod \begin{array}{cccc}
1 & 0 & 0 & r/m \\
data & data & data if w=1 \\
\end{array}
\]

Immediate to accumulator

\[
\begin{array}{cccc}
0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 \\
\end{array}
\mod \begin{array}{cccc}
reg & r/m \\
\end{array}
\]

TEST = And function to flags, no result
Register/memory and register

\[
\begin{array}{cccc}
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
\end{array}
\mod \begin{array}{cccc}
reg & r/m \\
\end{array}
\]

Immediate data and register/memory

\[
\begin{array}{cccc}
1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 \\
\end{array}
\mod \begin{array}{cccc}
0 & 0 & 0 & r/m \\
data & data & data if w=1 \\
\end{array}
\]

Immediate data and accumulator

\[
\begin{array}{cccc}
1 & 0 & 1 & 0 \\
1 & 0 & 0 & w \\
\end{array}
\mod \begin{array}{cccc}
data & data & data if w=1 \\
\end{array}
\]

OR = Or
Reg./memory and register to either

\[
\begin{array}{cccc}
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
\end{array}
\mod \begin{array}{cccc}
reg & r/m \\
\end{array}
\]

Immediate to register/memory

\[
\begin{array}{cccc}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & w \\
\end{array}
\mod \begin{array}{cccc}
0 & 0 & 1 & r/m \\
data & data & data if w=1 \\
\end{array}
\]

Immediate to accumulator

\[
\begin{array}{cccc}
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 \\
\end{array}
\mod \begin{array}{cccc}
data & data & data if w=1 \\
\end{array}
\]

XOR = Exclusive or
Reg./memory and register to either

\[
\begin{array}{cccc}
0 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 \\
\end{array}
\mod \begin{array}{cccc}
reg & r/m \\
\end{array}
\]

Immediate to register/memory

\[
\begin{array}{cccc}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & w \\
\end{array}
\mod \begin{array}{cccc}
1 & 1 & 0 & r/m \\
data & data & data if w=1 \\
\end{array}
\]

Immediate to accumulator

\[
\begin{array}{cccc}
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 \\
\end{array}
\mod \begin{array}{cccc}
data & data & data if w=1 \\
\end{array}
\]

B-10
STRING MANIPULATION

REP = Repeat
1 1 1 1 0 0 1 z

MOVS = Move String
1 0 1 0 0 1 0 w

CMPS = Compare String
1 0 1 0 0 1 1 w

SCAS = Scan String
1 0 1 0 1 1 1 w

LODS = Load String
1 0 1 0 1 1 0 w

STOS = Store String
1 0 1 0 1 0 1 w

CONTROL TRANSFER

CALL = Call
Direct within segment
1 1 1 0 1 0 0 0 disp-low disp-high

Indirect within segment
1 1 1 1 1 1 1 mod 0 1 0 r/m

Direct intersegment
1 0 0 1 1 0 1 0 offset-low offset-high

seg-low seg-high

Indirect intersegment
1 1 1 1 1 1 1 mod 0 1 1 r/m

JMP = Unconditional Jump
Direct within segment
1 1 1 0 1 0 0 1 disp-low disp-high

Direct within segment-short
1 1 1 0 1 0 1 1 disp
Indirect within segment

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

Direct intersegment

```
1 1 1 0 1 0 1 0  offset-low  offset-high
        seg-low  seg-high
```

Indirect intersegment

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

RET = Return from CALL
Within segment

```
1 1 0 0 0 0 1 1
```

Within seg. adding immed to SP

```
1 1 0 0 0 0 1 0  data-low  data-high
```

Intersegment

```
1 1 0 0 1 0 1 1
```

Intersegment, adding immediate to SP

```
1 1 0 0 1 0 1 0  data-low  data-high
```

JE/JZ = Jump on equal/zero

```
0 1 1 1 0 1 0 0  disp
```

JL/JNGE = Jump on less/not greater or equal

```
0 1 1 1 1 1 1 0  disp
```

JLE/JNG = Jump on less or equal/not greater

```
0 1 1 1 1 1 1 0  disp
```

JB/JNAE = Jump on below/not above or equal

```
0 1 1 1 1 0 0 1 0  disp
```

JBE/JNA = Jump on below or equal/not above

```
0 1 1 1 1 0 1 1 0  disp
```

JP/JPE = Jump on parity/parity even

```
0 1 1 1 1 0 1 0  disp
```

JO = Jump on overflow

```
0 1 1 1 1 0 0 0  disp
```
<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Opcode</th>
<th>Disp</th>
</tr>
</thead>
<tbody>
<tr>
<td>JS</td>
<td>Jump on sign</td>
<td>011111000</td>
<td>disp</td>
</tr>
<tr>
<td>JNE/JNZ</td>
<td>Jump on not equal/not zero</td>
<td>011110101</td>
<td>disp</td>
</tr>
<tr>
<td>JNL/JGE</td>
<td>Jump on not less/greater or equal</td>
<td>011111101</td>
<td>disp</td>
</tr>
<tr>
<td>JNLE/JG</td>
<td>Jump on not less or equal/greater</td>
<td>011111111</td>
<td>disp</td>
</tr>
<tr>
<td>JNB/JAE</td>
<td>Jump on not below/above or equal</td>
<td>011110011</td>
<td>disp</td>
</tr>
<tr>
<td>JNBE/JA</td>
<td>Jump on not below or equal/above</td>
<td>011110111</td>
<td>disp</td>
</tr>
<tr>
<td>JNP/JPO</td>
<td>Jump on not parity/parity odd</td>
<td>011111011</td>
<td>disp</td>
</tr>
<tr>
<td>JNO</td>
<td>Jump on not overflow</td>
<td>011110001</td>
<td>disp</td>
</tr>
<tr>
<td>JNS</td>
<td>Jump on not sign</td>
<td>011111001</td>
<td>disp</td>
</tr>
<tr>
<td>LOOP</td>
<td>Loop CX times</td>
<td>11100010</td>
<td>disp</td>
</tr>
<tr>
<td>LOOPZ/LOOPE</td>
<td>Loop while zero/equal</td>
<td>11100001</td>
<td>disp</td>
</tr>
<tr>
<td>LOOPNZ/LOOPNE</td>
<td>Loop while not zero/not equal</td>
<td>11100000</td>
<td>disp</td>
</tr>
<tr>
<td>JCXZ</td>
<td>Jump on CX zero</td>
<td>11100011</td>
<td>disp</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>“equal” or “zero”</td>
</tr>
<tr>
<td>JL or JNGE</td>
<td>(SF xor OF) = 1</td>
<td>“less” or “not greater or equal”</td>
</tr>
<tr>
<td>JLE or JNG</td>
<td>(SP xor OF) or ZF = 1</td>
<td>“less or equal” or “not greater”</td>
</tr>
<tr>
<td>JB or JNAE</td>
<td>CF = 1</td>
<td>“below” or “not above or equal”</td>
</tr>
<tr>
<td>JBE or JNA</td>
<td>(CF or ZF) = 1</td>
<td>“below or equal” or “not above”</td>
</tr>
<tr>
<td>JP or JPE</td>
<td>PF = 1</td>
<td>“parity” or “parity even”</td>
</tr>
<tr>
<td>JO</td>
<td>OF = 1</td>
<td>“overflow”</td>
</tr>
<tr>
<td>JS</td>
<td>SF = 1</td>
<td>“sign”</td>
</tr>
<tr>
<td>JNE or JNZ</td>
<td>ZF = 0</td>
<td>“not equal” or “not zero”</td>
</tr>
<tr>
<td>JNL or JGE</td>
<td>(SF xor OF) = 0</td>
<td>“not less” or “greater or equal”</td>
</tr>
<tr>
<td>JNLE or JG</td>
<td>(SF xor OF) or ZF = 0</td>
<td>“not less or equal” or “greater”</td>
</tr>
<tr>
<td>JNB or JAE</td>
<td>CF = 0</td>
<td>“not below” or “above or equal”</td>
</tr>
<tr>
<td>JNBE or JA</td>
<td>(CF or ZF) = 0</td>
<td>“not below or equal” or “above”</td>
</tr>
<tr>
<td>JNP or JPO</td>
<td>PF = 0</td>
<td>“not parity” or “parity odd”</td>
</tr>
<tr>
<td>JNO</td>
<td>OF = 0</td>
<td>“not overflow”</td>
</tr>
<tr>
<td>JNS</td>
<td>SF = 0</td>
<td>“not sign”</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 1
```

**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>CLC = Clear carry</th>
<th>STC = Set carry</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1 1 1 1 1 1 0 0 0</td>
<td>1 1 1 1 1 0 0 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CMC = Complement carry</th>
<th>NOP = No operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1 1 1 0 1 0 1</td>
<td>1 0 0 1 0 0 0 0</td>
</tr>
</tbody>
</table>
CLD = Clear direction
1 1 1 1 1 1 0 0

STD = Set direction
1 1 1 1 1 1 0 1

CLI = Clear interrupt
1 1 1 1 1 0 1 0

STI = Set interrupt
1 1 1 1 1 0 1 1

HLT = Halt
1 1 1 1 0 1 0 0

WAIT = Wait
1 0 0 1 1 0 1 1

LOCK = Bus lock prefix
1 1 1 1 0 0 0 0

ESC = Escape (to external device)
1 1 0 1 1 x x x r/m

Footnotes:
if d = 1 then "to"; if d = 0 then "from"
if w = 1 then word instruction; if w = 0 then byte instruction
if s:w = 01 then 16 bits of immediate data from the operand
if s:w = 11 then an immediate data byte is sign extended to form the
  16-bit operand
if v = 0 then "count" = 1; if v = 1 then "count" in (CL)
x = don't care
z is used for some string primitives to compare with ZF FLAG
AL = 8-bit accumulator
AX = 16-bit accumulator
CX = Count register
DS = Data segment
DX = Variable port register
ES = Extra segment
Above/below refers to unsigned value
Greater = more positive;
Less = less positive (more negative) signed values
## 8088 INSTRUCTION SET MATRIX

<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>HI</td>
<td>ADD</td>
<td>ADC</td>
<td>AND</td>
<td>XOR</td>
<td>INC</td>
<td>PUSH</td>
<td>AX</td>
<td></td>
</tr>
<tr>
<td></td>
<td>b.f.r/m</td>
<td>b.f.r/m</td>
<td>b.f.r/m</td>
<td>b.f.r/m</td>
<td>AX</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>ADD</td>
<td>ADD</td>
<td>ADD</td>
<td>ADD</td>
<td>ADD</td>
<td>PUSH</td>
<td>ES</td>
<td>POP</td>
</tr>
<tr>
<td></td>
<td>w.f.r/m</td>
<td>b.f.r/m</td>
<td>w.f.r/m</td>
<td>d.f.r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>ES</td>
<td>ES</td>
</tr>
<tr>
<td>1</td>
<td>ADC</td>
<td>ADC</td>
<td>ADC</td>
<td>ADC</td>
<td>ADC</td>
<td>PUSH</td>
<td>SS</td>
<td>POP</td>
</tr>
<tr>
<td></td>
<td>b.f.r/m</td>
<td>w.f.r.m</td>
<td>w.f.r/m</td>
<td>w.f.r/m</td>
<td>d.i</td>
<td>d.wi</td>
<td>SS</td>
<td>SS</td>
</tr>
<tr>
<td>2</td>
<td>AND</td>
<td>AND</td>
<td>AND</td>
<td>AND</td>
<td>AND</td>
<td>SEG</td>
<td>ES</td>
<td>AAA</td>
</tr>
<tr>
<td></td>
<td>b.f.r/m</td>
<td>w.f.r/m</td>
<td>w.f.r/m</td>
<td>w.f.r/m</td>
<td>d.i</td>
<td>d.wi</td>
<td>ES</td>
<td>AAA</td>
</tr>
<tr>
<td>3</td>
<td>XOR</td>
<td>XOR</td>
<td>XOR</td>
<td>XOR</td>
<td>XOR</td>
<td>SEG</td>
<td>ES</td>
<td>AAA</td>
</tr>
<tr>
<td></td>
<td>w.f.r/m</td>
<td>w.f.r/m</td>
<td>w.f.r/m</td>
<td>w.f.r/m</td>
<td>d.i</td>
<td>d.wi</td>
<td>ES</td>
<td>AAA</td>
</tr>
<tr>
<td>4</td>
<td>INC</td>
<td>INC</td>
<td>INC</td>
<td>INC</td>
<td>INC</td>
<td>INC</td>
<td>INC</td>
<td>INC</td>
</tr>
<tr>
<td></td>
<td>AX</td>
<td>CX</td>
<td>DX</td>
<td>BX</td>
<td>SP</td>
<td>BP</td>
<td>SI</td>
<td>DI</td>
</tr>
<tr>
<td>5</td>
<td>PUSH</td>
<td>PUSH</td>
<td>PUSH</td>
<td>PUSH</td>
<td>PUSH</td>
<td>PUSH</td>
<td>PUSH</td>
<td>PUSH</td>
</tr>
<tr>
<td></td>
<td>AX</td>
<td>CX</td>
<td>DX</td>
<td>BX</td>
<td>SP</td>
<td>BP</td>
<td>SI</td>
<td>DI</td>
</tr>
</tbody>
</table>

### Notes:
- **b** = byte operation
- **d** = direct
- **f** = from CPU reg
- **i** = immediate
- **ia** = immed. to accum.
- **id** = indirect
- **is** = immed. byte, sign ext.
- **l** = long ie. intersegment
- **m** = memory
- **r/m** = EA is second byte
- **si** = short intrasegment
- **sr** = segment register
- **t** = to CPU reg
- **v** = variable
- **w** = word operation
- **z** = zero
### 8088 Instruction Set Matrix

<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>0</td>
<td>OR</td>
<td>OR</td>
<td>OR</td>
<td>OR</td>
<td>OR</td>
<td>OR</td>
<td>OR</td>
<td>PUSH</td>
<td>CS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>b.f/r/m</td>
<td>w.f/r/m</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
</tr>
<tr>
<td>1</td>
<td>SBB</td>
<td>SBB</td>
<td>SBB</td>
<td>SBB</td>
<td>SBB</td>
<td>SBB</td>
<td>SBB</td>
<td>PUSH</td>
<td>POP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>b.f/r/m</td>
<td>w.f/r/m</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
</tr>
<tr>
<td>2</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SUB</td>
<td>SEG</td>
<td>DAS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>b.f/r/m</td>
<td>w.f/r/m</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
</tr>
<tr>
<td>3</td>
<td>CMP</td>
<td>CMP</td>
<td>CMP</td>
<td>CMP</td>
<td>CMP</td>
<td>CMP</td>
<td>CMP</td>
<td>CS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>b.f/r/m</td>
<td>w.f/r/m</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
<td>w.i</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.i</td>
</tr>
<tr>
<td>4</td>
<td>DEC</td>
<td>DEC</td>
<td>DEC</td>
<td>DEC</td>
<td>DEC</td>
<td>DEC</td>
<td>DEC</td>
<td>BP</td>
<td>SI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AX</td>
<td>CX</td>
<td>DX</td>
<td>BX</td>
<td>SP</td>
<td>SI</td>
<td>DI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>POP</td>
<td>POP</td>
<td>POP</td>
<td>POP</td>
<td>POP</td>
<td>POP</td>
<td>POP</td>
<td>BP</td>
<td>SI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AX</td>
<td>CX</td>
<td>DX</td>
<td>BX</td>
<td>SP</td>
<td>SI</td>
<td>DI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</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>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>JS</td>
<td>JS</td>
<td>JNS</td>
<td>JNS</td>
<td>JNS</td>
<td>JNS</td>
<td>JNS</td>
<td>JNS</td>
<td>JNS</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>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>b.f/r/m</td>
<td>w.f/r/m</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.t/r/m</td>
<td>w.t/r/m</td>
<td>b.t/r/m</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>CBW</td>
<td>CBW</td>
<td>CALL</td>
<td>WAIT</td>
<td>PUSHF</td>
<td>POPF</td>
<td>SAHF</td>
<td>LAHF</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>l.d</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>A</td>
<td>TEST</td>
<td>TEST</td>
<td>STOS</td>
<td>STOS</td>
<td>STOS</td>
<td>STOS</td>
<td>STOS</td>
<td>STOS</td>
<td>SCAS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>b.i</td>
<td>w.i</td>
<td>b</td>
<td>w</td>
<td>b</td>
<td>w</td>
<td>b</td>
<td>w</td>
<td>b</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td>MOV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>i→AX</td>
<td>i→CX</td>
<td>i→DX</td>
<td>i→BX</td>
<td>i→SP</td>
<td>i→BP</td>
<td>i→SI</td>
<td>MOV</td>
<td>MOV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>RET</td>
<td>RET</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>INTO</td>
<td>IRET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>l,(l+i+SP)</td>
<td></td>
<td>l</td>
<td>Type 3</td>
<td>INT (Any)</td>
<td>INTO</td>
<td>IRET</td>
<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>ESC</td>
<td>ESC</td>
<td>ESC</td>
<td>ESC</td>
<td>ESC</td>
<td>ESC</td>
<td>ESC</td>
<td>ESC</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>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>CALL</td>
<td>JMP</td>
<td>JMP</td>
<td>JMP</td>
<td>IN</td>
<td>IN</td>
<td>OUT</td>
<td>OUT</td>
<td>v.b</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>d</td>
<td>d</td>
<td>l.d</td>
<td>si.d</td>
<td>v.b</td>
<td>v.w</td>
<td>v.b</td>
<td>v.w</td>
<td>v.w</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>CLC</td>
<td>STC</td>
<td>CLI</td>
<td>STI</td>
<td>CLD</td>
<td>STD</td>
<td>Grp 2</td>
<td>Grp 2</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>b.t/r/m</td>
<td>w.t/r/m</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table: where

<table>
<thead>
<tr>
<th>mod r/m</th>
<th>000</th>
<th>001</th>
<th>010</th>
<th>011</th>
<th>100</th>
<th>101</th>
<th>100</th>
<th>111</th>
</tr>
</thead>
<tbody>
<tr>
<td>Immed</td>
<td>ADD</td>
<td>OR</td>
<td>ADC</td>
<td>SBB</td>
<td>AND</td>
<td>SUB</td>
<td>XOR</td>
<td>CMP</td>
</tr>
<tr>
<td>Shift</td>
<td>ROL</td>
<td>ROR</td>
<td>RCL</td>
<td>RCR</td>
<td>SHL/SAL</td>
<td>SHR</td>
<td>SAR</td>
<td></td>
</tr>
<tr>
<td>Grp 1</td>
<td>TEST</td>
<td>NOT</td>
<td>NEG</td>
<td>MUL</td>
<td>IMUL</td>
<td>DIV</td>
<td>IDIV</td>
<td></td>
</tr>
<tr>
<td>Grp 2</td>
<td>INC</td>
<td>DEC</td>
<td>CALL</td>
<td>CALL</td>
<td>JMP</td>
<td>JMP</td>
<td>PUSH</td>
<td></td>
</tr>
</tbody>
</table>

B-17
## INSTRUCTION SET INDEX

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA</td>
<td>6</td>
</tr>
<tr>
<td>AAD</td>
<td>8</td>
</tr>
<tr>
<td>AAM</td>
<td>8</td>
</tr>
<tr>
<td>AAS</td>
<td>7</td>
</tr>
<tr>
<td>ADC</td>
<td>6</td>
</tr>
<tr>
<td>ADD</td>
<td>6</td>
</tr>
<tr>
<td>AND</td>
<td>9</td>
</tr>
<tr>
<td>CALL</td>
<td>10</td>
</tr>
<tr>
<td>CBW</td>
<td>8</td>
</tr>
<tr>
<td>CLC</td>
<td>13</td>
</tr>
<tr>
<td>CLD</td>
<td>14</td>
</tr>
<tr>
<td>CLI</td>
<td>14</td>
</tr>
<tr>
<td>CMC</td>
<td>13</td>
</tr>
<tr>
<td>CMP</td>
<td>7</td>
</tr>
<tr>
<td>CMPS</td>
<td>10</td>
</tr>
<tr>
<td>CWD</td>
<td>8</td>
</tr>
<tr>
<td>DAA</td>
<td>6</td>
</tr>
<tr>
<td>DAS</td>
<td>7</td>
</tr>
<tr>
<td>DEC</td>
<td>7</td>
</tr>
<tr>
<td>DIV</td>
<td>8</td>
</tr>
<tr>
<td>ESC</td>
<td>14</td>
</tr>
<tr>
<td>HLT</td>
<td>14</td>
</tr>
<tr>
<td>IDIV</td>
<td>8</td>
</tr>
<tr>
<td>IMUL</td>
<td>7</td>
</tr>
<tr>
<td>IN</td>
<td>5</td>
</tr>
<tr>
<td>INC</td>
<td>6</td>
</tr>
<tr>
<td>INT</td>
<td>13</td>
</tr>
<tr>
<td>INTO</td>
<td>13</td>
</tr>
<tr>
<td>IRET</td>
<td>13</td>
</tr>
<tr>
<td>JA</td>
<td>12</td>
</tr>
<tr>
<td>JAE</td>
<td>12</td>
</tr>
<tr>
<td>JB</td>
<td>11</td>
</tr>
<tr>
<td>JBE</td>
<td>11</td>
</tr>
<tr>
<td>JCXZ</td>
<td>12</td>
</tr>
<tr>
<td>JE</td>
<td>11</td>
</tr>
<tr>
<td>JG</td>
<td>12</td>
</tr>
<tr>
<td>JGE</td>
<td>12</td>
</tr>
<tr>
<td>JL</td>
<td>11</td>
</tr>
<tr>
<td>JLE</td>
<td>11</td>
</tr>
<tr>
<td>JMP</td>
<td>10</td>
</tr>
<tr>
<td>JNA</td>
<td>11</td>
</tr>
<tr>
<td>JNBE</td>
<td>12</td>
</tr>
<tr>
<td>JNE</td>
<td>12</td>
</tr>
<tr>
<td>JNG</td>
<td>11</td>
</tr>
<tr>
<td>JNLE</td>
<td>12</td>
</tr>
<tr>
<td>JNL</td>
<td>12</td>
</tr>
<tr>
<td>JNS</td>
<td>12</td>
</tr>
<tr>
<td>JNZ</td>
<td>12</td>
</tr>
<tr>
<td>JP</td>
<td>11</td>
</tr>
<tr>
<td>JPE</td>
<td>11</td>
</tr>
<tr>
<td>JPO</td>
<td>12</td>
</tr>
<tr>
<td>JS</td>
<td>12</td>
</tr>
<tr>
<td>JZ</td>
<td>11</td>
</tr>
<tr>
<td>LAHF</td>
<td>5</td>
</tr>
<tr>
<td>LDS</td>
<td>5</td>
</tr>
<tr>
<td>LEA</td>
<td>5</td>
</tr>
<tr>
<td>LES</td>
<td>5</td>
</tr>
<tr>
<td>LOCK</td>
<td>14</td>
</tr>
<tr>
<td>LOOS</td>
<td>10</td>
</tr>
<tr>
<td>LOOP</td>
<td>12</td>
</tr>
<tr>
<td>LOOPNE</td>
<td>12</td>
</tr>
<tr>
<td>LOOPNZ</td>
<td>12</td>
</tr>
<tr>
<td>LOOPZ</td>
<td>12</td>
</tr>
<tr>
<td>MOV</td>
<td>4</td>
</tr>
<tr>
<td>MOVX</td>
<td>10</td>
</tr>
<tr>
<td>MUL</td>
<td>7</td>
</tr>
<tr>
<td>NEG</td>
<td>7</td>
</tr>
<tr>
<td>NOP</td>
<td>13</td>
</tr>
<tr>
<td>NOT</td>
<td>8</td>
</tr>
<tr>
<td>OR</td>
<td>9</td>
</tr>
<tr>
<td>OUT</td>
<td>5</td>
</tr>
<tr>
<td>POP</td>
<td>4</td>
</tr>
<tr>
<td>POPF</td>
<td>5</td>
</tr>
<tr>
<td>PUSH</td>
<td>4</td>
</tr>
<tr>
<td>PUSHF</td>
<td>5</td>
</tr>
<tr>
<td>RCL</td>
<td>8</td>
</tr>
<tr>
<td>RCR</td>
<td>8</td>
</tr>
<tr>
<td>REP</td>
<td>10</td>
</tr>
<tr>
<td>RET</td>
<td>11</td>
</tr>
<tr>
<td>ROL</td>
<td>8</td>
</tr>
<tr>
<td>ROR</td>
<td>8</td>
</tr>
<tr>
<td>SAHF</td>
<td>5</td>
</tr>
<tr>
<td>SAL</td>
<td>8</td>
</tr>
<tr>
<td>SAR</td>
<td>8</td>
</tr>
<tr>
<td>SBB</td>
<td>7</td>
</tr>
<tr>
<td>SCAS</td>
<td>10</td>
</tr>
<tr>
<td>SHL</td>
<td>8</td>
</tr>
<tr>
<td>SHR</td>
<td>8</td>
</tr>
<tr>
<td>STC</td>
<td>13</td>
</tr>
<tr>
<td>STD</td>
<td>14</td>
</tr>
<tr>
<td>STI</td>
<td>14</td>
</tr>
<tr>
<td>STOS</td>
<td>10</td>
</tr>
<tr>
<td>SUB</td>
<td>6</td>
</tr>
<tr>
<td>TEST</td>
<td>9</td>
</tr>
<tr>
<td>WAIT</td>
<td>14</td>
</tr>
<tr>
<td>XCHG</td>
<td>5</td>
</tr>
<tr>
<td>XLAT</td>
<td>5</td>
</tr>
<tr>
<td>XOR</td>
<td>9</td>
</tr>
</tbody>
</table>
## 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><strong>AS TEXT ATTRIBUTES</strong></th>
<th><strong>COLOR/GRAPHICS MONITOR ADAPTER</strong></th>
<th><strong>IBM MONOCROME DISPLAY ADAPTER</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
<td>BLANK (NULL)</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 ↓</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, SHIFT</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>BLUE</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>
<tr>
<td>HEX DEC</td>
<td>AS CHARACTERS</td>
<td>AS TEXT ATTRIBUTES</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>
</tr>
<tr>
<td></td>
<td>COLOR/GRAFICS</td>
<td>IBM MONOCHROME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MONITOR ADAPTER</td>
<td>DISPLAY ADAPTER</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BACKGROUND</td>
<td>FOREGROUND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18 24</td>
<td>↑ CTRL X</td>
<td>BLUE</td>
<td>DARK GREY</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19 25</td>
<td>↓ CTRL Y</td>
<td>BLUE</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1A 26</td>
<td>← CTRL Z</td>
<td>BLUE</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1B 27</td>
<td>← CTRL [, ESC, SHIFT ESC, CTRL ESC</td>
<td>BLUE</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1C 28</td>
<td>← CTRL \</td>
<td>BLUE</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1D 29</td>
<td>← CTRL ]</td>
<td>BLUE</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1E 30</td>
<td>▲ CTRL 6</td>
<td>BLUE</td>
<td>YELLLOW</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1F 31</td>
<td>▼ CTRL ←</td>
<td>BLUE</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 32</td>
<td>BLANK (SPACE</td>
<td>SPACE BAR, SHIFT SPACE, CTRL SPACE, ALT SPACE</td>
<td>GREEN</td>
<td>BLACK</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21 33</td>
<td>! !</td>
<td>SHIFT</td>
<td>GREEN</td>
<td>BLUE</td>
<td>UNDERLINE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22 34</td>
<td>&quot; &quot;</td>
<td>SHIFT</td>
<td>GREEN</td>
<td>GREEN</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23 35</td>
<td># #</td>
<td>SHIFT</td>
<td>GREEN</td>
<td>CYAN</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24 36</td>
<td>$ $</td>
<td>SHIFT</td>
<td>GREEN</td>
<td>RED</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25 37</td>
<td>% %</td>
<td>SHIFT</td>
<td>GREEN</td>
<td>MAGENTA</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26 38</td>
<td>&amp; &amp;</td>
<td>SHIFT</td>
<td>GREEN</td>
<td>BROWN</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27 39</td>
<td>' '</td>
<td>SHIFT</td>
<td>GREEN</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28 40</td>
<td>( (</td>
<td>SHIFT</td>
<td>GREEN</td>
<td>DARK GREY</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29 41</td>
<td>) )</td>
<td>SHIFT</td>
<td>GREEN</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2A 42</td>
<td>* *</td>
<td>NOTE 1</td>
<td>GREEN</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2B 43</td>
<td>+ +</td>
<td>SHIFT</td>
<td>GREEN</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2C 44</td>
<td>. .</td>
<td>GREEN</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2D 45</td>
<td>− −</td>
<td>GREEN</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2E 46</td>
<td>. .</td>
<td>NOTE 2</td>
<td>GREEN</td>
<td>YELLOW</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALUE</td>
<td>AS CHARACTERS</td>
<td>AS TEXT ATTRIBUTES</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>
</tr>
<tr>
<td>2F 47</td>
<td>/ /</td>
<td>COLOR/GRAPHICS MONITOR ADAPTER&lt;br&gt;BACKGROUND: GREEN&lt;br&gt;FOREGROUND: WHITE&lt;br&gt;IBM MONochrome DISPLAY ADAPTER: HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30 48</td>
<td>0 0</td>
<td>NOTE 3&lt;br&gt;CYAN&lt;br&gt;BLACK&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31 49</td>
<td>1 1</td>
<td>NOTE 3&lt;br&gt;CYAN&lt;br&gt;BLUE&lt;br&gt;UNDERLINE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32 50</td>
<td>2 2</td>
<td>NOTE 3&lt;br&gt;CYAN&lt;br&gt;GREEN&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33 51</td>
<td>3 3</td>
<td>NOTE 3&lt;br&gt;CYAN&lt;br&gt;CYAN&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>34 52</td>
<td>4 4</td>
<td>NOTE 3&lt;br&gt;CYAN&lt;br&gt;RED&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35 53</td>
<td>5 5</td>
<td>NOTE 3&lt;br&gt;CYAN&lt;br&gt;MAGENTA&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>36 54</td>
<td>6 6</td>
<td>NOTE 3&lt;br&gt;CYAN&lt;br&gt;BROWN&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>37 55</td>
<td>7 7</td>
<td>NOTE 3&lt;br&gt;CYAN&lt;br&gt;LIGHT GREY&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38 56</td>
<td>8 8</td>
<td>NOTE 3&lt;br&gt;CYAN&lt;br&gt;DARK GREY&lt;br&gt;HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39 57</td>
<td>9 9</td>
<td>NOTE 3&lt;br&gt;CYAN&lt;br&gt;LIGHT BLUE&lt;br&gt;HIGH INTENSITY UNDERLINE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3A 58</td>
<td>: :</td>
<td>SHIFT&lt;br&gt;CYAN&lt;br&gt;LIGHT GREEN&lt;br&gt;HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3B 59</td>
<td>: :</td>
<td>CYAN&lt;br&gt;LIGHT CYAN&lt;br&gt;HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3C 60</td>
<td>&lt; &lt;</td>
<td>SHIFT&lt;br&gt;CYAN&lt;br&gt;LIGHT RED&lt;br&gt;HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3D 61</td>
<td>= =</td>
<td>CYAN&lt;br&gt;LIGHT MAGENTA&lt;br&gt;HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3E 62</td>
<td>&gt; &gt;</td>
<td>SHIFT&lt;br&gt;CYAN&lt;br&gt;YELLOW&lt;br&gt;HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3F 63</td>
<td>? ?</td>
<td>SHIFT&lt;br&gt;CYAN&lt;br&gt;WHITE&lt;br&gt;HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40 64</td>
<td>@ @</td>
<td>SHIFT&lt;br&gt;RED&lt;br&gt;BLACK&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41 65</td>
<td>A A</td>
<td>NOTE 4&lt;br&gt;RED&lt;br&gt;BLUE&lt;br&gt;UNDERLINE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>42 66</td>
<td>B B</td>
<td>NOTE 4&lt;br&gt;RED&lt;br&gt;GREEN&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43 67</td>
<td>C C</td>
<td>NOTE 4&lt;br&gt;RED&lt;br&gt;CYAN&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>44 68</td>
<td>D D</td>
<td>NOTE 4&lt;br&gt;RED&lt;br&gt;RED&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45 69</td>
<td>E E</td>
<td>NOTE 4&lt;br&gt;RED&lt;br&gt;MAGENTA&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46 70</td>
<td>F F</td>
<td>NOTE 4&lt;br&gt;RED&lt;br&gt;BROWN&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>47 71</td>
<td>G G</td>
<td>NOTE 4&lt;br&gt;RED&lt;br&gt;LIGHT GREY&lt;br&gt;NORMAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48 72</td>
<td>H H</td>
<td>NOTE 4&lt;br&gt;RED&lt;br&gt;DARK GREY&lt;br&gt;HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49 73</td>
<td>I I</td>
<td>NOTE 4&lt;br&gt;RED&lt;br&gt;LIGHT BLUE&lt;br&gt;HIGH INTENSITY UNDERLINE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4A 74</td>
<td>J J</td>
<td>NOTE 4&lt;br&gt;RED&lt;br&gt;LIGHT GREEN&lt;br&gt;HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEX</td>
<td>DEC</td>
<td>SYMBOL</td>
<td>KEYSTrokes</td>
<td>MODES</td>
<td>BACKGROUND</td>
<td>FOREGROUND</td>
<td>COLOR/GRAphICS MONITOR ADAPTER</td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>--------</td>
<td>------------</td>
<td>-------</td>
<td>------------</td>
<td>------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>4B</td>
<td>75</td>
<td>K</td>
<td>K</td>
<td>NOTE 4</td>
<td>RED</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>4C</td>
<td>76</td>
<td>L</td>
<td>L</td>
<td>NOTE 4</td>
<td>RED</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>4D</td>
<td>77</td>
<td>M</td>
<td>M</td>
<td>NOTE 4</td>
<td>RED</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>4E</td>
<td>78</td>
<td>N</td>
<td>N</td>
<td>NOTE 4</td>
<td>RED</td>
<td>YELLOW</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>4F</td>
<td>79</td>
<td>O</td>
<td>O</td>
<td>NOTE 4</td>
<td>RED</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>50</td>
<td>80</td>
<td>P</td>
<td>P</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>BLACK</td>
<td>NORMAL</td>
</tr>
<tr>
<td>51</td>
<td>81</td>
<td>Q</td>
<td>Q</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>BLUE</td>
<td>UNDERLINE</td>
</tr>
<tr>
<td>52</td>
<td>82</td>
<td>R</td>
<td>R</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>GREEN</td>
<td>NORMAL</td>
</tr>
<tr>
<td>53</td>
<td>83</td>
<td>S</td>
<td>S</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>CYAN</td>
<td>NORMAL</td>
</tr>
<tr>
<td>54</td>
<td>84</td>
<td>T</td>
<td>T</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>RED</td>
<td>NORMAL</td>
</tr>
<tr>
<td>55</td>
<td>85</td>
<td>U</td>
<td>U</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>MAGENTA</td>
<td>NORMAL</td>
</tr>
<tr>
<td>56</td>
<td>86</td>
<td>V</td>
<td>V</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>BROWN</td>
<td>NORMAL</td>
</tr>
<tr>
<td>57</td>
<td>87</td>
<td>W</td>
<td>W</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
</tr>
<tr>
<td>58</td>
<td>88</td>
<td>X</td>
<td>X</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>DARK GREY</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>59</td>
<td>89</td>
<td>Y</td>
<td>Y</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
</tr>
<tr>
<td>5A</td>
<td>90</td>
<td>Z</td>
<td>Z</td>
<td>NOTE 4</td>
<td>MAGENTA</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>5B</td>
<td>91</td>
<td>[</td>
<td>[</td>
<td></td>
<td>MAGENTA</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>5C</td>
<td>92</td>
<td>\</td>
<td>\</td>
<td></td>
<td>MAGENTA</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>5D</td>
<td>93</td>
<td>]</td>
<td>]</td>
<td></td>
<td>MAGENTA</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>5E</td>
<td>94</td>
<td>▲</td>
<td>▲</td>
<td></td>
<td>MAGENTA</td>
<td>YELLOW</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>5F</td>
<td>95</td>
<td>—</td>
<td>—</td>
<td></td>
<td>MAGENTA</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>60</td>
<td>96</td>
<td>‘</td>
<td>‘</td>
<td></td>
<td>YELLOW</td>
<td>BLACK</td>
<td>NORMAL</td>
</tr>
<tr>
<td>61</td>
<td>97</td>
<td>a</td>
<td>a</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>BLUE</td>
<td>UNDERLINE</td>
</tr>
<tr>
<td>62</td>
<td>98</td>
<td>b</td>
<td>b</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>GREEN</td>
<td>NORMAL</td>
</tr>
<tr>
<td>63</td>
<td>99</td>
<td>c</td>
<td>c</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>CYAN</td>
<td>NORMAL</td>
</tr>
<tr>
<td>64</td>
<td>100</td>
<td>d</td>
<td>d</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>RED</td>
<td>NORMAL</td>
</tr>
<tr>
<td>65</td>
<td>101</td>
<td>e</td>
<td>e</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>MAGENTA</td>
<td>NORMAL</td>
</tr>
<tr>
<td>66</td>
<td>102</td>
<td>f</td>
<td>f</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>BROWN</td>
<td>NORMAL</td>
</tr>
<tr>
<td>VALUE</td>
<td>AS CHARACTERS</td>
<td>AS TEXT ATTRIBUTES</td>
<td>COLOR/GRAPHICS MONITOR ADAPTER</td>
<td>IBM MONOCHROME DISPLAY ADAPTER</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>
</tr>
<tr>
<td></td>
<td></td>
<td>NORM</td>
<td>wert</td>
<td>BACKGROUND</td>
<td>FOREGROUND</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>67 103</td>
<td>g g</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>68 104</td>
<td>h h</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>DARK GREY</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>69 105</td>
<td>i i</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6A 106</td>
<td>i i</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6B 107</td>
<td>k k</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6C 108</td>
<td>l l</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6D 109</td>
<td>m m</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6E 110</td>
<td>n n</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>YELLOW</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6F 111</td>
<td>o o</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>70 112</td>
<td>p p</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>BLACK</td>
<td>REVERSE VIDEO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>71 113</td>
<td>q q</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>BLUE</td>
<td>UNDERLINE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>72 114</td>
<td>r r</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>GREEN</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>73 115</td>
<td>s s</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>CYAN</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>74 116</td>
<td>t t</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>RED</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75 117</td>
<td>u u</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>MAGENTA</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>76 118</td>
<td>v v</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>BROWN</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>77 119</td>
<td>w w</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>78 120</td>
<td>x x</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>DARK GREY</td>
<td>REVERSE VIDEO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>79 121</td>
<td>y y</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A 122</td>
<td>z z</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B 123</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SHIFT</td>
<td>WHITE</td>
<td>LIGHT CYAN</td>
</tr>
<tr>
<td>C 124</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SHIFT</td>
<td>WHITE</td>
<td>LIGHT RED</td>
</tr>
<tr>
<td>D 125</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SHIFT</td>
<td>WHITE</td>
<td>LIGHT MAGENTA</td>
</tr>
<tr>
<td>E 126</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SHIFT</td>
<td>WHITE</td>
<td>YELLOW</td>
</tr>
<tr>
<td>F 127</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALUE</td>
<td>AS CHARACTERS</td>
<td>AS TEXT ATTRIBUTES</td>
<td>COLOR/GRAPHICS</td>
<td>IBM MONOCROME</td>
<td>DISPLAY ADAPTER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>---------------</td>
<td>------------------</td>
<td>---------------</td>
<td>---------------</td>
<td>-----------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEX</td>
<td>DEC</td>
<td>SYMBOL</td>
<td>KEYSTROKES</td>
<td>MODES</td>
<td>BACKGROUND</td>
<td>FOREGROUND</td>
<td>MONITOR ADAPTER</td>
</tr>
<tr>
<td>80 128</td>
<td>C</td>
<td>ALT 128</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>BLACK</td>
<td>NON-DISPLAY</td>
<td></td>
</tr>
<tr>
<td>81 129</td>
<td>ü</td>
<td>ALT 129</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>BLUE</td>
<td>UNDERLINE</td>
<td></td>
</tr>
<tr>
<td>82 130</td>
<td>ö</td>
<td>ALT 130</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>GREEN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>83 131</td>
<td>å</td>
<td>ALT 131</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>CYAN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>84 132</td>
<td>ä</td>
<td>ALT 132</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>RED</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>85 133</td>
<td>ä</td>
<td>ALT 133</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>MAGENTA</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>86 134</td>
<td>å</td>
<td>ALT 134</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>BROWN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>87 135</td>
<td>å</td>
<td>ALT 135</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>88 136</td>
<td>å</td>
<td>ALT 136</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>DARK GREY</td>
<td>NON-DISPLAY</td>
<td></td>
</tr>
<tr>
<td>89 137</td>
<td>ö</td>
<td>ALT 137</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINED</td>
<td></td>
</tr>
<tr>
<td>8A 138</td>
<td>ö</td>
<td>ALT 138</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>8B 139</td>
<td>i</td>
<td>ALT 139</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>8C 140</td>
<td>i</td>
<td>ALT 140</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>8D 141</td>
<td>i</td>
<td>ALT 141</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>8E 142</td>
<td>Ä</td>
<td>ALT 142</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>YELLOW</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>8F 143</td>
<td>Ä</td>
<td>ALT 143</td>
<td>NOTE 6</td>
<td>BLACK</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>90 144</td>
<td>E</td>
<td>ALT 144</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>BLACK</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>91 145</td>
<td>äc</td>
<td>ALT 145</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>BLUE</td>
<td>UNDERLINED</td>
<td></td>
</tr>
<tr>
<td>92 146</td>
<td>FE</td>
<td>ALT 146</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>GREEN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>93 147</td>
<td>ö</td>
<td>ALT 147</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>CYAN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>94 148</td>
<td>ö</td>
<td>ALT 148</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>RED</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>95 149</td>
<td>ö</td>
<td>ALT 149</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>MAGENTA</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>96 150</td>
<td>ö</td>
<td>ALT 150</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>BROWN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>97 151</td>
<td>ö</td>
<td>ALT 151</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>98 152</td>
<td>ö</td>
<td>ALT 152</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>DARK GREY</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>99 153</td>
<td>ö</td>
<td>ALT 153</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINED</td>
<td></td>
</tr>
<tr>
<td>9A 154</td>
<td>ü</td>
<td>ALT 154</td>
<td>NOTE 6</td>
<td>BLUE</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>VALUE</td>
<td>CHARS</td>
<td>KEYS</td>
<td>MODES</td>
<td>COLOR/GRAPHICS</td>
<td>IBM ADAPTER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>------</td>
<td>-------</td>
<td>----------------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9B</td>
<td>155</td>
<td>$^5$</td>
<td>ALT 155</td>
<td>NOTE 6</td>
<td>BLUE LIGHT</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>9C</td>
<td>156</td>
<td>£</td>
<td>ALT 156</td>
<td>NOTE 6</td>
<td>BLUE LIGHT</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>9D</td>
<td>157</td>
<td>∨</td>
<td>ALT 157</td>
<td>NOTE 6</td>
<td>BLUE LIGHT</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>9E</td>
<td>158</td>
<td>pts</td>
<td>ALT 158</td>
<td>NOTE 6</td>
<td>BLUE YELLOW</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>9F</td>
<td>159</td>
<td>ƒ</td>
<td>ALT 159</td>
<td>NOTE 6</td>
<td>BLUE WHITE</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>A0</td>
<td>160</td>
<td>ø</td>
<td>ALT 160</td>
<td>NOTE 6</td>
<td>GREEN BLACK</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>A1</td>
<td>161</td>
<td>å</td>
<td>ALT 161</td>
<td>NOTE 6</td>
<td>GREEN BLUE</td>
<td>UNDERLINE</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>162</td>
<td>ø</td>
<td>ALT 162</td>
<td>NOTE 6</td>
<td>GREEN GREEN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>A3</td>
<td>163</td>
<td>ø</td>
<td>ALT 163</td>
<td>NOTE 6</td>
<td>GREEN CYAN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>A4</td>
<td>164</td>
<td>ø</td>
<td>ALT 164</td>
<td>NOTE 6</td>
<td>GREEN RED</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>A5</td>
<td>165</td>
<td>ñ</td>
<td>ALT 165</td>
<td>NOTE 6</td>
<td>GREEN MAGENTA</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>A6</td>
<td>166</td>
<td>ø</td>
<td>ALT 166</td>
<td>NOTE 6</td>
<td>GREEN BROWN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>A7</td>
<td>167</td>
<td>ø</td>
<td>ALT 167</td>
<td>NOTE 6</td>
<td>GREEN LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>A8</td>
<td>168</td>
<td>ø</td>
<td>ALT 168</td>
<td>NOTE 6</td>
<td>GREEN DARK GREY</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>A9</td>
<td>169</td>
<td>ø</td>
<td>ALT 169</td>
<td>NOTE 6</td>
<td>GREEN LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
<td></td>
</tr>
<tr>
<td>AA</td>
<td>170</td>
<td>ø</td>
<td>ALT 170</td>
<td>NOTE 6</td>
<td>GREEN LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>AB</td>
<td>171</td>
<td>ø</td>
<td>ALT 171</td>
<td>NOTE 6</td>
<td>GREEN LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>AC</td>
<td>172</td>
<td>ø</td>
<td>ALT 172</td>
<td>NOTE 6</td>
<td>GREEN LIGHT RED</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>AD</td>
<td>173</td>
<td>ø</td>
<td>ALT 173</td>
<td>NOTE 6</td>
<td>GREEN LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>AE</td>
<td>174</td>
<td>&lt;</td>
<td>ALT 174</td>
<td>NOTE 6</td>
<td>GREEN YELLOW</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>AF</td>
<td>175</td>
<td>&gt;</td>
<td>ALT 175</td>
<td>NOTE 6</td>
<td>GREEN WHITE</td>
<td>HIGH INTENSITY</td>
<td></td>
</tr>
<tr>
<td>B0</td>
<td>176</td>
<td>ø</td>
<td>ALT 176</td>
<td>NOTE 6</td>
<td>CYAN BLACK</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>B1</td>
<td>177</td>
<td>ø</td>
<td>ALT 177</td>
<td>NOTE 6</td>
<td>CYAN BLUE</td>
<td>UNDERLINE</td>
<td></td>
</tr>
<tr>
<td>B2</td>
<td>178</td>
<td>ø</td>
<td>ALT 178</td>
<td>NOTE 6</td>
<td>CYAN GREEN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>B3</td>
<td>179</td>
<td>ø</td>
<td>ALT 179</td>
<td>NOTE 6</td>
<td>CYAN CYAN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>B4</td>
<td>180</td>
<td>ø</td>
<td>ALT 180</td>
<td>NOTE 6</td>
<td>CYAN RED</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>B5</td>
<td>181</td>
<td>ø</td>
<td>ALT 181</td>
<td>NOTE 6</td>
<td>CYAN MAGENTA</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>B6</td>
<td>182</td>
<td>ø</td>
<td>ALT 182</td>
<td>NOTE 6</td>
<td>CYAN BROWN</td>
<td>NORMAL</td>
<td></td>
</tr>
<tr>
<td>VALUE</td>
<td>AS CHARACTERS</td>
<td>AS TEXT ATTRIBUTES</td>
<td>COLOR/GRAPHICS MONITOR ADAPTER</td>
<td>IBM MONOCHROME DISPLAY ADAPTER</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>
</tr>
<tr>
<td>B7 183</td>
<td>ALT 183 NOTE 6</td>
<td>CYAN</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B8 184</td>
<td>ALT 184 NOTE 6</td>
<td>CYAN</td>
<td>DARK GREY</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B9 185</td>
<td>ALT 185 NOTE 6</td>
<td>CYAN</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BA 186</td>
<td>ALT 186 NOTE 6</td>
<td>CYAN</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BB 187</td>
<td>ALT 187 NOTE 6</td>
<td>CYAN</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BC 188</td>
<td>ALT 188 NOTE 6</td>
<td>CYAN</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BD 189</td>
<td>ALT 189 NOTE 6</td>
<td>CYAN</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BE 190</td>
<td>ALT 190 NOTE 6</td>
<td>CYAN</td>
<td>YELLOW</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BF 191</td>
<td>ALT 191 NOTE 6</td>
<td>CYAN</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO 192</td>
<td>ALT 192 NOTE 6</td>
<td>RED</td>
<td>BLACK</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C1 193</td>
<td>ALT 193 NOTE 6</td>
<td>RED</td>
<td>BLUE</td>
<td>UNDERLINE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2 194</td>
<td>ALT 194 NOTE 6</td>
<td>RED</td>
<td>GREEN</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C3 195</td>
<td>ALT 195 NOTE 6</td>
<td>RED</td>
<td>CYAN</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C4 196</td>
<td>ALT 196 NOTE 6</td>
<td>RED</td>
<td>RED</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C5 197</td>
<td>ALT 197 NOTE 6</td>
<td>RED</td>
<td>MAGENTA</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C6 198</td>
<td>ALT 198 NOTE 6</td>
<td>RED</td>
<td>BROWN</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C7 199</td>
<td>ALT 199 NOTE 6</td>
<td>RED</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C8 200</td>
<td>ALT 200 NOTE 6</td>
<td>RED</td>
<td>DARK GREY</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C9 201</td>
<td>ALT 201 NOTE 6</td>
<td>RED</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA 202</td>
<td>ALT 202 NOTE 6</td>
<td>RED</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CB 203</td>
<td>ALT 203 NOTE 6</td>
<td>RED</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CC 204</td>
<td>ALT 204 NOTE 6</td>
<td>RED</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD 205</td>
<td>ALT 205 NOTE 6</td>
<td>RED</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CE 206</td>
<td>ALT 206 NOTE 6</td>
<td>RED</td>
<td>YELLOW</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CF 207</td>
<td>ALT 207 NOTE 6</td>
<td>RED</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D0 208</td>
<td>ALT 208 NOTE 6</td>
<td>MAGENTA</td>
<td>BLACK</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALUE</td>
<td>AS CHARACTERS</td>
<td>AS TEXT ATTRIBUTES</td>
<td>COLOR/GRAPHICS MONITOR ADAPTER</td>
<td>IBM MONOCROME DISPLAY ADAPTER</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>
</tr>
<tr>
<td>D1</td>
<td>ALT 209</td>
<td>NOTE 6</td>
<td>MAGENTA BLUE</td>
<td>UNDERLINE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>ALT 210</td>
<td>NOTE 6</td>
<td>MAGENTA GREEN</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D3</td>
<td>ALT 211</td>
<td>NOTE 6</td>
<td>MAGENTA CYAN</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D4</td>
<td>ALT 212</td>
<td>NOTE 6</td>
<td>MAGENTA RED</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D5</td>
<td>ALT 213</td>
<td>NOTE 6</td>
<td>MAGENTA MAGENTA</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D6</td>
<td>ALT 214</td>
<td>NOTE 6</td>
<td>MAGENTA BROWN</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D7</td>
<td>ALT 215</td>
<td>NOTE 6</td>
<td>MAGENTA LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D8</td>
<td>ALT 216</td>
<td>NOTE 6</td>
<td>MAGENTA DARK GREY</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D9</td>
<td>ALT 217</td>
<td>NOTE 6</td>
<td>MAGENTA LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DA</td>
<td>ALT 218</td>
<td>NOTE 6</td>
<td>MAGENTA LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DB</td>
<td>ALT 219</td>
<td>NOTE 6</td>
<td>MAGENTA LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>ALT 220</td>
<td>NOTE 6</td>
<td>MAGENTA LIGHT RED</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DD</td>
<td>ALT 221</td>
<td>NOTE 6</td>
<td>MAGENTA LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DE</td>
<td>ALT 222</td>
<td>NOTE 6</td>
<td>MAGENTA YELLOW</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>ALT 223</td>
<td>NOTE 6</td>
<td>MAGENTA WHITE</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E0</td>
<td>ALT 224</td>
<td>NOTE 6</td>
<td>YELLOW BLACK</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E1</td>
<td>ALT 225</td>
<td>NOTE 6</td>
<td>YELLOW BLUE</td>
<td>UNDERLINE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E2</td>
<td>ALT 226</td>
<td>NOTE 6</td>
<td>YELLOW GREEN</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E3</td>
<td>ALT 227</td>
<td>NOTE 6</td>
<td>YELLOW CYAN</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E4</td>
<td>ALT 228</td>
<td>NOTE 6</td>
<td>YELLOW RED</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E5</td>
<td>ALT 229</td>
<td>NOTE 6</td>
<td>YELLOW MAGENTA</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E6</td>
<td>ALT 230</td>
<td>NOTE 6</td>
<td>YELLOW BROWN</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E7</td>
<td>ALT 231</td>
<td>NOTE 6</td>
<td>YELLOW LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E8</td>
<td>ALT 232</td>
<td>NOTE 6</td>
<td>YELLOW DARK GREY</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E9</td>
<td>ALT 233</td>
<td>NOTE 6</td>
<td>YELLOW LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EA</td>
<td>ALT 234</td>
<td>NOTE 6</td>
<td>YELLOW LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EB</td>
<td>ALT 235</td>
<td>NOTE 6</td>
<td>YELLOW LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALUE</td>
<td>AS CHARACTERS</td>
<td>AS TEXT ATTRIBUTES</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>
</tr>
<tr>
<td>COLOR/GRAPHICS</td>
<td>MONITOR ADAPTER</td>
<td>IBM MONOCHROME DISPLAY ADAPTER</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEX</td>
<td>DEC</td>
<td>SYMBOL</td>
<td>KEYSTROKES</td>
<td>MODES</td>
<td>BACKGROUND</td>
<td>FOREGROUND</td>
<td>ADAPTER</td>
</tr>
<tr>
<td>EC</td>
<td>236</td>
<td>∞</td>
<td>ALT 236</td>
<td>NOTE 6</td>
<td>YELLOW</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>ED</td>
<td>237</td>
<td>φ</td>
<td>ALT 237</td>
<td>NOTE 6</td>
<td>YELLOW</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>EE</td>
<td>238</td>
<td>ε</td>
<td>ALT 238</td>
<td>NOTE 6</td>
<td>YELLOW</td>
<td>YELLOW</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>EF</td>
<td>239</td>
<td>∩</td>
<td>ALT 239</td>
<td>NOTE 6</td>
<td>YELLOW</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>F0</td>
<td>240</td>
<td>≡</td>
<td>ALT 240</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>BLACK</td>
<td>REVERSE VIDEO</td>
</tr>
<tr>
<td>F1</td>
<td>241</td>
<td>±</td>
<td>ALT 241</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>BLUE</td>
<td>UNDERLINE</td>
</tr>
<tr>
<td>F2</td>
<td>242</td>
<td>∨</td>
<td>ALT 242</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>GREEN</td>
<td>NORMAL</td>
</tr>
<tr>
<td>F3</td>
<td>243</td>
<td>⇐</td>
<td>ALT 243</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>CYAN</td>
<td>NORMAL</td>
</tr>
<tr>
<td>F4</td>
<td>244</td>
<td>§</td>
<td>ALT 244</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>RED</td>
<td>NORMAL</td>
</tr>
<tr>
<td>F5</td>
<td>245</td>
<td>÷</td>
<td>ALT 245</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>MAGENTA</td>
<td>NORMAL</td>
</tr>
<tr>
<td>F6</td>
<td>246</td>
<td>×</td>
<td>ALT 246</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>BROWN</td>
<td>NORMAL</td>
</tr>
<tr>
<td>F7</td>
<td>247</td>
<td>≈</td>
<td>ALT 247</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
</tr>
<tr>
<td>F8</td>
<td>248</td>
<td>●</td>
<td>ALT 248</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>DARK GREY</td>
<td>REVERSE VIDEO</td>
</tr>
<tr>
<td>F9</td>
<td>249</td>
<td>•</td>
<td>ALT 249</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
</tr>
<tr>
<td>FA</td>
<td>250</td>
<td>.</td>
<td>ALT 250</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>FB</td>
<td>251</td>
<td>√</td>
<td>ALT 251</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>FC</td>
<td>252</td>
<td>n</td>
<td>ALT 252</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>FD</td>
<td>253</td>
<td>2</td>
<td>ALT 253</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>FE</td>
<td>254</td>
<td></td>
<td>ALT 254</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>YELLOW</td>
<td>HIGH INTENSITY</td>
</tr>
<tr>
<td>FF</td>
<td>255</td>
<td>BLANK</td>
<td>ALT 255</td>
<td>NOTE 6</td>
<td>WHITE</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
</tr>
</tbody>
</table>
NOTE 1 Asterisk (*) can easily be keyed using two methods: 1) hit the PRNTSC key or 2) in shift mode hit the 8 key.

NOTE 2 Period (.) can easily be keyed using two methods: 1) hit the 7 key or 2) in shift or NUM LOCK mode hit the del key.

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

NOTE 4 Upper case alphabetic characters (A–Z) can easily be keyed in two modes: 1) in shift mode hit 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 alphabetic 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.
## Character Set (00-7F) Quick Reference

<table>
<thead>
<tr>
<th>DECIMAL VALUE</th>
<th>HEXA-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>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>BLANK (NULL)</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>1</td>
<td>BLANK (SPACE)</td>
<td>0</td>
<td>@</td>
<td>P</td>
<td>'</td>
<td>p</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>A</td>
<td>Q</td>
<td>a</td>
<td>q</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>B</td>
<td>R</td>
<td>b</td>
<td>r</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>C</td>
<td>S</td>
<td>c</td>
<td>s</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>D</td>
<td>T</td>
<td>d</td>
<td>t</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>E</td>
<td>U</td>
<td>e</td>
<td>u</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>F</td>
<td>V</td>
<td>f</td>
<td>v</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>G</td>
<td>W</td>
<td>g</td>
<td>w</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>H</td>
<td>X</td>
<td>h</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>I</td>
<td>Y</td>
<td>i</td>
<td>y</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>A</td>
<td>J</td>
<td>Z</td>
<td>j</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>B</td>
<td>K</td>
<td>[</td>
<td>k</td>
<td>{</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>C</td>
<td>L</td>
<td>\</td>
<td>l</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>D</td>
<td>M</td>
<td>]</td>
<td>m</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>E</td>
<td>N</td>
<td>^</td>
<td>n</td>
<td>~</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>O</td>
<td>_</td>
<td>o</td>
<td>Δ</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALUE</td>
<td>AS CHARACTERS</td>
<td>COLOR/GRAPHICS</td>
<td>AS TEXT ATTRIBUTES</td>
<td>IBM MONOCHROME</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>MONITOR ADAPTER</td>
<td>BACKGROUND</td>
<td>FOREGROUND</td>
<td>DISPLAY ADAPTER</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEX</td>
<td>DEC</td>
<td>SYMBOL</td>
<td>KEYSTROKES</td>
<td>MODES</td>
<td>NOTE 5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>103</td>
<td>g</td>
<td>g</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>104</td>
<td>h</td>
<td>h</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>DARK GREY</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>105</td>
<td>i</td>
<td>i</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6A</td>
<td>106</td>
<td>j</td>
<td>j</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6B</td>
<td>107</td>
<td>k</td>
<td>k</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6C</td>
<td>108</td>
<td>l</td>
<td>l</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6D</td>
<td>109</td>
<td>m</td>
<td>m</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6E</td>
<td>110</td>
<td>n</td>
<td>n</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>YELLOW</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6F</td>
<td>111</td>
<td>o</td>
<td>o</td>
<td>NOTE 5</td>
<td>YELLOW</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>112</td>
<td>p</td>
<td>p</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>BLACK</td>
<td>REVERSE VIDEO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>113</td>
<td>q</td>
<td>q</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>BLUE</td>
<td>UNDERLINE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>114</td>
<td>r</td>
<td>r</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>GREEN</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>115</td>
<td>s</td>
<td>s</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>CYAN</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>116</td>
<td>t</td>
<td>t</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>RED</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>117</td>
<td>u</td>
<td>u</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>MAGENTA</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>118</td>
<td>v</td>
<td>v</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>BROWN</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>119</td>
<td>w</td>
<td>w</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>LIGHT GREY</td>
<td>NORMAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>120</td>
<td>x</td>
<td>x</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>DARK GREY</td>
<td>REVERSE VIDEO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>79</td>
<td>121</td>
<td>y</td>
<td>y</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>LIGHT BLUE</td>
<td>HIGH INTENSITY UNDERLINE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7A</td>
<td>122</td>
<td>z</td>
<td>z</td>
<td>NOTE 5</td>
<td>WHITE</td>
<td>LIGHT GREEN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7B</td>
<td>123</td>
<td>i</td>
<td>i</td>
<td>SHIFT</td>
<td>WHITE</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7C</td>
<td>124</td>
<td>i</td>
<td>i</td>
<td>SHIFT</td>
<td>WHITE</td>
<td>LIGHT CYAN</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7D</td>
<td>125</td>
<td>i</td>
<td>i</td>
<td>SHIFT</td>
<td>WHITE</td>
<td>LIGHT RED</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7E</td>
<td>126</td>
<td>i</td>
<td>i</td>
<td>SHIFT</td>
<td>WHITE</td>
<td>LIGHT MAGENTA</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7F</td>
<td>127</td>
<td>△</td>
<td>CTRL ←</td>
<td></td>
<td>WHITE</td>
<td>WHITE</td>
<td>HIGH INTENSITY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALUE</td>
<td>AS CHARACTERS</td>
<td>AS TEXT ATTRIBUTES</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>DEC</td>
<td>HEX SYMBOL</td>
<td>COLOR/GRAPHICS</td>
<td>IBM MONOCHROME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>KEYSTROKES</td>
<td>MODES BACKGROUND</td>
<td>DISPLAY ADAPTER</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FOREGROUND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>FF</td>
<td>80 - FF HEX ARE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FLASHING IN BOTH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>COLOR &amp; IBM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>128</td>
<td>80</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>129</td>
<td>81</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>129</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>130</td>
<td>82</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>130</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>131</td>
<td>83</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>131</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>84</td>
<td>132</td>
<td>84</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>132</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>133</td>
<td>85</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>133</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>86</td>
<td>134</td>
<td>86</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>134</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>87</td>
<td>135</td>
<td>87</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>135</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>88</td>
<td>136</td>
<td>88</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>136</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>89</td>
<td>137</td>
<td>89</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>137</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8A</td>
<td>138</td>
<td>8A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>138</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8B</td>
<td>139</td>
<td>8B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>139</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8C</td>
<td>140</td>
<td>8C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>140</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8D</td>
<td>141</td>
<td>8D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>141</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8E</td>
<td>142</td>
<td>8E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>142</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8F</td>
<td>143</td>
<td>8F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>143</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>144</td>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>144</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>91</td>
<td>145</td>
<td>91</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>145</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>92</td>
<td>146</td>
<td>92</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>146</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>93</td>
<td>147</td>
<td>93</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>147</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>148</td>
<td>94</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>148</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>95</td>
<td>149</td>
<td>95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>149</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>150</td>
<td>96</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>150</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>97</td>
<td>151</td>
<td>97</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>151</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>98</td>
<td>152</td>
<td>98</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>152</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>99</td>
<td>153</td>
<td>99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>153</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9A</td>
<td>154</td>
<td>9A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>154</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Character Set (80-FF) Quick Reference

<table>
<thead>
<tr>
<th>DECIMAL VALUE</th>
<th>128</th>
<th>144</th>
<th>160</th>
<th>176</th>
<th>192</th>
<th>208</th>
<th>224</th>
<th>240</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>10</td>
<td>A</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>12</td>
<td>C</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>13</td>
<td>D</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>14</td>
<td>E</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
<tr>
<td>15</td>
<td>F</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>D</td>
<td>C</td>
<td>E</td>
</tr>
</tbody>
</table>

*Note: The image contains various symbols and characters corresponding to the decimal and hexadecimal values listed.*
APPENDIX D  LOGIC DIAGRAMS

Contents:

System Board .......................................................... D-2
Keyboard ............................................................... D-12
IBM Monochrome Display And Parallel Printer Adapter ........................................ D-14
IBM Monochrome Display ............................................. D-24
Color/Graphics Monitor Adapter ..................................... D-25
IBM 80 CPS Matrix Printer .......................................... D-31
Parallel Printer Adapter ............................................. D-34
5 1/4" Diskette Drive Adapter ...................................... D-35
5 1/4" Diskette Drive .................................................. D-39
32 KB Memory Expansion ............................................. D-42
64 KB Memory Expansion ............................................. D-45
Asynchronous Communications Adapter .......................... D-48
Game Control Adapter ............................................... D-49
Note: Logics one and two of twelve are not applicable.
SYSTEM BOARD (DMA)
Keyboard Logic 1 of 2
Note: Logics one and two of twelve are not applicable.
IBM Monochrome Display And Parallel Printer Adapter Logic 4 of 12
IBM Monochrome Display And Parallel Printer Adapter Logic 5 of 12
IBM Monochrome Display and Parallel Printer Adapter Logic 9 of 12
IBM Monochrome Display And Parallel Printer Adapter Logic 10 of 12
IBM Monochrome Display And Parallel Printer Adapter Logic 11 of 12
IBM Monochrome Display And Parallel Printer Adapter Logic 12 of 12
DANGER
HAZARDOUS VOLTAGES UP TO 450 VOLTS EXIST ON THE PRINTED CIRCUIT BOARDS

IBM Monochrome Display
Color/Graphics Monitor Adapter Logic 1 of 6
Color/Graphics Monitor Adapter Logic 6 of 6
PARALLEL PRINTER ADAPTER
Note: Logics one and two of six are not applicable.

1. Signals on drive pins 10 thru 16 are swapped by the drive cable between drives 1 & 2 (and 3 & 4) as follows:
   10 to 16
   11
   12
   13
   14
   15
   16 to 10

2. All drives are jumpered for multiplex operation, head load with drive select and drive select via input pin 12. Terminating R-packs are left in drives 1 & 3 only.

3. 4.7nFD should be adjacent to modules, IN8BT, U4B, T4B, 74574, 160kHz.

4. All signal lines higher than or equal to 1MHz should be kept to the shortest possible length. This is a primary design goal.

5. Make no connection to unused pins on the VCO, charge pump & data separator modules

6. All voltage and ground connections to the VCO, charge pump and associated discrete components should be separate from the other circuits and then joined to the other circuits at one point.

5% Diskette Drive Adapter Logic 3 of 6

Appendix D
1. RESISTORS ARE IN OHMS, ± 5%, 1/4 W.
2. 1% RESISTORS ARE 1/8 W.
3. CAPACITORS ARE IN μF, ± 20%, 35 V.

5¼" Diskette Drive Logic 3 of 3
Appendix E
Unit Specifications

System Unit

Size:
   Length--19.6” (500 mm)
   Depth--16.1” (410 mm)
   Height--5.5” (142 mm)

Weight:
   Without Diskette Drive Unit-20.9 lbs (9.5 kg)
   With Diskette Drive Unit-25 lbs (11.4 kg)

Power Cable:
   Length--6’0” (1.83 mm)
   Size--18 AWG

Environment:
   Air Temperature
      System ON, 60° to 90° - F (15.6° to 32.2° C)
      System OFF, 50° to 110° - F (10° to 43° C)
   Humidity
      System ON, 8% to 80%
      System OFF, 20% to 80%
   Heat Output, 1083 BTU/HR (Maximum)

Noise Levels:
   Without Printer, 59 DBS
   With Printer, 66 DBS

Electrical:
   Nominal-120 VAC
   Minimum-104 VAC
   Maximum-127 VAC
   KVA-.3175 maximum

Keyboard

Size:
   Length--19.6” (500 mm)
   Depth--7.87” (200 mm)
   Height--2.2” (57 mm)

Weight:
   6.5 lbs (14.3 kg)
IBM Monochrome Display

Size:
- Length--14.9" (380 mm)
- Depth--13.7" (350 mm)
- Height--11" (280 mm)

Weight:
- 17.3 lbs (7.9 kg)

Heat Output:
- 325 BTU/HR

Power Cable:
- Length--3.0" (914 mm)
- Size--18 AWG

Signal Cable:
- Length--4'0" (1.22 mm)
- Size--22 AWG

IBM 80 CPS Matrix Printer

Size:
- Length--15.7" (400 mm)
- Depth--14.5" (370 mm)
- Height--4.3" (110 mm)

Weight:
- 12.9 lbs (5.9 kg)

Power Cable:
- Length--6.0" (1.83 mm)
- Size--18 AWG

Signal Cable:
- Length--6'0" (1.83 mm)
- Size--22 AWG

Heat Output:
- 341 BTU/HR (Max.)

Electrical:
- Nominal-120 VAC
- Minimum-104 VAC
- Maximum-127 VAC
1. **Address Buss**: A set of wires or signals carrying the binary-coded address from the Intel-8088 microprocessor throughout the rest of the IBM Personal Computer System Unit.

2. **AEN**: Address Enable. (Refer to System Board I/O Channel Descriptions).

3. **ALE**: Address Latch Enable. (Refer to System Board I/O Channel Descriptions).

4. **Analog**: (1) Pertaining to representation by means of continuously variable physical quantities. (2) Contrast with digital.

5. **A/N**: Alphanumeric: Pertaining to a character set that contains letters, digits, and usually other characters, such as punctuation marks. Synonymous with alphameric.

6. **A0-A19**: Address bits 0-19. (Refer to System Board I/O Channel Descriptions).

7. **APA**: All points addressable graphics.

8. **ASCII**: American Standard Code of Information Interchange. 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.

9. **Assembler**: A computer program used to assemble. Synonymous with assembly program.

10. **BASIC**: (Beginner’s all-purpose symbolic instruction code). A programming language with a small repertoire of commands and a simple syntax, primarily designed for numerical application.

11. **BAUD**: (1) A unit of signaling speed equal to the number of discrete conditions or signal events 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, i.e. if the duration of the unit interval is 20 milliseconds, the modulation rate is 50 baud.
12. 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.

13. BIOS: Basic Input/Output System.

14. Bootstrap: A technique or device designed to bring itself into a desired state by means of its own action, e.g. a machine routine whose first few instructions are sufficient to bring the rest of itself into the computer from an input device.

15. Buffer: 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. A portion of storage for temporarily holding input or output data.

16. Bus: One or more conductors used for transmitting signals or power.

17. Byte: (1) A binary character operated upon as a unit and usually shorter than a computer word. (2) The representation of a character.

18. CLK: Clock. (Refer to System Board I/O Channel Descriptions).

19. 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.

20. Computer: A data processor that can perform substantial computation, including numerous arithmetic operations, or logic operations, without intervention by a human operator during the run.

21. CPS: Characters per second.

22. CRC: The cyclic redundancy check character.

23. CRT: (1) A Cathode ray tube display. (2) A display device, such as the IBM Monochrome Display, that uses a cathode ray tube.

24. CTS: Conversational Terminal System. (2) Clear to Send. Associated with modem control.

25. DACK0-DACK3: DMA Acknowledge 0 to 3. (Refer to System Board I/O Channel Description).
26. 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.

27. Din Connectors: One of the connectors specified by the Din standardization committee.

28. DIP: "Dual In-Line Package." A widely used container for an integrated circuit. DIP's are pins usually in two parallel rows. These pins are spaced on 1/10" inters and come in different configurations ranging from a 14-pin assembly to a 40-pin configuration.

29. Display: A visual presentation of data.

30. DMA: Direct Memory Access.

31. DO-D7: Data Bits 0 to 7. (Refer to System Board I/O Channel Descriptions).

32. DRQ1-DRQ3: DMA Request 1 to 3. (Refer to System Board I/O Channel Descriptions).

33. DSR: Data Set Ready, associated with modem control.

34. DTR: Distribution Tape Reel.

35. Edge Connector: An opening which joins with the end of a circuit board. The purpose of this interface is to send electrical signals back and forth.

36. EIA/CCITT Drives: Electronic Industries Association/Consultative Committee on International Telegraphy and Telephony Drives.

EPROM or 'PROM': Term for "Programmable Read-Only Memory." An EPROM or 'PROM' is actually Read-Only Memory (ROM) but the contents may be changed by electrical means. EPROM or 'PROM' information is not destroyed when the power is cut off.

37. Firmware: Memory chips with the software programs already built in.

38. Graphics: Symbols Produced by a process such as handwriting, drawing or printing. Synonymous with graphic symbol.

39. Hexadecimal: Pertaining to a selection, choice, or condition that has sixteen possible values or states. These values or states usually contain 10 digits and six letters A through F. Hexadecimal digits are equivalent to a power of 16.
40. Hertz (Hz.): A unit of frequency equal to one cycle per second.

41. High order position: The leftmost position in a string of characters.

42. Input/Output (I/O): Pertaining to a device or to a channel that may be involved in an input process, and, at a different time, in an output process. (2) Pertaining to a device whose parts can be performing an input process and an output process at the same time.

43. Integrated Circuit: A combination of interconnected circuit elements inseparably associated on or within a continuous substrate.

44. Interpreter: A computer program used to interpret. Synonymous with interpretive program.

45. Interrupt: (1) A suspension of a process, such as the execution of a computer program, in such a way that the process can be resumed. (2) To stop a process in such a way that it can be resumed. (3) In data transmission, to take an action at a receiving station that causes the transmitting station to terminate a transmission.

46. I/O Channel: Input/Output Channel. In a data processing system, a functional unit, controlled by the processing unit, that handles the transfer of data between main storage and peripheral equipment.

47. I/O CH CK: I/O Channel Check. (Refer to System Board I/O Channel Descriptions).

48. I/O CH RDY: I/O Channel Ready. (Refer to System Board I/O Channel Descriptions).

49. IMR: Interruption Mask Register.

50. IOR: I/O Read Command. (Refer to System Board I/O Channel Descriptions).

51. IOW: I/O Write Command: (Refer to System Board I/O Channel Descriptions).

52. IRQ2-IRQ7: Interrupt Request 2 to 7. (Refer to System Board I/O Channel Descriptions).

53. K: An abbreviation for the prefix kilo, i.e. 1000 in decimal notation. To the tenth power, 1024 in decimal notation.

54. KB: Kilobyte.

55. Khz: Kilohertz. A unit of frequency equal to 1,000 hertz.
56. Low order position: The rightmost position in a string of characters.

57. Machine Language: (1) A language that is used directly by a machine. (2) Another term for computer instruction code.

58. Memory Address: A two-byte value selecting one specific memory location on a memory map.

59. Memory Location: The most specific part of a memory map that the computer can refer to.

60. Memory Map: The list of memory locations addressed directly by the microprocessor.

61. MEMR: Memory Read Command. (Refer to System Board I/O Channel Descriptions).

62. MEMW: Memory Write Command. (Refer to System Board I/O Channel Descriptions).

63. MFM Coded: Modified Frequency Modulation. It is double density encoding of information on a diskette.

64. Mhz: Megahertz. A unit of frequency equal to one million Hertz.

65. Microprocessor: A processing unit, or part of a processing unit, that consists of microcode. In the IBM Personal Computer, the microprocessor is the Intel-8088.

66. Mnemonic: Symbol or symbols used instead of terminology more difficult to remember. Usually a mnemonic has two or three letters.

67. 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.

68. 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.

69. Multiplexer: A device capable of interleaving the events of two or more activities or capable of distributing the events of an interleaved sequence to their respective activities.
70. 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. \( P \ OR \ Q \) is often represented by \( P+Q, \)
\( PVQ \). The term is synonymous with boolean add; logic add.

71. “ORed”: Past tense of OR.

72. OSC: Oscillator. (Refer to System Board I/O Channel
Descriptions).

73. Output: Pertaining to a device, process, or channel involved in
an output process, or to the data or states involved in an output
process.

74. Personal Computer: A small home or business computer
complete with a System Unit, keyboard, and available with a
variety of options such as monochrome display and a dot matrix
printer.

75. Pinout: A diagram of functioning pins on a pinboard.

76. Printed Circuit Board: A piece of material, usually fiberglass,
which contains a layer of conductive material, usually metal.
The metallic layer is then etched and electronic equipment is
then attached to the fiberglass. The electronic equipment then
has the capacity to transmit electronic signals through the board
by way of the etched metal tracks.

77. Program: (1) A series of actions designed to achieve a certain
result. (2) To design, write and test computer programs.

78. Read/Write Memory: Random access storage.

79. Reset Drv: Reset Driver. (Refer to System Board I/O Channel
Descriptions).

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

81. ROM: Read-only Memory.

82. ROM BIOS: Read-only Memory/Basic Input Output System.

83. RS 232 Port: Asynchronous Type Communications.

84. RTS: Ready to Send. Associated with modem control.
85. **Scan Line**: The use of a cathode beam to test the cathode ray tube of a display used with a personal computer.

86. **Schematic**: The description, usually in diagram form, of the logical structure and physical structure of an entire data base according to a conceptual model.

87. **Software**: (1) Computer programs, procedures, rules, and possibly associated documentation concerned with the operation of a data processing system. (2) Contrast with hardware.

88. **Strobe**: (1) An instrument used to determine the exact speed of circular or cyclic movement. (2) A flashing signal displaying an exact event.

89. **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.

90. **TX Data**: Transmit Data. External connections of the RS 232 Asynchronous Communications Adapter interface.

91. **Video**: Computer data shown or displayed on a cathode ray tube monitor or display.
BIBLIOGRAPHY

IBM Publications

1. IBM Personal Computer Guide to Operations-PN 6025000
   General information on using the IBM Personal Computer.

2. IBM Personal Computer Hardware and Service-PN 6025072
   Information on hardware and steps necessary when servicing this IBM Personal Computer.

3. IBM Personal Computer BASIC-PN 6025010
   Information for programmers who are using BASIC.

4. IBM Personal Computer Disk Operating System (DOS)-PN 6024001
   Information for programmers who are using DOS.

5. IBM Personal Computer MACRO Assembler-PN 6024002
   Information for experienced assembly language programmers using the Macro Assembler.

6. IBM Personal Computer Pascal Compiler-PN 6024010
   Information for programmers who are familiar with the Pascal language.
Other Related Publications

7. NATIONAL SEMICONDUCTOR
   INS 8250 Asynchronous Communications Element
   This book documents physical and operating characteristics
   of the INS 8250.

8. INTEL
   The 8086 Family Users Manual
   This manual introduces the 8086 family of microcomputing
   components and serves as a reference in system design
   and implementation.

9. INTEL
   8086/8087/8088 Macro
   ASSEMBLY Language
   Reference Manual for 8088/8085 Based Development
   System
   The manual describes the 8086/8087/8088 Macro Assembly
   Language, and is intended for use by persons who are familiar
   with assembly language.

10. MOTOROLA
    The complete Microcomputer Data Library
    This book can provide additional information on the Motorola
    6845 CRT Controller used in the IBM Monochrome Display
    and Parallel Printer Adapter, and the Color/Graphics
    Monitor Adapter.
INDEX

A

A0-A19 (Address Bits 0-19) 2-10
A.C. Output 2-34
Adapters
  Asynchronous Communications (RS232) 2-123
  Color/Graphics 2-45
  5 1/4” Diskette Drive 2-89
  Game Control 2-117
  IBM Monochrome Display and Parallel Printer 2-37
  Parallel Printer 2-65
Adapter Attribute Relationship 2-51
Adapter Inputs 2-107
Adapter Outputs 2-106
Address Bits (A0-A19) 2-10
Address Decode 2-118
Address Enable (AEN) 2-12
Address Latch Enable (ALE) 2-10
Address Strobe (ADS) 2-129
ADS (Address Strobe) 2-129
AEN (Address Enable) 2-12
ALE (Address Latch Enable) 2-10
Algorithms 3-8
All Points Addressable (APA) 2-45
Alphanumeric Mode 2-49
American Standard Code for Information Interchange (ASCII) 1-2
Analog Input 2-122
A/N (Alphanumeric) 2-49
Appendices A-0
  A - ROM BIOS Listing A-1
  B - Assembly Instruction Set Reference B-1
  C - Of Characters, Keystrokes and Color C-1
  D - Logic Diagrams D-1
  E - Unit Specifications E-1
APA (All Points Addressable) 2-45
ASCII - (See American Standard Code for Information Interchange) 1-2
ASCII Coding Table 2-78
ASCII Control Codes 2-79
ASCII (Extended) 3-11
Assembly Instruction Set Reference Appendix B
Assembly Language Reference Guide (See Bibliography)
Assessable Registers 2-154
Asynchronous Communications Adapter
   Block Diagram  2-124
   Current Loop Interface  2-127
   Input/Output Decode  2-125
   Input/Output Signals  2-133
   Input Signals  2-129
   Interface Description  2-126
   Interface Specifications  2-147
   Interrupt  2-126
   Interrupt Enable Register  2-141
   Interrupt Identification Register  2-139
   INS 8250 Accessible Registers  2-134
   INS 8250 Functional Pin Description  2-129
   INS 8250 Line Control Register  2-134
   INS 8250 Programmable Baud Rate Generator  2-135
   Line Status Register  2-137
   Logic Diagram  D-48
   Modem Control Register  2-142
   Modem Status Register  2-143
   Modes of Operation  2-125
   Output Signals  2-132
   Programming Considerations  2-133
   Receiver Buffer Register  2-144
   Reset Functions  2-133
   Selecting the Interface Format  2-146
   Transmitter Holding Register  2-145

B

BASIC (Beginner’s all-purpose symbolic instruction code)
   80 Interpreter  1-1
   Reserved Interrupts  3-23
   Screen Editor Special Functions  3-19
   Workspace Variables  3-23
Baud (See INS 8250 Programmable Baud Rate Generator)  2-132
Baud Out (BAUDOUT)  2-132
BEL (Bell)  2-82
Bell (BEL)  2-82
Berg Pin Connectors  2-63
BIOS
   (Basic Input Output System)  3-2
   Cassette Logic  3-8
   Memory Map  3-7
   Interrupt Vector Listing  3-3
   Parameter Passing  3-2
BIOS (continued)
  Programming Tip  3-2
  ROM (Read Only Memory)  3-2
  Use of  3-2
  Vectors With Special Meaning  3-5

BIT
  I/O Address  2-42
  I/O Bit Map  2-24
  Output Port Cassette  2-19
  Output Port Speaker  2-22
  Status Register  2-59

Block Diagrams
  Asynchronous Communications Adapter  2-124
  Cassette Motor Control  2-20
  Cassette Interface Read  2-19
  Cassette Interface Write  2-20
  Color/Graphics Monitor Adapter  2-47
  5 1/4” Diskette Drive Adapter  2-90
  Game Control Adapter  2-117
  Keyboard Interface  2-15
  IBM Monochrome Display Adapter  2-38
  Parallel Printer Adapter  2-66
  System  1-4
  Bootstrap  3-3

Byte:
  Attribute Definition  2-49
  Display Buffer  2-61

C

CAN (Cancel)  2-81
Cancel (CAN)  2-81
Carriage Return(CR)  2-79

Cassette
  Circuit Block Diagrams  2-19
  Data Record Architecture  3-10
  Error Recovery  3-10
  Interface Connector Specifications  2-21
  Interrupt 15  3-8
  Jumpers  2-19
  Logic (BIOS)  3-8
  Read  3-9
  Write  3-8
Character
  Codes 3-11
  Generator 2-48
  "Of Characters, Keystrokes and Color Appendix C
Set (00-7F) Quick Reference C-12
Set (80-FF) C-13
Chip Select Out (CSOUT) 2-132
Clear To Send (CTS) 2-130
Clock (4.77 mhz) 2-10
CLK (Clock) 2-10
CNTRL (Control) 3-14
Coding Table (ASCII) 2-78
Color/Graphics Monitor Adapter
  Block Diagram 2-47
  Character Generator 2-48
  Color/Graphics Mode 2-51
  Color Select Register 2-57
  Composite Color Generator 2-48
  Description of Basic Operations 2-54
  Display Buffer 2-48
  Graphics Storage Map 2-52
  I/O Address and Bit Map 2-61
  Interrupt Level 2-60
  Logic Diagrams D-25
  Major Component Definitions 2-48
  Memory Requirements 2-60
  Mode Register Summary 2-58
  Mode Select Register 2-58
  Mode Set and Status Registers 2-48
  Modes of Operation 2-49
  Monitor Adapter Auxiliary Connectors 2-63
  Monitor Adapter Direct Drive and Composite Interface
    Pin Assignment 2-62
  Monochrome Display Adapter Vs. Color/Graphics Adapter
    Attribute Relationship 2-51
  Motorola 6845 CRT Controller 2-48
  Programming the 6845 Controller 2-48
  Programming the Mode Control and Status Registers 2-57
  6845 Register Description 2-56
  Sequence of Events 2-59
  Timing Generator 2-48
  Composite Phone Jack 2-62
  Computer (IBM Personal) 1-1
  Command Phase 2-93
Command Status Registers
   Register 0 2-100
   Register 1 2-101
   Register 2 2-102
   Register 3 2-103

Control Codes
   CPS (Characters Per Second)  2-70
   CPU (Central Processing Unit) see System Board,
      Microprocessor  2-3
   CR (Carriage Return)  2-79
   CRT:
      (Cathode Ray Tube)  2-37
      Output Port 1 (I/O Address ‘3B8’)  2-42
      Status Port (I/O Address ‘3BA’)  2-43
   CTS (Clear To Send)  2-130
   CSOUT (Chip Select Out)  2-132
   Current  2-34
   Cursor  2-14

D

D0-D7 (Data Bits 0-7)  2-10
DACK0 - DACK3 (DMA Acknowledge 0 to 3)  2-12

DATA
   Bit (D0-D7)  2-10
   Bus Buffer/Driver  2-118
   Input Strobe (DISTR, DISTR)  2-129
   Output Strobe (DOSTR, DOSTR)  2-129
   Rates  2-39
   Record Architecture  3-10
   Set Ready (DSR)  2-131

Data Flow (System)  2-6
Data Flow (System)  2-6
DC 1 (Device Control 1)  2-81
DC 2 (Device Control 2)  2-81
DC 3 (Device Control 3)  2-81
DC 4 (Device Control 4)  2-80
DC Output  2-34
DEL (Delete)  2-81
Delete (DEL)  2-81
Device Control 1 (DC 1)  2-81
Device Control 2 (DC 2)  2-81
Device Control 3 (DC 3)  2-81
Device Control 4 (DC 4)  2-80
Digital Output Register (DOR)  2-91
DIN (Connectors) 2-5
DIP (Dual In-Line Package) 2-28
Diskettes 2-111
Diskette Drive (5 1/4") 2-110
Diskette Drive (5 1/4") Adapter
  Adapter Inputs 2-106
  Adapter Outputs 2-107
  Block Diagram 2-90
  Command Status Registers 2-100
  Command Summary 2-96
  Comments (Programming) 2-104
  Digital Output Register 2-91
  Drive A and B Interface 2-106
  Drive Constants 2-104
  DPC Registers 2-103
  External Interface Specifications 2-109
  Floppy Disk Controller 2-91
  Functional Description 2-91
  Internal Interface Specifications 2-108
  Logic Diagrams D-35
  Programming Considerations 2-94
  Programming Summary 2-103
  System I/O Channel Interface 2-104
Display (See IBM Monochrome Display) 2-43
Display Buffer 2-48
DISTR, DISTR (Data Input Strobe) 2-129
Divisor Latch 2-136
DMA (Direct Memory Access) 2-4
Dos Special Functions 3-19
DOSTR, DOSTR (Data Output Strobe) 2-129
Drive Constants 2-104
Drive Disable 2-132
DRQ1 - DRQ3 (DMA Request 1 to 3) 2-12
DSR (Data Set Ready) 2-131
DTR (Data Terminal Ready) 2-132
Dual In-Line Package Switches (DIP) 2-28

Edge Connector 2-108
Encoding 3-11
Error Recovery 3-10
ESC (Escape) 2-82
ESC A (Escape A) 2-83
ESC B (Escape B) 2-84
ESC C (Escape C)  2-85
ESC D (Escape D)  2-85
ESC E (Escape E)  2-86
ESC F (Escape F)  2-86
ESC G (Escape G)  2-86
ESC H (Escape H)  2-87
ESC 0 (Escape 0)  2-82
ESC 1 (Escape 1)  2-82
ESC 2 (Escape 2)  2-83
ESC 8 (Escape 8)  2-83
ESC 9 (Escape 9)  2-83
Escape (ESC)  2-82
Escape A (ESC A)  2-83
Escape B (ESC B)  2-84
Escape C (ESC C)  2-85
Escape D (ESC D)  2-85
Escape E (ESC E)  2-86
Escape F (ESC F)  2-86
Escape G (ESC G)  2-86
Escape H (ESC H)  2-87
Escape 0 (ESC 0)  2-82
Escape 1 (ESC 1)  2-82
Escape 2 (ESC 2)  2-83
Escape 8 (ESC 8)  2-83
Escape 9 (ESC 9)  2-83
Execution Phase  2-93
Extended Codes  3-13

F

FDC (Floppy Disk Controller)  2-91
Floppy Disk Controller (FDC)  2-91
Fonts  2-48
Functions and Conditions of DIP Switch 1  2-72
Functions and Conditions of DIP Switch 2  2-73
Functional Description
  5 1/4” Diskette Drive Adapter  2-91
  Game Control Adapter  2-118
  INS 8250  2-129

G

Game Control Adapter
  Address Decode  2-118
  Block Diagram  2-117
Game Control Adapter (continued)
  Connector Specifications 2-122
  Data Bus Buffer/Driver 2-118
  Functional Description 2-118
  Interface Description 2-119
  Joystick Positions 2-118
  Joystick Schematic 2-121
  Trigger Buttons 2-118

Glossary G-1
GND (Ground) 2-12
Graphics Character Extensions (Interrupt 1FH) 3-6
Graphics Mode (Color) 2-51
Graphics Storage Map 2-52

H

Hardware
  Asynchronous Communications Adapter 2-123
  Color/Graphics Monitor Adapter 2-45
  5 1/4” Diskette Drive 2-110
  5 1/4” Diskette Drive Adapter 2-89
  Game Control Adapter 2-117
  IBM Monochrome Display 2-43
  IBM Monochrome Display and Parallel Printer Adapter 2-37
  Memory Expansion Options 32KB and 64KB 2-113
  Printer 2-70
  Parallel Printer Adapter 2-65
  Power Supply 2-33
  System Board 2-3

Hardware
  Overview 1-1
  Data Flow 2-6

Hertz (Hz) 1-2
Horizontal Drive 2-43
Horizontal Tab (HT) 2-81
HT (Horizontal Tab) 2-81

I

IBM 80 CPS Matrix Printer 2-70
IBM Monochrome Display 2-43
IBM Monochrome Display and Parallel Printer Adapter
  Block Diagram 2-38
  Data Rates 2-39
  Direct Drive Interface and Pin Assignment 2-44
I-8
IBM Monochrome Display and Parallel Printer Adapter (continued)

DMA Channel 2-42
Interrupt and DMA Response Requirements 2-39
Interrupt Levels 2-42
I/O Address and Bit Map 2-42
Lines Used 2-39
Loads 2-39
Memory Requirements 2-41
Modes of Operation 2-40
Programming the 6845 CRT Controller 2-41
Sequence of Events 2-41

Important Operating Characteristics 2-36
IER (Interrupt Enable Register) 2-141
IIR (Interrupt Identification Register) 2-139

Input/Output Signals:
  Data (D0, D7) 2-133
  External Clock Input/Output (XTAL1, XTAL2) 2-133

Input Requirements (Power Supply) 2-34

Input Signals
  Address Strobe (ADS) 2-129
  Chip Select (SC0, CS1, CS2) 2-129
  Clear to Send (CTS) 2-130
  Data Input Strobe (DISTR, DISTR) 2-129
  Data Output Strobe (DOSTR, DOSTR) 2-129
  Data Set Ready (DSR) 2-131
  Master Reset (MR) 2-130
  Received Line Signal (RCLK) 2-130
  Receiver Clock (RCLK) 2-130
  Register Select (A0, A1, A2) 2-130
  Ring Indicator (RI) 2-131
  Serial Input (SIN) 2-130

Interface Diagram
  Asynchronous Communications Adapter 2-147
  Cassette Connector Specifications 2-21
  Color/Graphics Monitor Adapter 2-62
  5 1/4” Diskette Drive Adapter External 2-109
  5 1/4” Diskette Drive Adapter Internal 2-108
  Game Control 2-122
  IBM Monochrome Display Direct Drive 2-44
  Keyboard Connector Specifications 2-18
  Parallel Printer 2-69

INTRPT (Interrupt) 2-132
Interrupt Control Functions 2-140
Intel 8048 (Keyboard Microcomputer) 2-14
Intel 8088 (System Unit Microprocessor) 2-3
I/O (Input/Output)
  Address Map  2-33
  Channel  2-8
  Channel Description (System Board)  2-10
  Diagram  2-9
I/O CH CK (I/O Channel Check)  2-10
I/O CH RDY (I/O Channel Ready)  2-11
IOR (I/O Read Command)  2-11
IOW (I/O Write Command)  2-11
Interrupts
  And DMA Response Requirements  2-39
  Enable Register  2-141
  Identification Register  2-139
  Levels (0-XX)  2-42
  Vector Listing  3-3
  Vectors (0-7F)  3-21
  1 CH - Timer Tick  3-5
  1 DH - Video Parameters  3-5
  1 EH - Diskette Parameters  3-5
  1 FH - Graphics Character Extensions  3-6
15  3-8

K

KB - Kilobyte (See Memory Expansion Options)
Keyboard
  Break  3-16
  Character Codes  3-11
  Diagram  2-16
  Encoding  3-11
  Extended Codes  3-13
  Extended Functions  3-13
  Interface Block Diagram  2-15
  Interface Connector Specifications  2-18
  Pause  3-16
  Print Screen  3-16
  Scan Codes  2-17
  Shift States  3-14
  Shift Key Priorities  3-15
  Special Handling  3-15
  System Reset  3-15
  Usage  3-17
Kilobyte (KB) (See Memory Expansion Options)
L

LF (Line Feed) 2-79
Light Pen
  Interface 2-63
  Mode Control and Status Register 2-57
  Register Description 2-56
Line Control Register (LCR) 2-134
Line Feed 2-79
Line Status Register (LSR) 2-137
Lines Used 2-39
Loads 2-39
Logic Diagrams Appendix D
Low Memory Maps
  BASIC and DOS Reserved Interrupts (80-3FF) 3-22
  BASIC Workspace Variables 3-23
  Interrupt Vectors (0-7F) 3-21
  Reserved Memory Locations (400-5FF) 3-22

M

Major Component Definitions 2-48
Matrix Printer (IBM 80 CPS) 2-70
Megabyte 2-3
Memory
  BIOS Map 3-7
  Map (System) 2-25
  Module Description 2-114
  Module Pin Configuration 2-114
  Other Read/Write Usage 3-6
  Requirements (Color/Graphics) 2-60
  Requirements (IBM Monochrome) 2-41
  System Board Switch Settings 2-30
  32/64 KB Expansion Option Switch settings 2-31
Memory Expansion Options
  Memory Module Description 2-14
  Memory Module Pin Configuration 2-114
  Operating Characteristics 2-113
  Switch Configurable Starting Address 2-115
MEMR (Memory Read Command) 2-11
MEMW (Memory Write Command) 2-11
MFM (Modified Frequency Modulation) 2-110
Mhz (Megahertz) 2-43
Microprocessor 2-3
Microsecond 2-3
Mnemonic B-18
Mode Set and Status Register 2-48
Modem Control Register 2-142
Modem Status Register 2-143
Modes of Operation
  Asynchronous Communications Adapter 2-125
  Color/Graphics Monitor Adapter 2-49
  IBM Monochrome Display Adapter 2-40
Mode Register Summary 2-58
Mode Select Register 2-58
Monitor Type Switch Settings 2-29
Monochrome Display (IBM) 2-43
Monochrome Display and Parallel Printer Adapter (See IBM)
Motorola 6845 CRT Controller 2-48
MR (Master Reset) 2-130

N
NMI (Non-Maskable Interrupt)
  of the 8088 2-4
  to the 8088 2-8
NMI Mask. Reg. - (I/O Address Map) 2-23
Nominal Power Requirements 2-24
NUL (Null) 2-82
Null (NUL) 2-82
NUM LOCK 3-14

O
Of Characters, Keystrokes and Color C-1
OHM Resistors 2-67
Operating Characteristics
  Memory Expansion Options 2-113
  Monochrome Display 2-43
  Power Supply 2-36
Options
  Asynchronous Communications Adapter 2-12
  Color/Graphics Monitor Adapter 2-45
  5 1/4” Diskette Drive 2-110
  5 1/4” Diskette Drive Adapter 2-89
  Game Control Adapter 2-117
  IBM 80 CPS Matrix Printer 2-70
  IBM Monochrome Display 2-43
  32/64 KB Memory Expansion Options 2-113
Options (continued)
  Parallel Printer Adapter  2-65
“ORed”  2-65
OSC (Oscillator)  2-10
Other Read/Write Memory Usage  3-6
OUT PORT  2-39
Output
  AC  2-34
  Address  2-67
  DC  2-34
  Port  2-65
  OUT 1  2-132
  OUT 2  2-132
Output Signals
  Baud Out (BAUDOUT)  2-132
  Chip Select Out (CSOUT)  2-132
  Data Terminal Ready (DTR)  2-132
  Driver Disable (DDIS)  2-132
  Interrupt (INTRPT)  2-132
  Output 1 (OUT1)  2-132
  Output 2 (OUT2)  2-132
  Request to Send (RTS)  2-132
  Serial Output (SOUT)  2-132
Overview (Hardware)  1-1
Over Voltage/Current Protection  2-36

P

Parallel Printer Adapter
  ASCII Coding Table  2-78
  ASCII Control Codes  2-79
  Block Diagram  2-66
  Description  2-37
  DMA Channel  2-42
  IBM 80 CPS Matrix Printer  2-70
  I/O Address and Bit Map  2-42
  Interrupt Levels  2-42
  Logic Diagram  D-31
  Parallel Interface Description  2-73
  Printer Specifications  2-71
  Programming Considerations  2-67
  Programming the 6845 CRT Controller  2-41
  Sequence of Events  2-41
  Setting the DIP Switches  2-72
  Timing  2-77
Parameters, 6845 Initialization  2-41
Parameter Passing, ROM BIOS  3-2
Parity Flag, 8080 Flags  B-1
Pause, BIOS Cassette Logic Special Handling  3-16
Pin Connectors
  P2-6 Pin Berg Strip for Light Pen Connector  2-63
  P1-4 Pin Berg Strip for RF Modulator  2-63
  9 Pin Connector, Color Direct Drive  2-61
  9 Pin Connector, IBM Monochrome Display  2-44
  25 Pin Connector, Parallel Printer Adapter Block Diagram  2-66
  25 Pin ‘D’ Shell Connector, Asynchronous Adapter Block
  Diagram  2-124
  15 Pin ‘D’ Shell Connector, Game Controller Adapter (Analog
  Input Connector Specifications)  2-122
  25 Pin ‘D’ Shell Connector, Parallel Printer Adapter  2-69
  5 Pin Din Connector, Keyboard Interface Connector
  Specifications  2-18
  15 Pin Male ‘D’ Shell Connector, Joystick Schematic  2-121
Planar (See System Board  2-3; Intel 8088  2-3)
Power-on Self-Test
  System Board  2-4
  Keyboard  2-14
Power Supply  2-33
  AC Output  2-34
  DC Output  2-34
  Important Operating Characteristics  2-36
  Over Voltage/Current Protection  2-36
  Signal Requirements  2-36
  Input Requirements  2-34
  Power Supply Connectors and Pin Assignments  2-35
  Power Supply Location  2-34
Preface  i
Print Screen, Special Handling  3-16
Printer, IBM 80 CPS Matrix  2-70
Printer Specifications  2-71
Programmable Peripheral Interface (PPI) 8255A-5  2-19
Programming Considerations
  Asynchronous Communications Adapter
    Asynchronous Communications Reset Functions  2-133
    INS 8250 Accessable Registers  2-134
    INS 8250 Line Control Register  2-134
    INS 8250 Programmable Baud Rate Generator  2-135
    Interrupt Identification Register  2-139
    Interrupt Enable Register  2-141
    Line Status Register  2-137
    Modem Control Register  2-142
Programming Considerations
Asynchronous Communications Adapter (continued)
  Modem Status Register  2-143
  Receiver Buffer Register  2-144
  Transmitter Holding Register  2-145
Color/Graphics Monitor Adapter
  Color Select Register  2-57
  Control and Status Register  2-57
  I/O Address and Bit Map  2-61
  Interrupt Level  2-60
  Mode Register Summary  2-58
  Mode Select Register  2-58
  Programming the 6845 Controller  2-55
  Programming the Modem Control and Status Register  2-57
  6845 Register Descriptions  2-56
  Status Register  2-59
  Sequence of Events  2-59
5 1/4" Diskette Drive Adapter
  Command Status Registers  2-100
  Command Summary  2-96
  Symbol Descriptions  2-94
IBM Monochrome Display and Parallel Printer Adapter
  DMA Channel  2-42
  I/O Address and Bit Map  2-42
  Interrupt Levels  2-42
  Memory Requirements  2-41
  Sequence of Events  2-41
  Programming the CRT Controller  2-41
Programming the 6845 CRT Controller  2-41
  Comments  2-104
  DPC Registers  2-103
  Drive Constants  2-104

R

RAS  2-114
Rating Amps, Over Voltage/Current Protection  2-36
Ready Line  2-8
Read Block  3-9
Read Data  2-107, 2-108 (5 1/4" Diskette Drives, External)  2-109
Read Status (Parallel Printer Adapter Block Diagram)  2-66
Read/Write Memory
  Color Monitor  2-50
  Color TV  2-49
  Future Expansion in I/O Channel, 384 KB  2-26
Read/Write Memory (continued)

Graphics Storage Map 2-53
Hardware Overview 1-1
I/O Address Map 2-24
Memory Address Space 2-61
Memory Expansion Options 2-113
User 3-7
Receive Circuit 2-127
Receiver Clock, (RCLK) 2-128
Recording Medium 2-111
Refresh Cycles 2-8
Registers, Address
   Command Status 2-100
   Data 2-9
   Initialization
      Parameters 2-41
      INS 8250 Accessible 2-134
      INS 8250 Line Control 2-134
      Interrupt Enable 2-141
      Line Control 2-134
      Line Status 2-137
      Main Status 2-9
      Modem Control 2-142
      Modem Status 2-143
      Receiver Buffer 2-144
      Transmitter Holding 2-145
6845 2-55
Register File
   Color Select 2-57
6845 Data 2-41
   Description 2-56
6845 Index 2-42
6845 Initialization Parameters 2-41
   Mode Select 2-58
   Status 2-57, 2-59
Requirements
   Input (Power Supply) 2-34
   Memory 2-60
   Signal (Power Supply) 2-36
RESET DRV (Reset Drive) 2-10
   I/O Channel Description 2-119
   ROM Address Space, 256 KB 2-25
   =RESET 2-105
Reserved Memory Locations (400-5FF) 3-22
Result Phase 2-93
Response Requirements, Interrupt and DMA  2-39

Reverse Video
  Modes of Operation  2-40
  Color Graphics Monitor Adapter  2-45

RF Modulator
  Auxiliary Video Connector P1-4 Pin Berg Strip  2-63
  Color Graphics Monitor Adapter  2-45
  Interface  2-63

Ring Indicator  2-127

ROM
  Character Generator  2-48
  Color Graphics Mode  2-52
  Color Graphics Monitor Adapter  2-46
  Diagram  2-13
  Hardware Overview  1-1
  Keyboard  2-14
  Memory Expansion Options  2-113
  System Board  2-3, 2-4
  System Board Component Diagram  2-13
  System Memory Map  2-27

ROM and System Usage  3-1

ROM BIOS
  BIOS Cassette Logic  3-8
  BIOS Memory Map  3-8
  Cassette Read  3-9
  Cassette Write  3-9
  Data Record Architecture  3-10
  Description  3-2
  Error Recovery  3-10
  Interrupt 15  3-8
  Interrupt ICH Timer Tick  3-5
  Interrupt IDH Video Parameters  3-5
  Interrupt IEH Diskette Parameters  3-5
  Interrupt IFH Graphics Character Extensions  3-6
  Interrupt Vector Listing  3-3
  Other Read/Write Memory Usage  3-7

ROM BIOS Listing  Appendix A

ROS (Read Only Storage) (See ROM)

ROM, Request for Master  2-92

RS232-C (See Asynchronous Communications Adapter)  2-123

RTS (Ready to Send)  2-123

R/W (Read/Write)
  Symbol Description  2-95
Scan Codes  2-17
Screen  2-43
Schematic (See Logic Diagrams)
SCROLL LOCK  3-15
Selecting the Interface Format  2-146
Sequence of Events  2-59
Serial Input (SIN)  2-130
Serial Output (SOUT)  2-132
Setting the DIP Switches  2-72
Shift In  2-80
Shift Out  2-80
Shift States  3-14
SI (Shift In)  2-80
Signal Requirements  2-36
SIN (Serial Input)  2-130
SO  2-80
Software Algorithms  3-8
SOUT (Serial Output)  3-132
Speaker
   Drive System Block Diagram  2-22
   Interface  2-22
Special Handling  3-15
Special Timing  2-39
Specifications
  5 1/4" Diskette Drive  2-112
  Printer  2-71
  System Appendix E
Status Registers
  Color/Graphics  2-59
  0 (5 1/4" Diskette Drive Adapter)  2-100
  1 (5 1/4" Diskette Drive Adapter)  2-101
  2 (5 1/4" Diskette Drive Adapter)  2-102
  3 (5 1/4" Diskette Drive Adapter)  2-103
Storage (See Memory)
Strobe
   Address  2-129
   Data Output  2-129
Summary of Available Colors  2-55
Switch Settings
   Configurable Start Address  2-115
   5 1/4" Diskette Drives  2-29
   32/64 KB Memory Expansion Option  2-31
   Monitor Type  2-29
   System Board Memory  2-30
I-18
System Board
Cassette Interface Connector Specifications 2-21
Cassette Jumpers 2-19
Cassette User Interface 2-19
Circuit Block Diagrams (Cassette) 2-19
Component Diagram 2-13
Data Flow 2-6
5 1/4” Diskette Drive Switch Settings 2-29
I/O Address Map 2-23
I/O Channel 2-8
I/O Channel Description 2-10
I/O Channel Diagram 2-9
Keyboard 2-14
Keyboard Diagram 2-16
Keyboard Interface Block Diagram 2-15
Keyboard Interface Connector Specifications 2-18
Keyboard Scan Codes 2-17
32/64 KB Memory Expansion Option Switch Settings 2-31
Memory Switch Settings 2-30
Monitor Type Switch Settings 2-29
Speaker Drive System Block Diagram 2-22
Speaker Interface 2-22
System Memory Map 2-25
System Expansion Slots (See I/O Slots) 2-3
System Memory Map 2-25
System Memory Map (16 KB Increments) 2-26
System Unit 1-1
System Unit Power Connector 2-35
System Usage (ROM and) 3-1

T
Tab (Vertical) 2-79
T/C (Terminal Count) 2-12
Terminal Count (T/C) 2-12
THR (Transmitter Holding Register) 2-145
Timing Generator 2-48
Transmit Circuit 2-127
Transmit Data (TX) 2-127
Transmitter Holding Register (THR) 2-145
Transmitter Output and Receiver Input 2-126
Trigger Buttons 2-118
TX Data (Transmit Data) 2-127
Unit Specifications  Appendix E
Usage (Keyboard)  3-17
Use of BIOS  3-2
User Interface (Cassette)  2-19

Vectors
(0-7F Interrupt)  3-21
(Interrupt Listing)  3-3
Vectors With Special Meaning
  Interrupt 1CH Timer Tick  3-5
  Interrupt 1DH Video Parameters  3-5
  Interrupt 1EH Diskette Parameters  3-5
  Interrupt 1FH Graphics Character Extensions  3-6
  Other Read/Write Memory Usage  3-6
Vertical Drive  2-43
Video
  Monitor  2-62
  (Reverse)  2-45
  Signal  2-43
Voltage Interchange Information  2-128
Voltage
  Power Supply  2-33
  I/O Channel  2-8
  System Board I/O Channel Description  2-10
  Keyboard Interface Connector Specifications  2-18
  Cassette Interface Connector Specifications  2-21
  Speaker Interface  2-22
  Power Supply  2-33
    Power Supply Location  2-34
    Important Operating Characteristics  2-36
  Color Graphics Monitor Adapter Direct Drive and
    Composite Interface Pin Assignment  2-62
    Color/Graphics Monitor Adapter Auxiliary Video
      Connectors  2-62
  Printer Specifications  2-71
    Game Controller Adapter (Analog Input) Connector
      Specifications  2-122
  Voltage Interchange Information, Asynchronous Communications
    Adapter  2-128
    Selecting the Interface Format, Asynchronous Communications
      Adapter  2-146
I-20
W

Workspace (BASIC Variables) 3-23
Write (Cassette) 3-8
Write (Cassette Interface Hardware) 2-20
Write Data 2-106
Write Enable 2-106
Write Protect 2-107

Numerics

5 1/4” Diskette Drive 2-110
5 1/4” Diskette Drive Adapter 2-91
32/64 KB Memory Expansion Options 1-3
80 CPS Matrix Printer, IBM 2-70
80 Interpreter, BASIC 1-1
6845 CRT Controller 2-48
8080 Parity Flags B-1
8088, Intel 2-3
8250 INS Accessible Registers 2-134
RS232C-A Asynchronous Communications Adapter 2-123
Product Comment Form

TECHNICAL REFERENCE 6025008

Your comments assist us in improving our products. 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.

Comments:

If you wish a reply, provide your name and address in this space.

Name ________________________________________
Address ________________________________________
City ________________ State ________________
Zip Code __________

Product Comment Form

TECHNICAL REFERENCE 6025008

Your comments assist us in improving our products. 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.

Comments:

If you wish a reply, provide your name and address in this space.

Name ____________________________________________
Address __________________________________________
City ____________ State ________________
Zip Code __________
POSTAGE WILL BE PAID BY ADDRESSEE

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

TECHNICAL REFERENCE 6025008

Your comments assist us in improving our products. IBM may use and distribute any of the information you supply in anyway it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Comments:

If you wish a reply, provide your name and address in this space.

Name ________________________________
Address ______________________________
City ___________ State ________________
Zip Code __________
IBM PERSONAL COMPUTER
SALES & SERVICE
P.O. BOX 1328-C
BOCA RATON, FLORIDA 33432