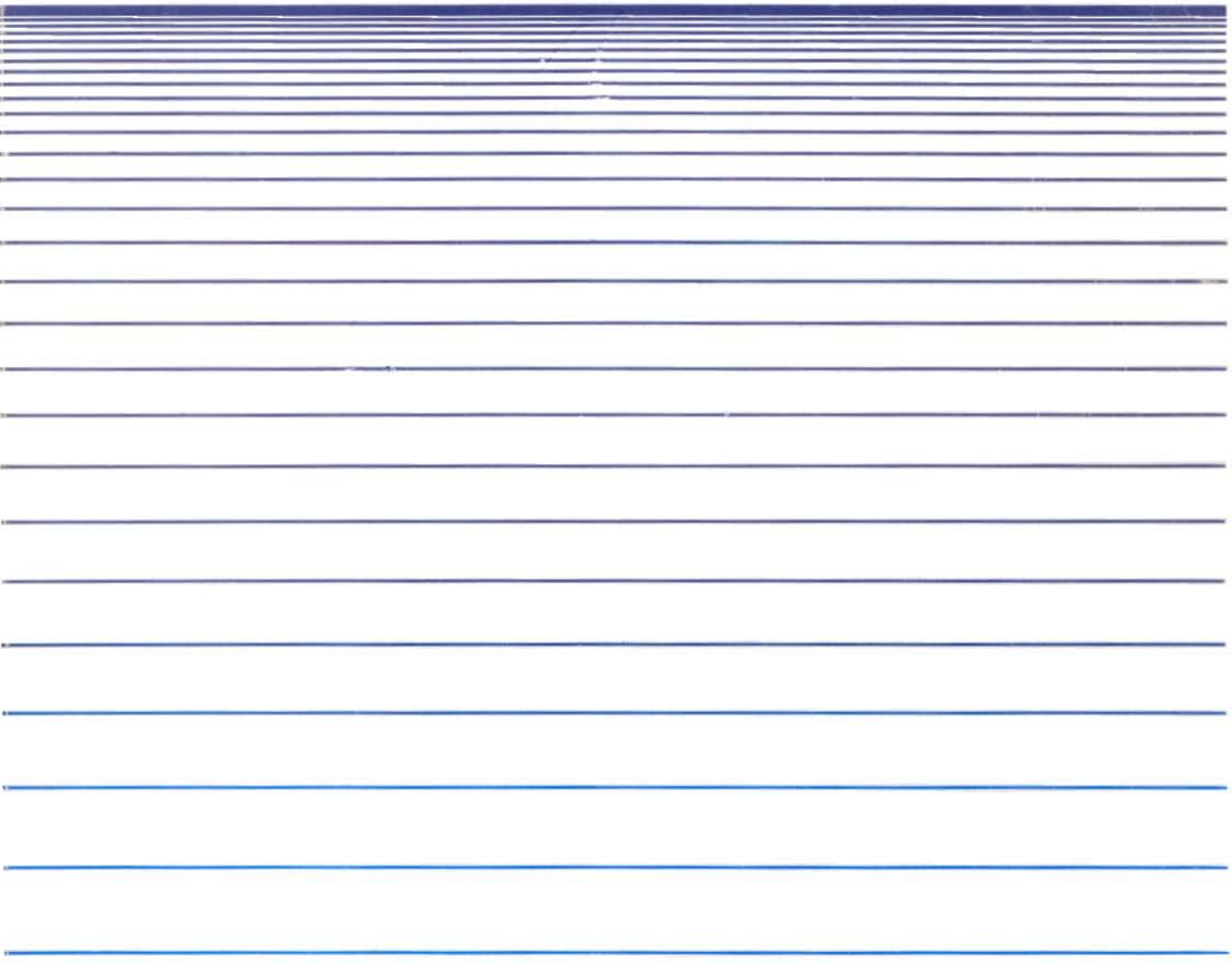
SupersPort Portable Computer

User/Technical Manual



ZENITH data systems

THE QUALITY GOES IN BEFORE THE NAME GOES ON

SupersPort Portable Computer

User/Technical Manual

595-4078
ZFL-184-2/4

ZENITH | data systems

THE QUALITY GOES IN BEFORE THE NAME GOES ON

FEDERAL COMMUNICATIONS COMMISSION RADIO FREQUENCY INTERFERENCE STATEMENT

WARNING: As sold by the manufacturer, the equipment described in this manual 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 with radio and TV reception.

NOTE: In order to meet Class B emission limits, the I/O cables that interconnect between the computer and any peripheral (such as a printer, external modem, etc.) must be shielded.

This equipment generates and uses radio frequency energy for its operation and if not installed and used properly, that is, in strict accordance with the instruction manual, may cause interference with radio and television reception. It has been tested and found to comply with the RF emission limits for a Class B computing device which are intended to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause interference with radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Move the computing device away from the receiver being interfered with.
- Relocate the computing device with respect to the receiver.
- Reorient the receiving antenna.
- Plug the computing device into a different AC outlet so that the computing device and receiver are on different branch circuits.
- Be certain that the computing device is plugged into grounded outlet receptacles. Avoid using AC cheater plugs. Lifting of the power cord ground may increase RF emission levels and may present a lethal shock hazard to the user.

If you need additional help, consult your dealer or ask for assistance from the manufacturer. You may also find the following booklet helpful: How to Identify and Resolve Radio-TV Interference Problems. This booklet is available from the U.S. Government Printing Office, Washington D.C., 20402, Stock No. 004-000-00345-4.

FCC regulations Part 68 places three restrictions on using the modem:

1. The modem cannot be connected to a party line or coin-operated telephone line.
2. You must notify the telephone company that the modem is being installed and provide the following information: modem registration number, ringer equivalence number, and telephone number to which the modem is connected.
3. Notify the telephone company if you have questions about the operation of your telephone line when connected to the modem.

Limited Rights Legend

Contractor is Zenith Data Systems Corporation of St. Joseph, Michigan 49085. The entire document is subject to Limited Rights data provisions.

Trademarks and Copyrights

Microsoft® MS-DOS® is a registered trademark of Microsoft Corporation.

Zenith Data Systems Corporation
St. Joseph, Michigan 49085

Contents

Contents

Contents

Contents

Tables

Contents

Listings

Welcome

Congratulations on your purchase of a new SupersPort Portable computer. This computer, shown in Figure 1, is for people who travel or spend time away from their desk. For these people — including executives, managers, sales representatives, and writers — the portable computer is capable of satisfying both business and personal needs.

With Zenith Data Systems, the future is here today. Your new computer can act as a portable, stand-alone tool or function as the heart of a powerful business automation system. Combined with software supplied by Zenith Data Systems, it can produce practical and affordable solutions to your business needs. It can easily satisfy such typical applications as data processing, telecommunications, financial analysis, and word processing, and aid you in making decisions that affect your business.

This Owner's Manual is for you, the new computer user. In the first part of this manual, you will learn how to set up and operate your new computer for the first time. In the second part, you will learn about the firmware of the computer and how to program it. In the third part, you will learn about the hardware and how to use the programmable registers in the computer. In the last part, you will find the specifications, along with a chapter on what to do when things don't work the way you think they should.

Welcome to computing at its best and thank you for selecting Zenith Data Systems, where the quality goes in before the name goes on.

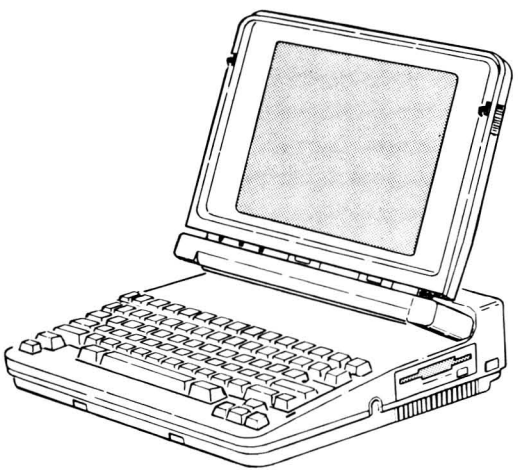


Figure 1. The SupersPort Portable Computer

Part I
Operation

Chapter 1

Up and Running

In this chapter you will learn how to use your computer for the first time. If you are new to computing or have not used this computer before, follow the procedures in this chapter.

Getting Started

Before you are ready to operate this computer, you need to have some basic information about this powerful portable computer.

Refer to Figure 1-1 for the following steps.

1. To open the computer, slide the latches on each side of the top of the computer toward the front as illustrated.

2. Lift the top from the front as illustrated.

3. The top contains the video display. The screen can be adjusted through an arc of 180°. Adjust the screen for the most comfortable viewing angle.

NOTE: The latches used on the Portable computer will only open with the computer in a horizontal position. This prevents the top from opening during transportation and possibly damaging the computer.

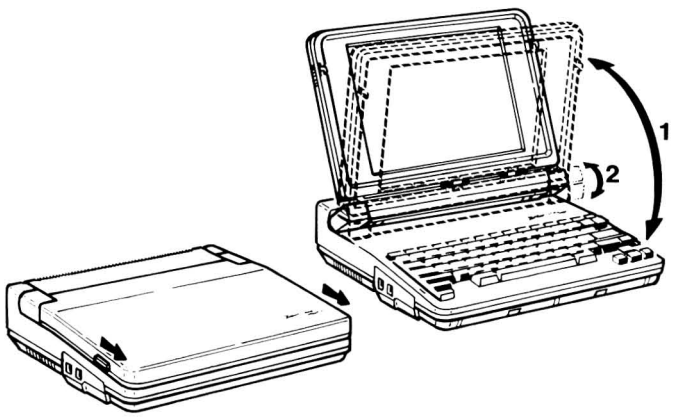


Figure 1-1. Opening the Computer

Keyboard

The keyboard, illustrated in Figure 1-2, contains 78 keys. It can emulate the full keyboard of a standard PC-compatible computer. For more information about the keys and the modes of operation, refer to Chapter 3.

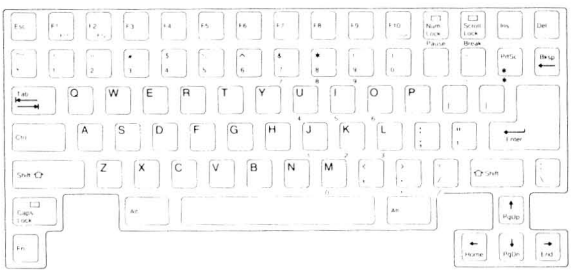


Figure 1-2. Keyboard

Disk Drives

The disk drives are on the sides of the computer. Some models contain one or two 3.5-inch floppy disk drives. Other models contain one 3.5-inch floppy disk drive and an internal hard disk drive. Disk drives are storage devices that you use to transfer information into and out of the computer's memory.

Figure 1-3 shows the position of the disk drives. Your Portable computer contains a floppy disk drive in location A. If the computer contains a second floppy disk drive, it is in location B. If the computer contains a hard disk drive, it is in location B.

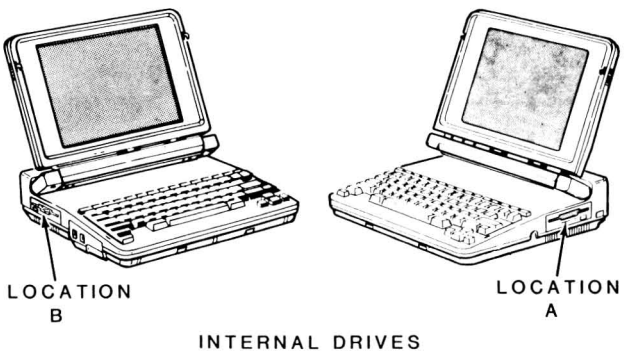


Figure 1-3. Disk Drive Locations

Disk drive systems have air holes that allow air movement through the disk drive. Do not block the air holes of the computer.

Indicators and Controls

Refer to Figure 1-4. Below the LCD are the two green DRIVE ACTIVE indicators and the red LOW POWER indicator. Beside these are the CONTRAST and BRIGHTNESS controls.

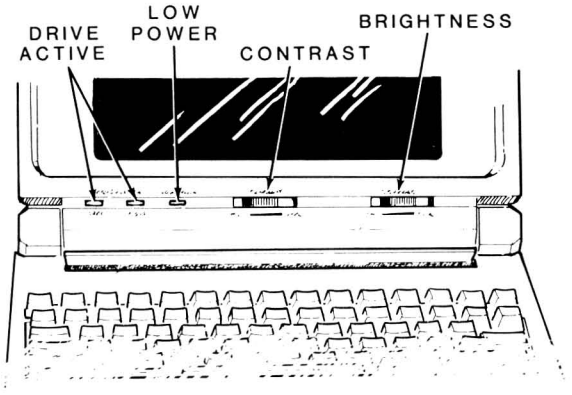


Figure 1-4. Indicators and Controls

Slide the CONTRAST control in the direction labeled MIN to decrease the contrast. Slide it toward MAX to increase the contrast. Slide the BRIGHTNESS control in the direction labeled MIN to decrease the brightness of the backlight in the display. Slide the control toward MAX to increase the backlight's output.

For now, set the CONTRAST and BRIGHTNESS controls to the middle of their ranges.

Disks and Disk Loading

The following discussion introduces you to the 3.5-inch disk. For more information on all types of disks, refer to Chapter 3.

Refer to Figure 1-5. The 3.5-inch disk consists of several parts. The top of the disk is solid, except for a metal shield and a write-protect hole. An arrow points toward the front of the disk.

The bottom of the disk has an opening through which the hub of the disk itself protrudes. Also, the write-protect slide is visible from the bottom.

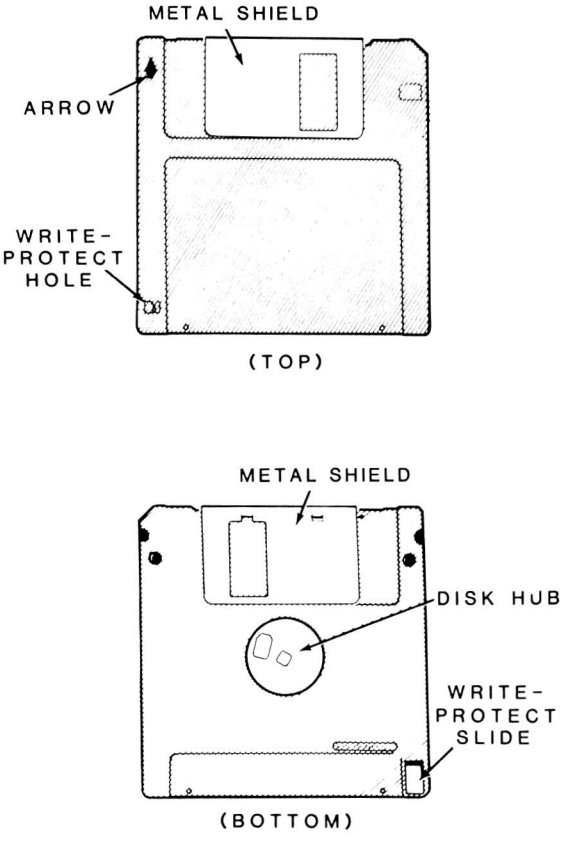


Figure 1-5. 3.5-Inch Disk

Refer to Figure 1-6 for the following discussion on loading disks.

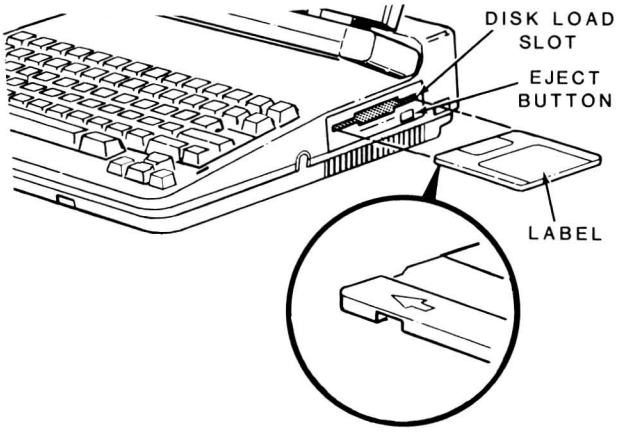


Figure 1-6. Loading a Disk

Insert a disk in the disk load slot with the arrow up and pointing into the drive. Slide the disk all the way into the drive until it latches into place.

To release the disk from the drive, press the eject button.

Power Up

Power Connections

You can operate this computer from the detachable battery pack or an external power adapter. Also, you use the external adapter to recharge the battery pack. The battery pack can be attached or separate from the computer while being charged.

There are two different external power adapters: 120/240-volt AC, or 12-volt DC. The AC adapter has a transformer and AC-to-DC conversion circuits. The DC adapter draws current from a cigarette lighter socket.

CAUTION: Do not attempt to start a motor vehicle while you have the computer connected to the vehicle's cigarette lighter. The starting circuits of most motors generate transient signals that can damage the computer through the power supply.

Refer to Figure 1-7 and the following steps to connect an external adapter to the computer:

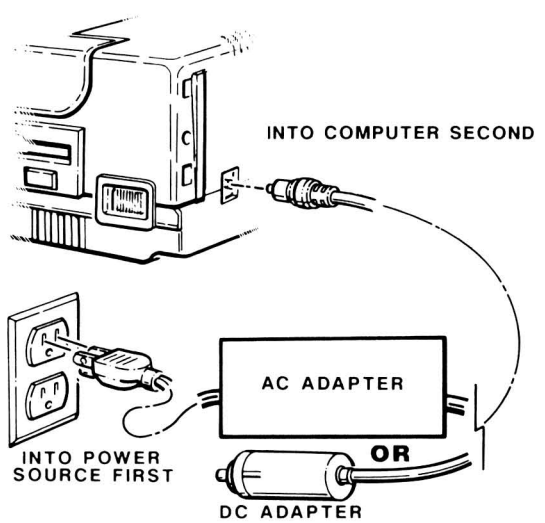


Figure 1-7. External Power Adapters

1. Plug the adapter into its AC or DC power source.
2. Plug the adapter into the computer.

NOTE: Always plug the AC adapter into its AC or DC power source first to prevent power surges from damaging the computer.

You are now ready to operate your new computer. The directions in this section will help you turn on the computer and load the operating system from the disk.

When you turn on the power, the computer performs a sequence of operations that makes sure everything is ready to go. These power-up self-tests check the CPU and other parts of the computer. The hard disk drive, if present, starts up and synchronizes the heads. The floppy disk drives also make some noise as they synchronize their heads with the computer.

Autoboot

Portable computers autoboot (they automatically load a bootable disk) when you turn on the computer. The computer attempts to load the disk from drive A first. If the computer cannot find a floppy disk in drive A, it will check the hard disk system, if present.

To defeat this process, press the ESC key. This will enter the Monitor program. Refer to Chapter 4 for additional information. You cannot permanently disable the autoboot feature.

If it does not find a bootable disk or partition, the computer will display an error message. In floppy disk systems, the following message appears about 10 seconds after you turn on the system if there is no disk in drive A:

```
+++ DISK ERROR: DRIVE NOT READY! +++
```

If there is a disk in drive A, but it does not have an operating system on it, then the computer will display:

```
Non-system disk.
Correct and press any key to reboot.
```

In computers with hard disk systems, the following message appears about 30 seconds after you turn on the computer:

```
Not a bootable partition
```

These error messages are normal. They tell you that you did not place a bootable disk in the drive or you have not transferred DOS to the hard disk.

Power-Up Sequence

The following is a typical power-up sequence for this computer. The procedure uses a common operating system, MD-DOS®. Other operating systems can produce different results.

1. Locate the operating system disk. Refer to Figure 1-8 and make sure the write-protect slide is in the write-protect position. When you write-protect a disk, you cannot change the information on the disk accidentally.

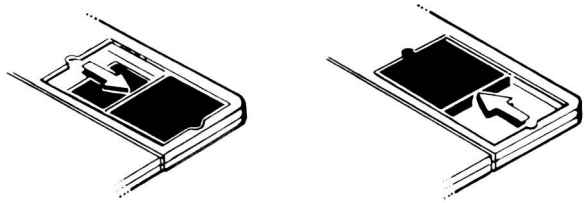


Figure 1-8. Write-Protect Slide

2. Insert the operating system disk into drive A with the arrow up and pointing into the drive.

3. Refer to Figure 1-9 and position the switch to the 1 setting to turn on the computer. The screen will light up. After a few seconds, the hard disk drive, if present, will start up. On computers that do not have a hard disk, the operating system will now load from the disk.

   After the hard disk drive starts up, you will hear a clicking sound. This is the normal sound the hard disk drive makes when it moves its heads. For more information on the operation of the hard disk drive, refer to Chapter 3.

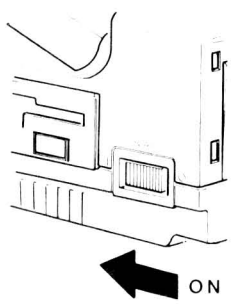


Figure 1-9. Power Switch

4. During the booting process, the disk drive heads move back and forth over the surface of the disk. With most drives, you will hear some noise.

NOTE: The following material uses MS-DOS version 3.2 as an example. Your operating system can produce different results. Some systems can require that you enter special information. For more information refer to your operating system documentation.

   After another 13 or 14 seconds have passed, your display will show a message similar to the following:

```
A>date
Current date is Fri 9-19-1986
Enter new date (mm-dd-yy):
```

5. With a display on the screen, adjust the CONTRAST and BRIGHTNESS controls for the best display. Keep the BRIGHTNESS control as low as possible to extend battery pack operation.

6. The Portable computer contains a real-time clock. The date actually shown by the computer will differ from the sample in step 4. MS-DOS contains a command (RTCLOCK) to let you set the real-time clock. For now, press the ENTER key. The computer will accept the date as it is currently displayed and MS-DOS will display a message similar to the following:

```
A>time
Current time is 8:54:52.63
Enter new time:
```

7. Press the ENTER key to accept the time as it is displayed. The computer will now load a program called SETUP. SETUP allows you to backup your distribution disk and create working MS-DOS floppy disks. If a hard disk drive is present, SETUP will guide you through preparing it for use. Refer to your operating system instructions for further information.

For information on how to set the real-time clock time and date, refer to RTCLOCK in your MS-DOS documentation.

Power Down

To turn off the computer:

1. Save any work that you are doing on the computer.
2. Remove any floppy disks from the drives.
3. Turn the power off.

If you plan to move the computer, close the cover and make sure it latches.

Chapter 2

Installation and Configuration

The heart of a computer system is the computer with its built-in keyboard, video display, and disk drives. Refer to Figure 2-1.

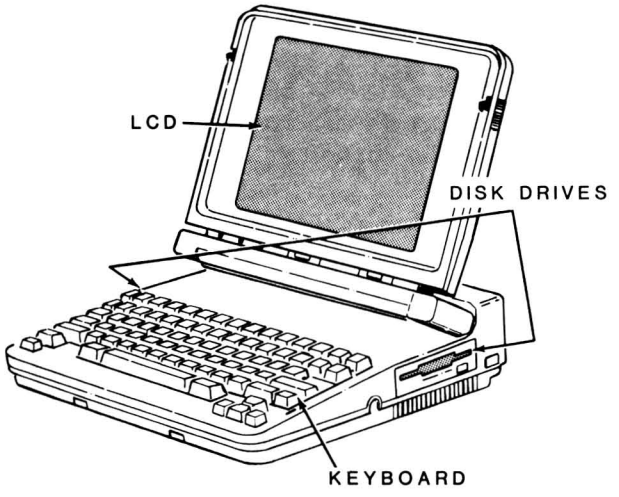


Figure 2-1. The Portable Computer

You can compare the video display (often called the LCD, or Liquid Crystal Display) to a television screen. This is where you see and work with the programs you use in the computer. Or, if you prefer, you can connect a separate video monitor to the computer for use in a desktop system.

You use the keyboard to enter information into the computer. It has all the keys contained on a standard typewriter keyboard plus additional special function keys. (For more information on the keyboard, refer to Chapter 3.)

You use the disk drives to store large amounts of information on 3.5-inch floppy disks or the hard disk system. Each 3.5-inch disk holds about 350 pages of typewritten text. In comparison, the built-in hard disk drive can hold about 10,000 pages of typewritten text. (For more information on disks, refer to Chapter 3.)

A peripheral is any piece of equipment you attach to and control with the computer. It may be an external video display, a printer, or an external floppy disk drive. Figure 2-2 illustrates typical system components.

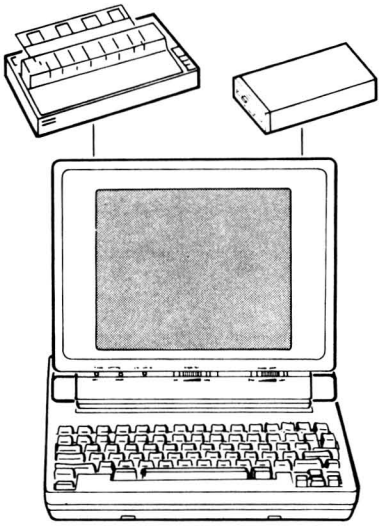


Figure 2-2. Typical System Components

External peripherals can enhance the operation of your computer. An external 5.25-inch disk drive makes it possible for you to transfer programs to or from 5.25-inch floppy disks. You can produce copies of letters, reports, accounting records, and other material with a printer. The modem adds the ability to exchange information with other computers over the telephone.

Installation

The following paragraphs describe how you can use the connectors on the computer to attach peripherals. When you use your computer with peripherals, select an area that:

- Can accommodate the computer and all of its peripherals.
- Has a level work surface that is near a power source and telephone.
- Has an environment range of 40°-104°F (10°-40°C) and 20% - 80% relative humidity.

Turn the computer off and unplug any external power source. Make sure you can reach the back panel to make the following connections.

Back Panel of the Computer

Refer to Figure 2-3 to locate the back panel connectors.

The following information briefly describes the connectors located on the back panel. For more information about these connectors, refer to Chapter 12.

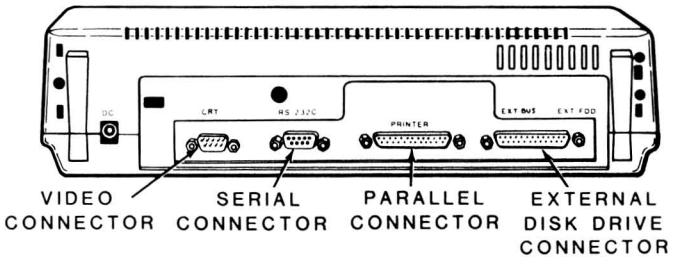


Figure 2-3. Back Panel Connections

Video Connector — The CRT connector supplies RGB video output for an external RGB video monitor or a composite monochrome signal for a composite monochrome video monitor.

- **RGB external monitor** — Connect the RGB monitor cable to the connector labeled CRT. Use an RGB monitor that has a resolution of 640 × 200 pixels or better. You can use either color or monochrome CGA monitors with the computer.

- **Composite monochrome external monitor** — Connect a special composite monochrome monitor cable to the connector labeled CRT. Connect the other end to the composite jack on the monitor. This cable is available from your Zenith Data Systems dealer.

NOTE: Do not use a TTL-compatible Monochrome Display Adapter (MDA) monitor or a TTL-compatible Hercules monitor with this computer.

Refer to Figure 2-4 for the video connector.

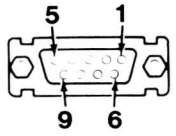


Figure 2-4. Video Connector

Serial Connector — The RS-232C connector provides the signals necessary for serial RS-232C communication to a printer or other serial device. Operating systems usually contain the necessary software to configure the connector for use with most serial devices. Connect a 9-pin serial communications cable to the connector labeled RS-232C. You may need an additional cable to route the signals properly for some serial peripherals.

Refer to Figure 2-5 for the serial connector.

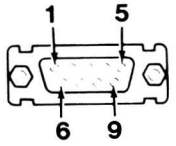


Figure 2-5. Serial Connector

Parallel Connector — The PRINTER connector provides Centronics-type output signals for a parallel printer or peripheral. Operating systems usually contain the necessary software to configure this connector for use with most parallel devices. Connect the parallel cable to the connector marked PRINTER. Connect the other end to the peripheral.

Refer to Figure 2-6 for the parallel connector.

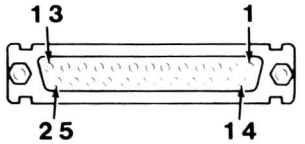


Figure 2-6. Parallel Connector

External Disk Drive Connector — The EXT FDD connector provides the signals necessary to connect an external floppy disk drive (FDD). Connect the external floppy disk drive cable to the connector labeled EXT FDD.

Refer to Figure 2-7 for the external disk drive connector.

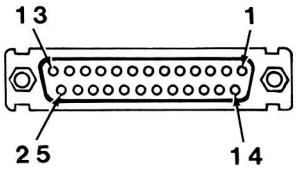


Figure 2-7. External Disk Drive Connector

Internal Modem

With the installed modem, you can connect your computer to the telephone lines. Refer to Figure 2-8 for the following connections.

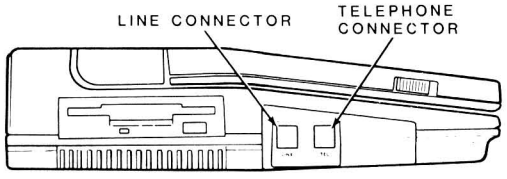


Figure 2-8. Left Side View

The modem board has two 6-pin, RJ-11 telephone connectors mounted on it for connection to the telephone line and a telephone.

Line Connector — This connector supplies the necessary signals for telephone computer-to-computer communication. Use it to connect the internal modem to the telephone company line. Connect one end of a standard, single-line telephone line (not handset) cord to the connector labeled LINE. Connect the other end to a telephone line jack. You can connect either end first. Refer to Figure 2-9 for the line connector.

Telephone Connector — The TEL connector supplies the necessary signals for a standard, single-line telephone through the computer from the telephone company line. Connect the telephone to the connector labeled TEL. Refer to Figure 2-9 for the telephone connector.

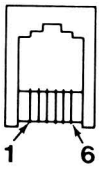


Figure 2-9. Line and Telephone Connectors

Keypad Connector — This connector is located on the right side of the computer as shown in Figure 2-10.

The keypad connector supplies the necessary power and signal lines for an external numeric keypad.

Refer to Figure 2-11 for the external keypad connector.

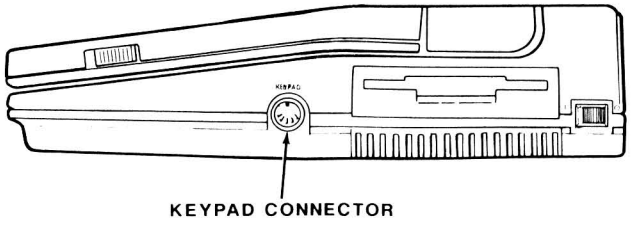


Figure 2-10. Right Side View

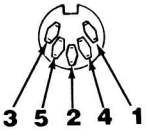


Figure 2-11. External Keypad Connector

Battery Pack Installation

Use the following procedure for installing the battery pack. You do not need any special tools. Refer to Figure 2-12 for the following steps.

1. Position the battery pack as shown in the illustration.
2. Insert the battery pack hooks labeled A into the holes in the back of the computer.
3. Rotate the battery pack toward the computer until the guide pin enters its receptacle in the computer.
4. Press the battery pack button and insert the hooks labeled B into the holes in the computer.
5. Release the button and press in on the battery pack until it locks in place.

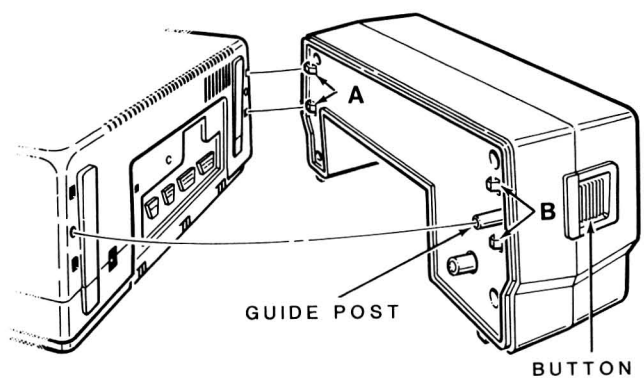


Figure 2-12. Battery Pack Installation and Removal

Battery Pack Removal

Refer to Figure 2-12 for the following steps.

1. Press in on the battery pack button and rotate the button end of the battery pack away from the computer.
2. Remove the hooks labeled A from the holes in the back of the computer.

Configuration

Hardware jumpers and switches control the configuration of some computers. Other computers use a software or firmware setup procedure. This computer uses a combination of switch settings and firmware to configure it for proper operation.

You use a single, 8-section switch to configure the computer. You can change the settings on the switch without disassembling the computer. A removable access cover, located on the bottom of the computer, protects the switch. Refer to Figure 2-13 for the switch's location and Table 2-1 for a description of each switch section. The text that follows the table explains each switch option.

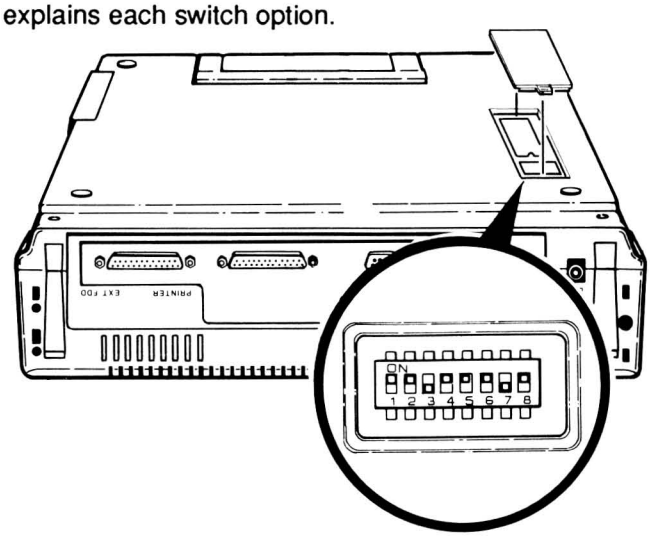


Figure 2-13. Configuration Switch Location

Table 2-1. Configuration Switch Settings

SWITCH SECTION	DESCRIPTION	SETTINGS
1	Left Drive Active LED	
	Hard disk drive	ON**
	Floppy disk drive	OFF*
2	Drive A location	
	Internal drive A location	ON*
	External drive A location	OFF
3	CPU clock speed	
	4.77 MHz	ON
	8 MHz	OFF*

Table 2-1 (cont'd.). Configuration Switch Settings

SWITCH SECTION	DESCRIPTION	SETTINGS
4	Reserved for future use	
5	Display width	
	80 characters	ON*
	40 characters	OFF
6	Display type	
	Internal (LCD)	ON*
	External (CRT)	OFF
7 and 8	Floppy Drive count	
	One drive	ON, ON**
	Two drives	OFF, ON*
	Three drives	ON, OFF
	Four drives	OFF, OFF

* = factory setting
** = factory setting for computers with hard disk drives.

Left Drive Active LED — The left Drive Active LED lights when the drive in location B is accessed. When this drive is a floppy disk drive, select the OFF position. If a hard disk is installed, select the ON position.

Drive A location — Some copy-protected software requires that drive A contain the distribution disk. Software distributed this way is on 5.25-inch disks, which you cannot use in the built-in 3.5-inch disk drives. If you need to use the external disk drive as drive A, select the OFF position. Select the ON position to use the 3.5-inch disk drive in location A as disk drive A.

CPU clock speed — Some software requires the original PC speed (4.77 MHz) for software-driven timing loops. If you have software that requires the slower speed to load or operate, select the ON position. For 8 MHz operation, select the OFF position.

Display width — You can select one of two display widths: 40 or 80 characters per line. Select the ON position for a display width of 80 characters per line. Select the OFF position for a display width of 40 characters per line. 80 characters per line is the most widely used width.

Display type — The video controller in the computer supports only one type of display at a time. Select the ON position to use the built-in display. Select the OFF position to use an external monitor.

Floppy Drive count — These two switches indicate the number of floppy disk drives connected to the computer. The external drives can be either 3.5-inch or 5.25-inch floppy disk drives.

Refer to Table 2-2 for the default drive name assignments.

Table 2-2. Drive Name Assignments

SWITCH SECTIONS 7 AND 8 SETTINGS	SWITCH SECTION 2 SETTING	INTERNAL DRIVE LOCATION A	INTERNAL DRIVE LOCATION B	EXTERNAL DRIVE LOCATION A	EXTERNAL DRIVE LOCATION B	HARD DISK DRIVE PARTITIONS 1	2	3	4
ON, ON	ON	A and B	—	—	—	C	D	E	F
OFF, ON	ON	A	B	—	—	C	D	E	F
ON, OFF	ON	A	B	C	—	D	E	F	G
OFF, OFF	ON	A	B	C	D	E	F	G	H
ON, ON	OFF	—	—	A and B	—	C	D	E	F
OFF, ON	OFF	B	—	A	—	C	D	E	F
ON, OFF	OFF	C	B	A	D	D	E	F	G
OFF, OFF	OFF	C	B	A	D	E	F	G	H

NOTE: If you access a disk that is not present, the system generates an error message.

Optional EMS Memory Configuration

Up to 1 megabyte of RAM with EMS (Extended Memory Specification) can be added to your computer. This memory expansion can only be installed by a qualified service technician.

Some software programs are designed to use this memory when available. Before this memory can be recognized and used by an application, a device driver (EMM.SYS) must be loaded into memory.

To load the EMM.SYS driver into memory, the command DEVICE = EMM.SYS must exist in a file named CONFIG.SYS and this file must exist on your bootable disk or in the root directory of the boot partition on your hard disk.

If the file CONFIG.SYS exists on your system, use a text editor to add the command to the file.

If the file does not exist, use a text editor or EDLINE to create the file.

Chapter 3
Hardware Operation

This chapter introduces you to the operation of certain parts of the hardware. The subjects include battery operation, keyboard operation, disk and disk drive theory and operation, introduction to modem operation, and video theory. Refer to Chapter 5 for a complete discussion of modem operation and programming.

Battery Operation

Your computer uses a rechargeable NiCad (Nickel-Cadmium) battery for portable operation. The length of time a computer will operate depends upon the computer's features and the battery's capabilities.

Nickel-cadmium batteries (NiCads) have three unique operating characteristics.

First, they are rechargeable. Rechargeable batteries, while having a higher initial cost, are less expensive to operate over an extended period of time.

Second, NiCads maintain their voltage level over the operating cycle of a single charge. Carbon, lithium, or alkaline batteries do not maintain this constant voltage level over their operating cycle. Instead their output gradually drops. This results in erratic performance and a shorter operating cycle than with NiCads of similar capacity.

The third unique characteristic is that NiCads can develop a severely reduced operating period if you do not care for them properly. This reduced operating period can be as short as 10 minutes. This usually happens if you do not operate the battery as long as possible before recharging it.

Battery Care

The nickle-cadmium battery pack is designed to provide full operating voltage and current for a maximum length of time. If battery charging is not correctly handled, you may experience shortened battery life. This section will describe how to remedy that problem.

All ni-cad batteries have an operating ledge, which is the point at which the ni-cad will no longer supply an operating current. This ledge is normally located at the point where the ni-cad reaches its lowest level of charge before it must be recharged. Under certain conditions this ledge will rise to the point that the battery can be operated only a short time (sometimes, as short as ten minutes) before it fails to deliver adequate voltage and current and must be recharged.

As long as you remember the following points, you should not experience high operating ledge levels in the battery pack.

- When you use the battery pack as a source of power for the computer, always use it as long as possible before plugging in one of the external power adapters or recharging the batteries.

- Charge a discharged battery pack from eight to twelve hours. Eight hours will fully charge the battery pack.

- You may operate the computer from an external power source without overcharging the battery pack. However, you should disconnect the adapter whenever you turn the computer off, unless you are charging the battery.

- If the computer is not going to be used for portable operation, remove the battery pack entirely and operate the computer only from an external voltage adapter.

Once the operating ledge has moved to a high level, the battery pack must be fully discharged, sometimes several times. The following procedure will cause the ledge to move back to its normal position, allowing you to use the battery to its fullest potential.

1. To fully discharge a ni-cad battery pack, operate the computer until the system shuts down. To provide the highest amount of battery drain, use the read boot track test from the built-in diagnostic test menu (refer to Chapter 19).

2. When the drive stops operating, continue to operate the computer until the low-power indicator stops glowing.

3. Recharge the battery for a minimum of 8 hours but not more than twelve hours.

At this point, you can attempt to use the battery pack for portable operation. However, if the battery pack continues to fail to deliver a normal operating life on a single charge, repeat the full discharge-recharge process three times.

Charging the Battery

Before charging the battery, completely discharge it. An easy way to do this is to perform a disk read test from the Monitor program. Allow the computer to run until the backlight ceases to function and the screen data becomes garbled.

The battery does not have to be attached to the computer to be charged. Plug the AC adapter into an AC power source and then into the battery pack, as shown in Figure 3-1. Charge the battery for 8 to 12 hours.

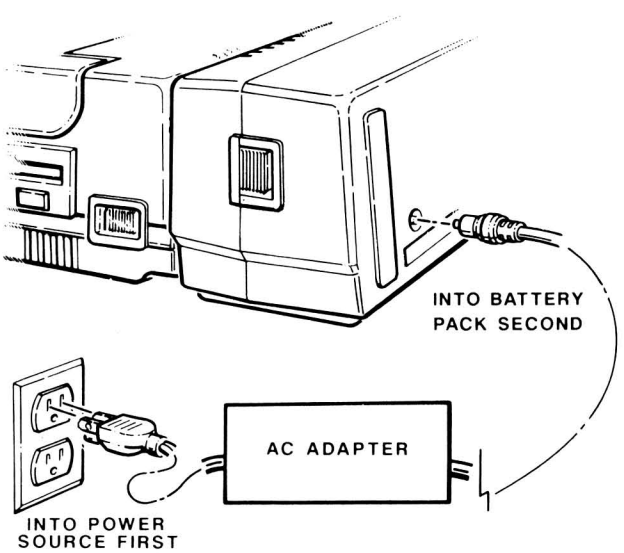


Figure 3-1 Charging the Battery

Getting the Most from the Battery

Several things shorten the effective operating time of the battery:

- Peripherals and accessories attached to computer
- Internal modem turned on
- Disk activity
- Use of the backlight
- Having a numeric coprocessor installed
- Having EMS memory installed
- Operating the computer at the high speed (8 MHz).

To get the most use from the battery:

- Disconnect all unnecessary peripherals and accessories.

- Turn off the modem when it is not being used. The modem is normally off at power-up. For information on how to turn the modem on and off, refer to Chapter 5.

- Keep disk drive use to a minimum. Some programs use a lot of disk input/output. For these programs, set up a virtual disk (sometimes called a RAM disk or memory disk). Refer to your operating system documentation for more information on virtual disks.

On computers equipped with a hard disk drive, the hard disk drive motor is set to run whenever the computer is powered on. The MODE command allows you to set the length of time, from 0 to 1200 seconds, the motor will continue to run after a disk access. Setting the time to 0 causes the motor to remain on all the time. To set the time, enter:

MODE MOTOR *n*

Where *n* is the number of seconds, from 0 to 1200, that the motor is to remain on after a disk access.

- Operate the computer at the slower (4.77 MHz) speed. Set configuration switch section 3 to ON, refer to Chapter 2, "Configuration."

Also, keep the display backlight set to the lowest level possible for comfortable operation. In some lighting situations, natural light will be enough for normal operation without the backlight. This is particularly true in bright sunlight.

The length of time the backlight remains lit after a key has been pressed can be varied from 0 to 10 minutes. The default time is two minutes. You can change the time with the MODE utility. Enter:

MODE EL*n*

where *n* is the number of minutes, from 0 to 10 that the backlight is to remain lit after a key is pressed.

A numeric coprocessor significantly reduces the battery operating time, but do not remove the coprocessor to conserve battery power. If you are going to be using the computer primarily on battery power, consider not installing a numeric coprocessor. EMS memory will also cause a reduced operating time during battery operation. However, if the application program being run does not access EMS memory, the EMS memory enters an idle state and requires less power.

When the battery power is low, the Low Power indicator starts flashing and an audible tone pattern is heard (three beeps every 30 seconds). You have about ten minutes to complete and save your work when this happens.

Keyboard

The keyboard contains 78 keys. Most generate repeated entries. If you hold a key down for more than a half-second, the character starts repeating at about 10 times a second. The following keys do not generate repeated entries:

- ALT
- CAPS LOCK
- CTRL
- NUM LOCK/PAUSE
- SCROLL LOCK/BREAK
- SHIFT

Alphanumeric Keys

The alphanumeric keys consist of several groups. Refer to Figure 3-2 for the following discussions.

Alphabetic characters — Use the alphabetic keys to enter the standard 26 letters of the alphabet as either lower- or uppercase letters. To enter uppercase letters, hold down either SHIFT key or activate the CAPS LOCK mode.

NOTE: The SHIFT keys reverse the action of the CAPS LOCK key. If the CAPS LOCK mode is active, the SHIFT keys produce lowercase characters. If the CAPS LOCK mode is not active, the SHIFT keys produce uppercase characters.

Numeric characters — Use the keys 0 through 9 to enter numbers. The CAPS LOCK mode does not affect these keys.

Punctuation characters — Use these keys to enter punctuation characters. With keys that have more than one symbol, use the SHIFT keys to generate the upper character.

Special characters — Most typewriters do not have the special characters found on the computer keyboard. With keys that have more than one symbol, use the SHIFT keys to generate the upper character.

The special characters include the following:

- The tilde (~)
- The grave accent (`)
- The caret (^)
- The vertical bar (|)
- The backslash (\)
- Opening bracket ([)
- Closing bracket (])
- Opening brace ({)
- Closing brace (})
- The greater-than symbol (>)
- The less-than symbol (<)

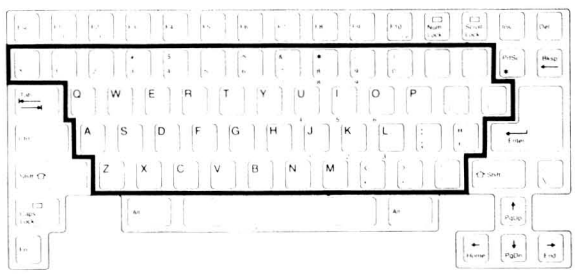


Figure 3-2. Alphanumeric Keys

Control and Special Purpose Keys

The remaining keys provide control over the computer, the screen, and the keyboard. The following material describes the normal function of each key. However, software can direct almost any key to cause a defined function to occur. Software documentation usually describes such functions. Refer to Figure 3-3 for the following discussions.

NOTE: The control keys (CTRL, ALT, FN, and SHIFT) modify the codes produced by other keys.

ESC — The escape key performs special functions when you press it and another key in sequence. You do not have to hold the ESC key down after you press it.

Function keys (F1 through F12) — In some programs, you use these keys to perform special functions.

LCD/CRT — You use this key with the FN key to toggle the video between the LCD video display and the CRT connector.

NUM LOCK — This key provides two functions. When used with the FN key (FN-NUM LOCK), it toggles the keypad lock. The LED lights when the keypad lock is active. When the keypad lock is on, the NUM LOCK key toggles the keypad between the numeric mode and the cursor control mode.

PAUSE — You use this key in some programs for special functions.

SCROLL LOCK — You use this key in some programs to control screen movement. The LED lights when the scroll lock mode is active.

BREAK — You use this key with the CTRL key in some programs to interrupt operation.

INS — You use this key in most text editing and word processing packages to insert material. You also use it with the CTRL and ALT keys to enter the Monitor program.

DEL — You use this key in most text editing and word processing packages to delete material. You also use it with the CTRL and ALT keys to reset the computer.

TAB — Use this key to move the cursor to the next tab column. Operating systems usually set tab columns every eight spaces. Most software programs, especially word processing programs, modify this value.

CTRL — The control key provides additional codes when you hold it down and press another key. Most of the time, you will see instructions to use this key expressed as a combination of keys. For example, CTRL-C tells you to press and hold the CTRL key while you press the C key.

SHIFT — The SHIFT key provides uppercase characters when you hold it down and press an alphabetic key. It is also used to generate the upper character on keys with more than one symbol.

CAPS LOCK — Use this key to toggle the CAPS LOCK mode. When the CAPS LOCK mode is on, the LED lights and the alphabetic keys produce uppercase letters. When the CAPS LOCK mode is off, the alphabetic keys produce lowercase letters. The SHIFT keys reverse the action of the CAPS LOCK mode.

FN — The function key acts similar to the CTRL key, providing additional codes when you hold it down and press another key. The FN key causes the keys with blue labels to generate those characters and functions.

ALT — The alternate key acts similar to the CTRL key, providing additional codes when you hold it down and press another key. Most of the time, you will see instructions to use this key expressed as a combination of keys. For example, ALT-C tells you to press and hold the ALT key while you press the C key.

SPACE BAR — Use this key to enter a blank character (space).

PRT SC — The print screen key sends the currently displayed information to the printer.

BK SP — Use this key to move the cursor one space to the left. Software usually also erases the character to the left. The cursor is an indicator on the display that lets you know where the next character will appear. Software controls the shape of the cursor.

ENTER — Use this key to return the cursor to the left side of the display. Usually, software adds a line feed, moving the cursor to the beginning of the next line on the screen. Most software packages use this key to tell the program that entry of data or instructions is complete.

Cursor keys (←, ↑, ↓, →) — In some programs you use these four keys to move the cursor around the screen or through a file.

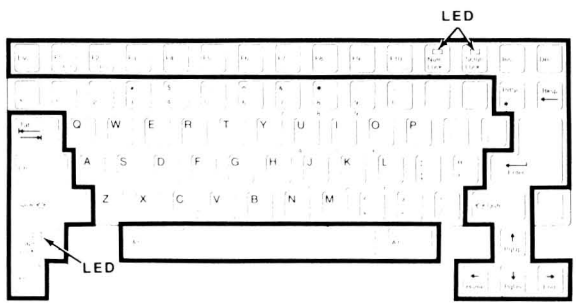


Figure 3-3. Control and Special Purpose Keys

Keyboard Operation

The keyboard operates in one of the four modes described in the following paragraphs. Some software programs create special keyboard modes not described in this manual.

Unshifted keyboard mode — The unshifted keyboard mode generates lowercase alphanumeric codes. The SHIFT keys cause the keys to generate uppercase letters or the top character printed on the key. The FN key causes the keys with blue labels to generate those characters and functions.

CAPS LOCK mode — The CAPS LOCK mode generates uppercase alphabetic codes for the alphabetic keys. It does not affect the numeric/punctuation keys. The SHIFT keys reverse this action: alphabetic keys produce lowercase letters and the numeric/punctuation keys produce the upper character. The FN key causes the keys with blue labels to generate those characters and functions. The LED lights when the CAPS LOCK mode is active.

Keypad lock modes — Refer to Figure 3-4 for the Portable computer's keypad. The FN-NUM LOCK key combination toggles the keypad lock on and off. When the keypad lock is active, the FN key causes the keypad keys to generate alphanumeric characters. The key codes produced are dependent upon the state of the CAPS LOCK mode. The indicator (LED) lights when the keypad lock is on.

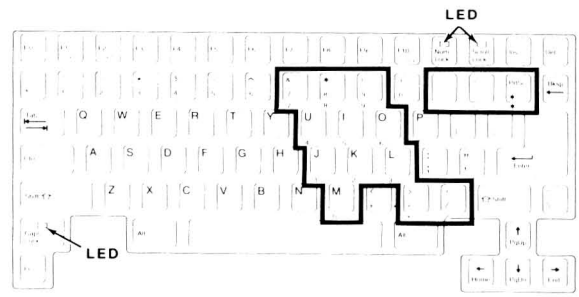


Figure 3-4. Portable Computer's Keypad

When the keypad lock is active, the NUM LOCK key toggles the keypad numeric/cursor modes. There is no indicator for the numeric/cursor modes. The SHIFT keys reverse the action of the NUM LOCK key. If the keypad is in the numeric mode, press the SHIFT keys to generate cursor movement codes. If the keypad is in the cursor mode, press the SHIFT keys to generate numeric codes.

- Numeric mode — The first time you activate the keypad lock mode, the numeric mode is active. The numeric mode causes the keys with blue labels to generate numeric codes.

- Cursor mode — The cursor mode causes the keys with blue labels to generate cursor movement codes.

Table 3-1 lists the result of pressing a keypad key in the unshifted mode, with the SHIFT (or CAPS LOCK) key, with the FN key, and with the SHIFT and FN keys for each of the possible modes of the keypad: inactive, numeric, and cursor. Figure 3-5 illustrates the cursor functions of the keypad keys.

Table 3-1. Keypad Results

KEYS PRESSED	RESULT: NORMAL KEYBOARD[1]	RESULT: NUMERIC MODE[2]	RESULT: CURSOR MODE[2]
Keypad key	Lowercase letter or number printed in black.	Number or symbol printed in blue.	Generates cursor function. blue.
SHIFT-keypad key	Uppercase letter or symbol printed in black.	Generates cursor function. blue.	Number or symbol printed in blue.
FN-keypad key	Number or symbol printed in blue.	Lowercase letters or number printed in black.	Lowercase letter or number printed in black.
SHIFT-FN-keypad key	Generates cursor function.	Uppercase letters or symbol printed in black.	Uppercase letters or symbol printed in black.

NOTES:
1. Keypad inactive.
2. Keypad active.

NOTE: An optional external keypad is available as as alternative to using the keypad on the keyboard.

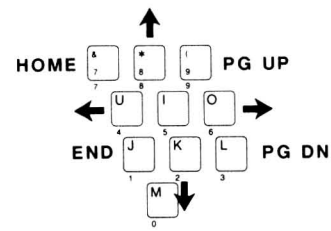


Figure 3-5. Keypad Cursor Functions

The computer has two other special modes: the smart keyboard mode and the straight keyboard mode. These affect the way some software packages interact with the keyboard, but not the codes generated.

Smart keyboard mode — The computer enters the smart keyboard mode when you first turn on the computer or press the FN-1 key combination. This mode causes all functions of the keyboard to function as labeled. The four cursor control keys generate a prefix code in addition to their normal code. This prefix code tells the Monitor program that you pressed a cursor control key. The Monitor program then checks the keyboard mode and, if software allows, generates the appropriate codes to execute the intended cursor movement. However, this action can cause the keyboard to drop out of sync with any onscreen mode indicators.

Straight keyboard mode — When you press the FN-2 key combination, the computer enters the straight keyboard mode. This mode does not check the keyboard mode before it generates the code for the key pressed. You could call this a dumb mode since the computer ignores the mode-checking operation of the smart keyboard mode. In this mode, the four cursor control keys can produce incorrect results when you toggle the keyboard from one mode to another. Therefore, do not toggle the other keyboard modes while the straight keyboard mode is active.

Some programs manipulate the keyboard codes by intercepting the up and down codes for each key. These programs can leave the keypad lock and numeric/cursor modes in an unknown state with respect to the keyboard. To resynchronize the keyboard and these modes, press FN-(space bar).

Three function key combinations affect video operation. The first combination is FN-F10, which toggles the video output between the RGB connector and the LCD. The other two combinations (FN-F8 and FN-F9) toggle the LCD through eight display palettes. These two key combinations do not affect the normal color video palettes.

The video system in the Portable computer is functionally equivalent to the color graphics adapter in PC computers. Some software checks for color and automatically goes into a color display mode. Although the controller can produce an effective gray scale on the LCD, not all colors are dark enough to be readable. Eight LCD display palettes allow you to select a gray scale that makes the colors used in a software package more visible. Since colors used varies from one program to another, a pattern that works for one program may not necessarily work for another. To see how these keys operate, press CTRL-ALT-INS to place the computer in the Monitor program. Then press C to call up the color bar and use the FN-F8 and FN-F9 key combinations to change the gray scales.

Table 3-2 contains a summary of the various keyboard special features.

Table 3-2. Keyboard Special Features

KEY COMBINATION	DESCRIPTION
FN-(blue labels)	This key combination generates the function printed in blue. If the keypad lock is active (LED lit), this combination generates the alphanumeric code for the keypad key you pressed.
FN-NUM LOCK	This key combination toggles the keypad lock on and off. The keypad lock establishes the keys in Figure 3-3 as a numeric keypad emulator. The LED lights when the keypad lock is on.
FN-1	This key combination places the keyboard in the smart keyboard mode. In this mode the cursor control keys always function as cursor control keys, regardless of other keyboard modes that can be active.
FN-2	This key combination places the keyboard in the straight keyboard mode. This mode provides straightforward operation and ignores the need for intelligent operation of control keys.
FN-(space bar)	This key combination resynchronizes the keyboard with the keypad lock mode and the cursor control keys.
FN-F8 or FN-F9	These two combinations toggle the screen (not video) palette. You can move the LCD display palette selection forward (FN-F8) or backward (FN-F9) through eight different gray-scale palettes.

Keyboard Functions

Table 3-3 describes the actions caused by the control key functions of the Portable computer. The words "software dependent" in the description indicate those key combinations that are dependent on software to produce an action.

NOTE: Some computer publications discuss pressing control key combinations simultaneously. Actually, the key combination indicates that you should press and hold the control key and then press the other keys in sequence. After you press all the keys, you can release them simultaneously.

Table 3-3. Control Key Functions

KEY SEQUENCE	DESCRIPTION
CTRL-ALT-DEL	Reset computer to autoboot sequence.
CTRL-ALT-INS	Reset computer to Monitor program.
CTRL-BREAK	Stops operation (software dependent).
CTRL-PAUSE	Pause (software dependent).
DEL	Delete (software dependent).
ESC	Defeats autoboot (during boot procedure).
FN-(down arrow)	Page down (software dependent).
FN-(keypad keys)	Numeric keypad mode code.
FN-(left arrow)	Home (software dependent).
FN-(right arrow)	End (software dependent).
FN-(up arrow)	Page up (software dependent).
FN-1	Activates smart keyboard mode.
FN-2	Activates straight keyboard mode.
FN-BACK SPACE	Blank (software dependent).
FN-ESC	System request (software dependent).
FN-F1	F11 (software dependent).
FN-F2	F12 (software dependent).
FN-F8	Changes LCD gray-scale palette.
FN-F9	Changes LCD gray-scale palette.
FN-LCD/CRT	Toggles LCD/CRT output.
FN-(Spacebar)	Synchronizes the keyboard with the cursor control keys.
FN-NUM LOCK	Toggles numeric keypad lock.
FN-SHIFT-(keypad keys)	Keypad cursor control mode code.
INS	Insert (software dependent).
NUM LOCK (keypad keys)	Toggles numeric/cursor control code.
PRTSC	Print screen.
SCROLL LOCK	Scroll lock (software dependent).

Disk Drives and Disks

The following paragraphs describe the general concepts behind disk drive systems. The drives described in this section are typical of the type supplied in most computers.

Disk Drives

This computer supports 3.5-inch floppy disk drives, external 5.25-inch floppy disk drives, and a hard disk drive. Most operating systems, such as MS-DOS, identify these drives with drive names. Typically, the first floppy drive is drive A, the second floppy drive is drive B, and so on. Likewise, the operating system gives hard disk partitions drive names. Hard disk partition names start at drive C, D, or E, depending upon the number of floppy disk drives used. In the Portable computer, the configuration switch settings identify the drive name assignments. Refer to Chapter 2 for the identification of the drive names. Figure 3-6 identifies disk drive locations.

The 3.5-inch disk drives used with this computer are double-sided, double-density disk drives. The disk drive records eighty soft-sectored, 2-track cylinders at 135 tpi (tracks per inch). The disk rotates at 300 rpm and the storage capacity is 720K of information.

The 5.25-inch disk drives used with the computer are also double-sided, double-density disk drives. The disk drive records forty soft-sectored, 2-track cylinders at 48 tpi. The disk speed is 300 rpm and storage capacity is 360K of information.

The hard disk drive used with the computer is a fixed-disk design. The capacity of the drive is 20M of information.

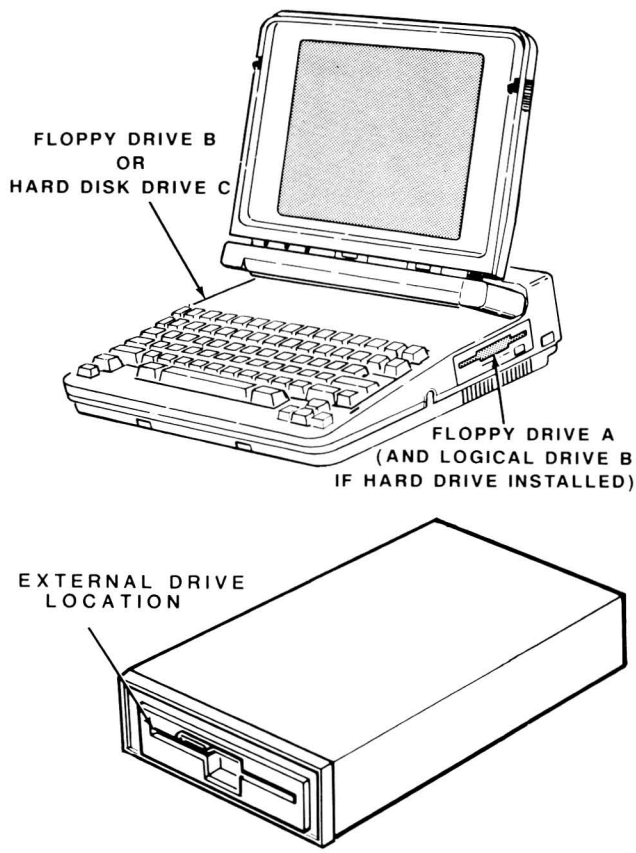


Figure 3-6. Disk Drive Locations

Disk drives contain one or more read/write heads. The design of these heads provide low noise levels and high performance. Pairs of heads face each other, with the disk rotating between them. When you move the computer or the external drives, place shipping inserts into the drives and close them. Otherwise, the heads could damage each other through vibration or jarring. The hard disk system automatically moves the heads to a safe location upon power down.

Disks

Refer to Figure 3-7. Four parts make up the 3.5-inch disk: the disk itself, the disk liner, the disk case, and a sliding metal cover.

The disk liner cleans the disk and traps dust particles. The disk case has three openings, one of which is under the sliding metal cover. When you place the disk in the disk drive, the drive moves the metal cover to one side. This allows the drive's read/write heads to access the disk. The disk drive spindle grips the disk's metal drive hub. The metal hub's design maintains a fixed drive-to-disk relationship.

At one corner of the disk is a small hole and movable tab. When the hole is open, the disk is write protected and you cannot write data to the disk. When you close the hole, you can write data to the disk.

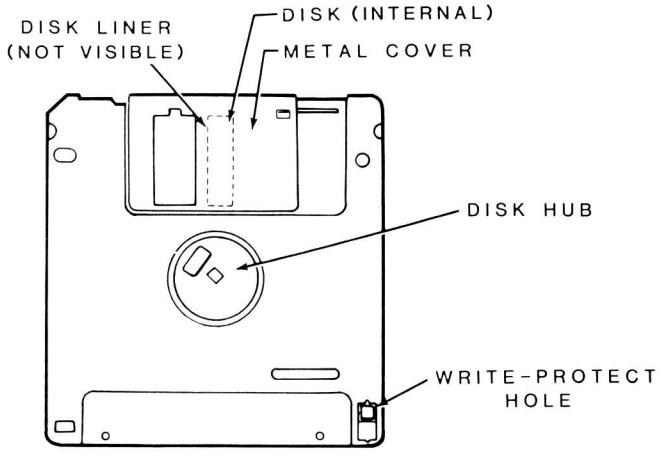


Figure 3-7. A 3.5-Inch Disk

Refer to Figure 3-8. Three parts make up the 5.25-inch disk: the disk itself, the disk liner, and the disk jacket. A disk envelope provides additional protection when the disk is not in the disk drive.

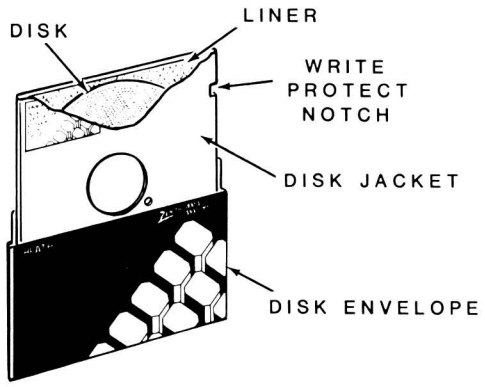


Figure 3-8. A 5.25-Inch Disk

The disk liner cleans the disk and traps dust particles. The outer jacket has three openings and a notch in one side. The disk drive spindle grips the disk through the large center opening. The small circular hole is a timing hole. The read/write heads access the disk through the long slot.

The notch on one side of the disk is the write-protect notch. When you cover the notch, the disk is write protected. When you expose the notch, you can write data to the disk.

Refer to Figure 3-9. The hard disk is not removable. It consists of a precision metal disk called a platter. The read/write heads float above the disk as it turns. In contrast, the heads of floppy disk drives actually ride on the surface of the disk. The heads of both systems function the same.

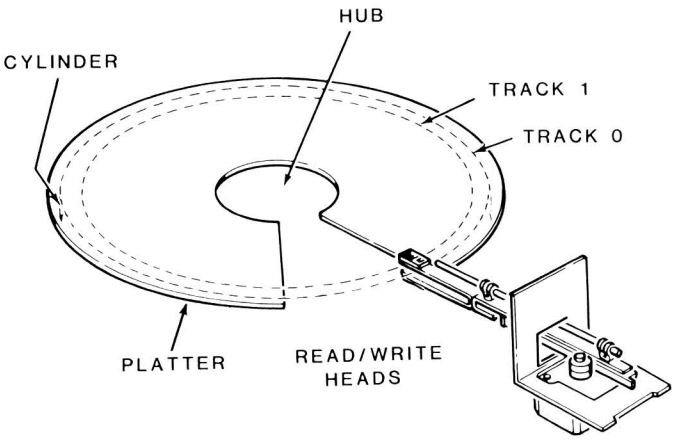


Figure 3-9. The Hard Disk System

Since the disk itself is very hard, you can easily damage the surface of the disk and the heads. To minimize damage to a disk system, manufacturers created a safe parking cylinder. This location on the disk minimizes head and disk damage if somebody jars the hard disk drive while they are moving it. In the case of portable computers, such as this one, the system automatically moves the heads to the shipping cylinder. This happens when you turn the computer off and whenever you have not used the disk for more than a few seconds.

NOTE: The MS-DOS version supplied with this computer provides a **Mode Motor** command for powering down the hard disk drive after a specified period of non-use. This conserves battery power. Refer to the MS-DOS manual for further information.

The materials used to manufacture the surface of the disk is similar to audio and video tape. Floppy disks have a base of very thin plastic material. Hard disks are a carefully machined and polished non-magnetic material. A magnetic substance covers the surface of the disk. The read/write heads record on this magnetic material and read from it in a manner similar to audio recording and playback.

Disk Organization

Cylinders, tracks, and sectors divide the disk's surface into small, workable parts. Refer to Figure 3-10 for an illustration of these parts.

A track is that portion of the disk that passes under a read/write head. Most disk systems contain more than one read/write head. Each head is in alignment with the heads for the other platters or sides and all heads move at the same time. Where there is more than one head, all the tracks that pass under heads at the same time make up a cylinder. Therefore, a cylinder has as many tracks as there are heads in the disk system.

Software divides each track into sectors. Software also establishes the timing for the sectors through one of the following means.

- In 5.25-inch floppy disk drives, software looks for a timing hole in the disk itself.
- In 3.5-inch disks, software looks at the position of the drive mechanism.
- In some hard disk systems, software looks for timing information recorded on a special track.
- In other hard disk systems, software looks at the position of the drive mechanism.

The amount of information each sector can hold determines the disk's capacity. Other factors, including disk size, number of tracks, and track spacing, determine the amount of data you can store on a disk.

The Portable computer uses double-density disk drives that require double-sided, double-density disks.

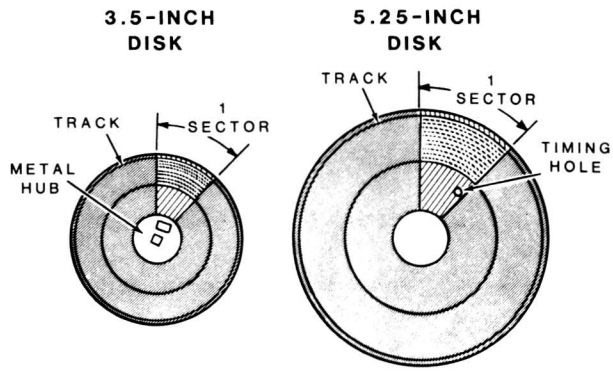


Figure 3-10. The Disk Surface

Data Encoding Methods

The Portable computer uses two methods to encode data on the disk. Single-density recording uses FM encoding and double-density recording uses MFM encoding.

FM (Frequency Modulation) encoding, illustrated in Figure 3-11, is the simplest form of encoding. It records fixed clock pulses and a data bit pulse. If the data bit is 1, the system records a pulse. If the data bit is zero, the system does not record a pulse for the data bit.

Figure 3-11 also illustrates MFM (Modified Frequency Modulation) encoding. This method records twice as many data bits as FM encoding in the same disk area. However, the number of actual pulses recorded is no greater than FM encoding. This is because MFM encoding records the clock pulses only under certain conditions. When the system does not record a data pulse in either the present or preceding bit cells, it records a clock pulse.

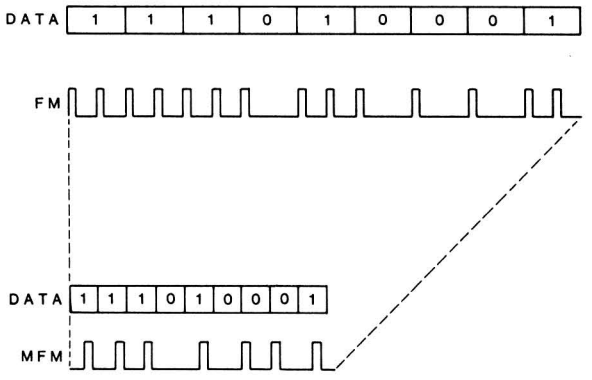


Figure 3-11. Data Encoding Methods

Recording data on a disk is similar to recording audio or video signals on tape. Recording FM encoded digital signals on a disk requires two frequencies: 62.5 kHz and 125 kHz. The higher frequency records adjacent (clock and data bit) pulses while the lower frequency records non-adjacent (clock only) pulses. MFM encoding modifies only the data and clock-to-frequency relationships, not the recording or playback method.

Figure 3-12 illustrates writing pulses to and reading pulses from a disk. All cores in a read/write head have coils wound at some point on the rings. When current flows through the coil, it creates a flux field at the gap in the ring. As the disk's surface passes the gap, the flux field magnetizes the recording surface in a longitudinal manner. Both erase and read/write coils operate in the same manner. The current flow in the erase coil remains stable, erasing the surface of the disk that passes under it. The current flow reverses in the read/write coil, recording the clock and data pulses on the surface of the disk.

The erase coils energize during the write operation. These two cores "trim erase" the track being recorded so that it does not exceed 0.006 inches in width on 48-tpi, 5.25-inch disks. The width on disks with a higher track-per-inch density is smaller. Trim erasing (sometimes called tunnel erasing) prevents minor recording current deviations from affecting material recorded on adjacent tracks of the disk.

During a write operation, the system records a pulse whenever the magnetic field produced by the read/write core reverses. This field reversal occurs almost instantaneously in comparison to the motion of the disk. Each time a reversal occurs, the gap magnetizes the disk surface in the opposite direction from the preceding disk surface. The point of reversal on the disk represents a pulse.

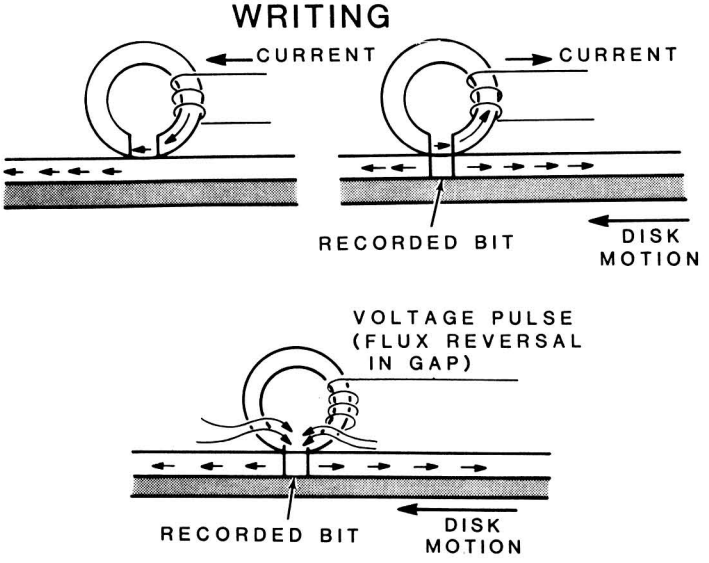


Figure 3-12. Recording and Playing Back Signals

During a read operation, the magnetized surface of the disk produces a constant field at the gap in the core. Whenever the magnetized surface changes direction, signifying a recorded pulse, the field reverses. This induces a current in the coil on the core. The disk drive electronics converts this current into a pulse that the disk controller decodes as a clock or data pulse.

The method of recording and playing back the pulses varies from manufacturer to manufacturer. However, the principles remain the same, regardless of the manufacturer. Therefore, signals recorded on one disk drive are readable by disk drives that use the same disk size, track density, and encoding method.

Disk Care

Since floppy disks are thin, you can permanently distort them, making them unusable. Therefore, keep the following in mind when you handle disks:

- When you prepare a label for a disk, write on it before you place it on the disk. If you must write on a label that is already on the disk, use a felt-tip pen.

- Make backups of all disks that contain important programs and information.
- Store backup disks and distribution disks away from the computer.
- Store the disk in its protective envelope or a container in an upright position.
- Keep disks away from heat or direct sunlight when they are not being used.
- Do not bend the disk or use paper clips on it.
- Do not touch any exposed areas of the disk or attempt to open the metal shield on 3.5-inch disks.
- Keep the disk away from magnets and magnetized objects, including power supply adapters and telephones.

Modem

Two modems are available for the portable computers, a 1200 and a 2400 baud modem. The modems have the following features:

- Hayes AT command set compatibility
- Bell 103/212A and CCITT V.22 synchronous modem specifications and protocol
- Touch-tone or pulse dialing
- Automatic or manual answering
- Programmable registers
- 40-character command line
- Call progress monitoring
- Automatic speed and parity detection
- Automatic dialing

The modem operates in one of two modes: command or on-line.

In the command mode, you can enter any command from the command set. The command set allows you to control the modem through your own software. After the modem receives a command, it returns a response code that indicates the result of the command. You can obtain additional modem control by programming the modem's registers. Refer to Chapter 5 for information about turning the modem on and off and on programming and operating the modem.

In the on-line mode, the modem transmits and receives data through a telephone line. Since telephone communication is bidirectional, you can use full duplex operation. Your software must accommodate software handshaking.

If you use the modem with communications software, refer to the software documentation for operation.

Video Systems

The two different display types used with this computer, LCD and external CRT monitor, are raster-scan devices. The computer creates the image on the screen by scanning a predetermined pattern of lines. Although the LCD and CRT displays operate differently, they both produce recognizable images by activating individual points called pixels (picture elements). The CRT system produces light while the LCD blocks light as the computer scans the display. The display of the LCD will be the reverse of the display on the CRT.

The screen of the displays consists of a matrix of pixels 640 columns wide by 200 rows deep. Each pixel forms part of the display. A matrix of eight pixels wide by eight pixels deep forms each displayed character (letter, number, or punctuation mark, or symbol). This results in a display 80 characters wide by 25 characters deep. The aspect ratio of both are the same.

CRT Operation

Figure 3-13 illustrates the scanning of a typical CRT. The display area is the portion of the CRT where the user may actually see the data being displayed. In the Portable computer this area is 640 pixels wide by 200 pixels high. Electronic circuits in the monitor supply the voltages that fire a beam of electrons at a phosphors coating on the CRT's inside face. The beam, as it strikes the phosphors, agitates or excites them, producing visible light for each pixel. The color of the phosphor determines the color of the pixel. In monochrome monitors and televisions, there is only one type of phosphor and only one electron beam. The intensity of the beam determines the amount of light given off by each pixel. This provides the monitor with the ability to produce different shades of light, called a gray scale.

In color monitors and televisions, there are three beams firing at three differently colored phosphors deposited on the inside face of the CRT. These phosphors produce the red, green, and blue colors when struck by the appropriate beam. The monitor displays the different colors you see by firing the beams in varying intensities at corresponding phosphors located at one pixel location.

In the illustration, the border area is that area outside the active display area. The phosphor coating does not always extend to the edge of the tube. The electron beam does not illuminate any area not coated with phosphors.

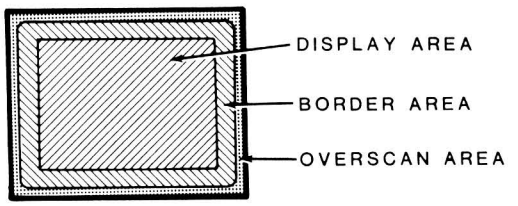


Figure 3-13. Scanning the Face of the CRT

Scanning the CRT actually exceeds the area coated by phosphors, and often exceeds the width and height of the CRT. This overscan area, illustrated in Figure 3-11, allows the use of the full CRT screen for display purposes. You cannot see the beginning and end of the synchronization pulses because they take place outside the overscan area.

LCD Operation

Liquid crystal displays do not emit light. Instead, they depend upon two different methods to produce a readable display: reflected light or light sent through the LCD. The majority of clocks, including watches, that have LCDs use room light that reflects through the LCD to produce a display. Even at night, when you press the buttons on your watch to light the display, the light reflects through the LCD. Some newer LCD displays, such as those used in Zenith portables, send light directly through the LCD. By selectively blocking this light and forcing the display to block or transmit light, the LCD produces an image.

The LCD displays used in Zenith portables are supertwist displays. They provide 270° of twist compared to the 220° of a standard LCD. This provides a sharper, brighter display.

One of the materials used in liquid crystal displays remains in a state between a liquid and crystalline (solid) state. This material remains in its liquid crystal state through a wide temperature range (32° – 140° F or 0° – 60°C). When the material receives an electrical charge, the electrons in the material align themselves so light can pass through. When the electrons do not align the material is opaque (polarized) and light cannot pass through it. Light passes through the material while in a charged transparent state. The uncharged opaque areas become visible to the viewer. High polarization angles and a very fine grid produce high contrast and a wide viewing angle. Backlighting makes it possible to use the display under all lighting conditions.

In this computer, the liquid crystal material is encased between two transparent plates. One of the plates contains an etched matrix 640 columns wide by 200 rows deep. The matrix consists of an electrically conductive material. Row and column drivers on the LCD circuit board drive the matrix. The drivers provide the polarizing voltages to charge the liquid crystal material for any pixel and make it transparent. The video controller operates the drivers. The controller addresses (scans) the drivers by column and row, starting at the uppermost and leftmost point on the screen, moving to the right until it reaches the end of a row. The scan then returns to the beginning of the next row, and so on, scanning the entire screen. The voltages precisely control the alignment of the material, partially turning on the pixel. This produces the gray scale display. There are ten display palettes available to allow you to select a gray scale that will provide the most visible display. Use the FN-8 or FN-9 key combination to change the gray scale.

Chapter 4

The Monitor Program

The MFM-180 (Multi-Function Monitor) program maintains software compatibility with desktop PC-compatible computers. During the initial power-up sequence, the Monitor program performs tests to make sure the computer is ready to function. These include tests of major circuits in the computer. If the program detects a malfunction, it sends one or more messages to the system's display to alert the operator. Refer to Chapter 19 for an explanation of these error messages.

When the program successfully finishes the tests, it starts a routine called autoboot. This routine attempts to load the disk operating system from the first drive, drive A. If no disk is in drive A, the routine attempts to load the operating system from the hard disk drive. If the program cannot load the operating system, it will display an error message. Refer to Chapter 19 for an explanation of the error messages.

To reach the Monitor program, you can bypass (or defeat) the autoboot program. To defeat autoboot, press the ESC key after you first turn on the system. The screen will show the monitor prompt (->). You can also access the Monitor program by pressing CTRL-ALT-INS. The screen will display the Monitor program opening message, similar to the following:

```
MFM-180 Monitor, version 2.0
Memory Size: 640K
Press "?" for help.
->
```

NOTE: The version number will vary from the one in this example.

The Monitor program has six basic features:

- Automatic power-up self-tests
- User-executed tests
- Video commands
- Disk boot routines
- A machine language debugger
- A BIOS (Basic Input/Output System)

You read about the power-up self-tests in Chapter 1. Refer to Chapter 19 for additional information on the user-executed tests and Chapter 6 for the machine language debugger and BIOS.

Commands

Once the monitor prompt appears on the screen, you can display a summary of the Monitor program. To display the Monitor command summary, enter a question mark and press the ENTER key. You will see a display similar to the one shown in Figure 4-1.

```
                    - MFM-180 Command Summary -

CMD:     Explanation            Syntax
----     -----------            ------
?        Help                   ?
B        Boot from disk         B [{0|1|2|3}]
C        Color Bar              C
D        Display memory         D [<range>]
E        Examine memory         E <addr>
F        Fill memory            F <range>,{<byte>|"<string>"}...
G        Execute (Go)           G [=<addr>][,<breakpoint>]...
H        Hex math               H <number1>,<number2>
I        Input from port        I <port>
M        Move memory block      M <range>,<dest>
O        Output to port         O <port>,<value>
R        Examine Registers      R [<register>]
S        Search memory          S <range>,[<byte>|"<string>"}...
T        Trace program          T [<count>]
U        Unassemble program     U [<range>]
V        Set Video/Scroll       V [M<mode>][S<scroll>]
TEST     Extended diagnostics   TEST
Where    <range> is:            <addr>{<addr>|L<length>}

         Copyright (C) 1987, by Zenith Data Systems
```

Figure 4-1. MFM-180 Command Summary Menu

The Monitor program contains four general user commands and a machine language debugger. The four general commands are summarized in Table 4-1 and described in the following paragraphs.

Table 4-1. General User Commands

COMMAND	DESCRIPTION	SYNTAX
?	Help	?
C	Display color bar	C
V	Set video/scroll mode	V [M<mode>] [S<scroll>]
B	Boot disk	B [F|W][0|1|2|3][:<partition>]

Help

Syntax: ?

Example: ? *ENTER*

Use the help command to display a summary of the Monitor program commands. The summary lists each Monitor program command, followed by an explanation and the syntax for entering the command. If you enter the example, the Monitor program will display the command summary illustrated in Figure 4-1.

Display Color Bar

Syntax: C

Example: C *ENTER*

Use the color bar command to display a set of 16 color bars. You can use the color bars to adjust a color or monochrome monitor and to adjust the contrast and brightness controls of the built-in LCD. Use the FN-F8 and FN-F9 key combinations to adjust the LCD palette. If you enter the example, the screen will clear and display the color bar pattern.

Set Video/Scroll Mode

Syntax: V [M<mode>] [S<scroll>]

Examples:
```
V M3 ENTER
V S0 ENTER
V M6 S2 ENTER
```

Use the set video/scroll mode command to set the video and scroll modes described in Table 4-2. Seven video modes are available, along with three scrolling modes. The first example sets the display to video mode 3 (color, 80 characters by 25 rows). The second example sets the scrolling mode to 0 (software-controlled scrolling). The third example sets the video mode to 6 (monochrome graphics 640 by 200 resolution) and scrolling mode 2 (hardware-controlled smooth scrolling).

Table 4-2. Video and Scroll Modes

MODE	DESCRIPTION
M0	40 characters by 25 rows text mode, monochrome display on the LCD screen or RGB output.
M1	40 characters by 25 rows text mode, color display at the RGB output or as a gray-scale display on the LCD screen.
M2	80 characters by 25 rows text mode, monochrome display on the LCD screen or RGB output. You can scroll individual video pages without affecting other video pages.
M3	80 characters by 25 rows text mode, color display at the RGB output or as a gray-scale display on the LCD screen. You can scroll individual text pages.
M4	320 × 200 pixel resolution graphics mode, color display at the RGB output or as a gray-scale display on the LCD screen. Text is 40 characters by 25 rows.
M5	320 × 200 pixel resolution graphics mode, monochrome display on the LCD screen or RGB output. Text is 40 characters by 25 rows.
M6	640 × 200 pixel resolution graphics mode, monochrome display on the LCD screen or RGB output. All three scrolling modes are available. Text is 80 characters by 25 rows.

Table 4-2 (Continued) Video and Scroll Modes

MODE	DESCRIPTION
S0	Software scrolling. PC-compatible software commonly uses this mode. It works in all video modes. When software scrolls the material, the display moves a line at a time.
S1	Hardware jump scrolling. This mode is available for video modes 3 through 6 and is similar in action to software scrolling. However, hardware handles the scrolling, instead of software.
S2	Hardware smooth scrolling. This mode works only in video mode 6. When software scrolls the material, the display moves a partial line at a time. This provides a smooth appearance to the scrolling action.

Boot Disk

Syntax: B [F|W][0|1|2|3][:<partition>]

Examples:
```
B ENTER
BF1 ENTER
B3 ENTER
BW0:1 ENTER
```

Use the boot disk command to boot the operating system manually from any disk drive attached to the computer.

If you enter the first example, the program boots from the default drive. When you first turn on the computer, drive A is the default drive. After you manually boot from another disk drive, that drive becomes the default drive until you turn the system off or boot from another drive.

If you enter the second example, the program boots from drive B if it is present. The F in the syntax is optional and specifies floppy disk drives. The letter W specifies hard disk drives.

If you enter the third example, the program boots from drive D, the second external disk drive attached to the system.

If you enter the fourth example, the program boots from the first partition on the hard disk.

Part II

Firmware

Chapter 5

Modem Operation

There are two modems (1200 and 2400 baud) that may be installed in the computer. Commands for both modems are explained in this section of the manual. In cases where the information is identical for both modems, it is presented together. If there is a difference in a command or instruction, it is labeled (1200 only) or (2400 only). The modems have the following features:

- Hayes AT command set compatibility
- Bell 212A and 103 synchronous modem specifications and protocol
- CCITT V.22 synchronous modem specification and protocol (2400 only)
- Touch-tone or pulse dialing
- Automatic or manual answering
- Software programmable registers
- 40-character command line
- Call progress monitoring
- Automatic speed and parity detection
- Automatic dialing

NOTE: You must use the modem with communications software; refer to the software documentation for operation.

The modem operates in one of two modes: command or on-line. Your communications software will allow you to establish communications in the on-line mode.

In the on-line mode, the modem transmits and receives data through a telephone line. You can use full duplex operation since telephone communication is bidirectional. Your software must accommodate software handshaking.

You must be in the on-line mode to enter the command mode. When in the command mode, you can enter any command from the command set. After the modem receives a command, it generates response code that indicates the result of the command.

Operation

This section of the chapter will show you how to:

- turn the modem on.
- move from the on-line to the command mode.
- enter a command.
- move from the command to the on-line mode.
- turn the modem off.

Turning the Modem On

The modem is normally off, to keep power requirements to a minimum. To turn on the modem, use the following procedure.

1. Boot the MS-DOS distribution disk from drive A.

2. Follow the instructions on the screen to exit SETUP.

3. Enter the following command line at the prompt:

```
MODE MODEM ON
```

4. Press the ENTER key. The modem is now on.

The modem uses the second communications channel. Operating systems, such as MS-DOS, refer to this channel as COM2. Refer to your operating system documentation and configure the operating system for operation with this channel.

Moving from On-Line to Command Mode

The time guard discussed in the following procedure is a time frame between normal on-line transmission and the command to exit to the command mode. You can modify the time guard through one of the modem's registers. Refer to the information on register 12, later in this chapter.

Use the following procedure to move to the command mode from the on-line mode.

1. Pause for the time guard to elapse (the default is one second).

2. Enter the following quickly (do not allow more than the time guard to elapse between each character):

   ```
   +++
   ```

3. Pause for the time guard to elapse. Do not prefix the three pluses with any characters. Also, do not press the ENTER key to end the line.

The modem is now in the command mode.

Entering a Command

Except for the A/ command and the +++ escape code, all command lines must start with the attention command (AT). You can place more than one command on a single command line. Command lines must not exceed 40 characters in length.

Use the following procedure to enter a command.

1. If you are not in the command mode, move to it from the on-line mode.

2. Unless you are entering the A/ command or the +++ escape code, type:

   ```
   AT
   ```

3. Enter commands on a single line up to 40 characters long. Do not separate commands with spaces or punctuation marks.

   Refer to each command for the exact format. You must not exceed 40 characters or the modem will return an error and will not execute any command.

4. Press the ENTER key

Moving from Command to On-Line Mode

The O (letter oh) command moves operation from the command to the on-line mode. In the example that follows, the O command is the only command issued in the command line. The following procedure assumes that the modem is in the command mode.

1. Enter the following command line:

   ```
   ATO
   ```

2. Press the ENTER key.

Turning the Modem Off

If you want to turn the modem off, use the following procedure.

1. Boot the distribution disk from drive A.

2. Follow the instructions on the screen to exit the SETUP program.
3. Enter the following command line at the prompt:

   ```
   MODE MODEM OFF
   ```

4. Press the ENTER key. The modem is now off.

Command Set

The Hayes modem AT command set allows you to control the modem through your own software. You can obtain additional modem control by programming the modem's registers, described later in this chapter.

To enter a command, you must be in the command mode. The command set consists entirely of printable ASCII characters that you send as data to the communications port. The command line contains one or more commands that usually start with the letters AT (attention). You can use up to 40 command characters in any one line. The ENTER key ends the line.

Refer to Table 5-1 for a summary of each command. Detailed explanations follow the table.

Table 5-1. Modem Command Set

CODE	COMMAND
A	Answer incoming call.
A/	Repeat last command.
AT	Attention.
B*n*	Select communication protocol.
C*n*	Carrier on/off.
D	Dial number.
E*n*	Echo on/off.
F*n*	Half/full duplex.
H*n*	Hang up.
I	Identification code.
L*n*	Speaker volume.
M*n*	Speaker control.
O	On-line.
Q	Result code display on/off.
S	Modify register.
V*n*	Verbose result codes on/off.
X*n*	Result level.
Y*n*	Determines if the modem will send or respond to long space disconnect signals. Y0 = no response (default). Y1 = responds
Z	Reset.

Answer Incoming Call

You can set the modem to automatically answer the phone, or you can use the A command to manually answer an incoming call. Register 0 determines whether the modem will answer automatically or not. Refer to "Modifying Registers" later in this chapter.

The following procedure describes how the modem responds to an incoming call.

1. When the telephone rings, the modem generates a RING result code. Refer to Result Codes later in this chapter.

2. If you set the modem to automatically answer the telephone, register 0 contains a value equivalent to the number of rings that must pass before the modem answers the phone.

3. If you set the modem to manually answer the phone, register 0 contains a 0. You must send the following command line to the modem *between* rings. The modem does not accept commands while the telephone is ringing.

4. First, enter the attention command (do not press the ENTER key):

   ```
   AT
   ```

5. Next, enter the answer command:

   ```
   A
   ```

   The entire command line is ATA. You can append additional commands to this command line.

6. Finally, press the ENTER key.

7. When the modem answers the telephone, it transmits a continuous carrier through the telephone line.

8. Next, the modem waits for an answering carrier. If the modem does not detect a carrier signal within the time specified by register 7 (30 seconds is the default), the modem hangs up. It generates a NO CARRIER result code and remains in the command state.

9. If the modem detects a carrier, it generates a CONNECT, CONNECT 1200, or CONNECT 2400 result code and goes on-line.

NOTE: It is possible that if you set the modem to echo the remote modem's commands, the remote modem's escape code can put your own modem into the command state. Modifying the escape code value can prevent this. Refer to "Modify Registers" later in this chapter.

Changing Between Voice and Data

You can change operation between voice and data communication without breaking the connection to another system. To change from voice to data, use the following procedure.

1. With the modem in the command state, establish a connection either by manually dialing the telephone or using the modem as a dialer (refer to "Dial Number" later in this chapter).

2. When you hear the carrier of the other line, enter the answer command (A).

3. Hang up the telephone.

Changing Between Data and Voice

You can change between computer data communications and voice without breaking the connection to the other system. Use the following procedure.

1. With the modem on-line to the other system, enter the command mode.
2. Pick up the telephone.
3. Make sure the other operator has picked up the phone at the other computer.
4. Enter the hang up command (described later in this chapter). The modem will release the line and since the phone is off the hook, the telephone company does not break the connection.

Repeat Last Command

When you send a command line to the modem, it places all of the line, except the AT (attention) command, in a 40-character command buffer. The repeat last command does not transfer anything to the command buffer. Instead, it causes the modem to execute the command line currently in the buffer, just as if you are sending the previous command line once again to the modem.

The repeat last command is a "stand-alone" command. That is, it does not require that you send any other characters with it. Use the following procedure to execute this command:

1. Enter the following line:

   A/

2. Do not press the ENTER key. The modem will execute the command stored in the command buffer as soon as the modem receives the slash (/) character.

You can use the A/ command to redial a telephone number or repeat any other command line you send to the modem.

Attention

The attention command clears the command buffer and enters all following commands into it (up to 40 characters). Use the following procedure.

1. With the modem in the command mode, enter the following characters (do not press the ENTER key):

   AT

2. Now enter any other command from the command set. You can enter up to 40 characters, not counting the AT, into the command line.
3. If you make a mistake entering commands, use the BACK SPACE key to erase your mistake.

Refer to the description of each command for the specific parameters to use.

Select Communication Protocol

The B*n* command establishes communication protocol. Most of the telephone companies in the United States use the Bell 103/212A protocol for telephone communication. Other countries may use another protocol. For example, many countries in Europe use the CCITT V.22 modem specifications. The 1200 baud modem supports the Bell 103/212A protocol only.

Use the following procedure to select communications protocol. The Bell 103/212A protocol is the default protocol.

1. Enter the command mode.
2. If you have not done so, enter the attention command:

   AT

3. Enter the select protocol command:

   B

4. Enter the parameter:

   0 for CCITT V.22 protocol
   1 for Bell 103/212A protocol

5. Press the ENTER key.

The modems always uses the Bell protocol at the slower baud rates (110 and 300 baud). Even if you modify the protocol, the modem will ignore it for the duration of 110 or 300 baud communication.

The modem's sending carrier will be whatever you specify to the communications device. You change the baud rate by modifying the registers of the communications device. For more information on programming serial communications devices, refer to Chapter 16.

This modem handles baud rates between 0 and 300 baud, 1200 baud, and 2400 baud. The modem uses the following, and 2400 baud procedure with Bell 103/212A protocol when it originates the call:

1. The modem connects to the telephone line.
2. It changes its mode to on-line.
3. The modem waits for a carrier from the remote modem. The value in register 7 determines the length of the wait time.
4. If the answering modem is using Bell 103/212A protocol, it always sends a 300 baud carrier, whether you set its baud rate to 300, 1200, or 2400 baud.
5. After detecting the answering modem's carrier, the modem will do one of two things, depending upon the baud rate set.

   a. If you set the modem's baud rate to 300 baud, it will transmit a 300 baud carrier and establish communication at 300 baud. (The modem generates a CONNECT result code.)

   b. If you set the modem's baud rate to 1200 or 2400 baud, it will transmit a 1200 or 2400 baud carrier. If the answering modem has the capability, it will switch to 1200 or 2400 baud. The modem establishes communication at 1200 or 2400 baud. (The modem generates a CONNECT, CONNECT 1200, or CONNECT 2400 result code.)

      If the answering modem cannot switch to 1200, or 2400 baud, the modem generates a NO CARRIER result code.

The modem uses the following procedure with Bell 103/212A protocol when it answers a call and its baud rate is set for 300 baud:

1. The modem detects the ring signal.
2. The modem either answers the call automatically or you issue an answer incoming call command.
3. The modem sends a 300 baud carrier.
4. If the originating modem responds with a 1200, or 2400 baud carrier, the modem establishes communication at 1200, or 2400 baud. (It generates a CONNECT, CONNECT 1200, or CONNECT 2400 result code and goes on-line).

   If the originating modem responds with a 300 baud carrier, the modem establishes communication at 300. Your software must adjust the serial device to 300 baud. (The modem generates a CONNECT result code. Your software should detect the difference between the codes. Refer to "Result Codes" later in this chapter.)

NOTE: The modem can communicate at speeds less than 300 baud. If the modem establishes the connection at a speed below 300 baud (for example, 110 baud), the CONNECT result code cannot specify the exact baud. Your software must identify the lower speed.

Carrier On/Off

The modem automatically turns the data carrier on or off. When the modem makes a call, answers a call, or connects directly to another modem, the modem turns the carrier on. Otherwise, it is off. If you want software to control the carrier, use the following procedure:

1. Place the modem in the command mode.
2. Enter the following attention command:

   `AT`

3. Enter the carrier command:

   `C`

4. Enter the parameter:

   `1` to turn the carrier on

   or

   `0` to turn the carrier off

5. Press the ENTER key. The full command line is ATC1 (turn carrier on) or ATC0 (turn carrier off).

Dial Number

The dial number command includes the telephone number and one or more modifying parameters. The number used in the example is for Zenith Data Systems Software Consultation. If you live within the 616 area code, do not use the area code when you enter the examples.

You can also use this command to initiate telephone calls for voice communication. After issuing the dialing command, pick up the telephone connected to the modem and send the hang up command to the modem.

You can use one or more dial command modifying parameters in a single command line. The modem's registers control timing parameters for some of the modifiers. Refer to "Modify Registers" later in this chapter. Table 5-2 describes the modifying parameters you can use with the dial number command.

Table 5-2. Dial Number Modifying Parameters

PARAMETER	DESCRIPTION
T	Touch-tone operation (pulse dialing is the default).
P	Pulse operation.
,	Pause.
/	Wait for 0.125 second
W	Wait for dial tone.
@	Wait for silence.
!	Hang up, wait for .5 second, reconnect.
R	Originate call in answer mode.
;	Return to command state after dialing.

Touch-tone operation — Most telephone system accommodate pulse dialing, even though they have touch tone capabilities. However, pulse dialing is slower than touch tone and therefore, if your telephone system can handle it, you should use touch tone dialing. Add the T modifier after the D (dial number) command. Do not use the T and P modifiers together.

Pulse operation — If your telephone system does not have touch tone operation, you must use pulse dialing. This is the default dialing mode. However, if you tried the T modifier and it failed, you must use the P modifier to change operation back to pulse dialing. Add the P modifier after the D (dial number) command. Do not use the T and P modifiers together.

Pause — The pause modifier introduces a two-second pause into the operation. If you place the comma (,) between two numbers, the modem will pause for two seconds before continuing. You can place two or more commas next to each other for longer pauses.

Wait for 0.125 second — The wait for 0.125 second modifier (/) causes the modem to pause 0.125 second before processing the rest of the command line.

Wait for dial tone — The wait for dial tone modifier (W) causes the modem to wait for a dial tone before it continues. If the modem does not detect a dial tone within 30 seconds, it generates the NO DIAL TONE result code.

Wait for silence — Some phone systems do not produce a dial tone. The ampersand (@) modifier causes the modem to do the following:

1. Detect some noise, such as a ringing signal. The noise must last a minimum of 210 milliseconds.
2. Wait for five seconds of silence.
3. Continue operation.

If the modem does not detect a five-second period of silence in 30 seconds, it generates a NO ANSWER result code.

Hang up, wait for .5 second, reconnect — Some telephone companies offer call transfer. This is where the telephone connection breaks for a minimum amount of time (usually under 1/2 second) and then makes the connection again. Hayes identifies this feature as hookflash. To use this feature, place an exclamation point (!) where you want the hookflash to take place.

Originate call in answer mode — Some communication systems are originate-only. That is, they do not have an answer mode. You can place the R modifier anywhere in a dialing command line. It instructs the modem to issue the answer mode carrier frequency once it establishes a connection.

Return to command state after dialing — This option allows the modem to return to the command mode after dialing the number. It does not break the connection. You must place the semicolon (;) modifier at the end of the command line.

You can use the ; modifier when you call a service that communicates through touch-tone dialing, such as an electronic banking service. You can repeatedly use the dial command and this modifier to communicate with the service. Remember, to keep the modem in the command state, end all subsequent dial commands with the semicolon modifier.

Procedure and Examples

Use the following procedure to dial a number. Examples that use the modifiers follow the procedure.

1. Turn the modem on and enter the command mode.

2. Enter the following attention command:

   ```
   AT
   ```

3. Enter the dial number command:

   ```
   D
   ```

4. Enter the telephone number you want to dial:

   ```
   16169823503
   ```

5. Press the ENTER key. The number includes:

   1 the long distance access code (this can vary from area to area).

   616 the area code for southwestern Michigan.

   982 the local calling exchange for Zenith Data Systems in St. Joseph, Michigan.

   3956 the number for the HUG (Heath User's Group) bulletin board.

The following examples use this same telephone number with one or more modifiers.

Example: `ATD16169823956`

This basic dial number instruction:

1. gets attention of the modem (AT).
2. enters the dial number command (D).
3. dials the telephone number (1-616-982-3956) using pulse or touch-tones, depending upon the mode used for the last number. Pulse dialing is the default mode.

Example: `ATDT1,,6169823956`

This dial number instruction:

1. gets attention of the modem (AT).
2. enters the dial number command (D).
3. tells the modem to use touch-tones for all following numbers (T).
4. dials the first number (1).
5. pauses twice (,,).
6. dials the rest of the telephone number (6169823956).

Example: `ATD9W1W616/982,3956;`

This dial number instruction:

1. gets attention of the modem (AT).
2. enters the dial number command (D).
3. dials an access code (9). The modem dials all numbers in the default mode, depending upon the preceding dial number commands.
4. waits for a dial tone (W).
5. dials a long distance access code (1).
6. waits for another dial tone (W).
7. dials the area code (616).
8. pauses for 0.125 second (/).
9. dials the local calling exchange (982).
10. pauses (,).
11. dials the rest of the telephone number (3956).
12. remains in the command mode, waiting for the next command.

Example: `ATDP9W1WT6169823956`

This dial number instruction:

1. gets attention of the modem (AT).
2. enters the dial number command (D).
3. tells the modem to use pulse dialing (P).
4. dials an access code (9).
5. waits for a dial tone (W).
6. dials a long distance access code (1).
7. waits again for a dial tone (W).
8. tells the modem to now use touch-tone dialing (T).
9. dials the rest of the telephone number (6169823956).

Example: `ATDP9@112WT6169823956`

This dial number instruction:

1. gets attention of the modem (AT).
2. enters the dial number command (D).
3. tells the modem to use pulse dialing (P).
4. dials an access code (9).
5. waits for five seconds of silence — the equivalent of a dial tone for some systems (@).
6. dials a long distance access code (112).
7. waits again for a dial tone (W).
8. tells the modem to now use touch-tone dialing (T).
9. dials the rest of the telephone number (6169823956).

Example: `ATDP9!11!W9W12WT6169823956`

This dial number instruction:

1. gets attention of the modem (AT).
2. enters the dial number command (D).
3. tells the modem to use pulse dialing (P).
4. dials an access code (9).
5. hangs up for 0.5 second and then reconnects (!).
6. dials a transfer code (11).
7. hangs up again for 0.5 second and then reconnects (!).
8. waits for a dial tone (W).
9. dials another access code (9).
10. waits for another dial tone (W).
11. dials a long distance access code (12).
12. waits again for a dial tone (W).
13. tells the modem to now use touch-tone dialing (T).
14. dials the rest of the telephone number (6169823956).

Example: `ATD9823956R`

This dial number instruction:

1. gets attention of the modem (AT).
2. enters the dial number command (D).
3. tells the modem to use answer number procedure (R). Note that this code is at the end of the number. You can place it anywhere after the dial number command.
4. dials the local telephone number (6169823956) using either pulse or touch-tone dialing, depending upon the default dialing mode.

If the number is busy, you can use the repeat last command (A/) instruction to redial the telephone number.

Echo On/Off

The modem normally echoes characters to the screen since most telephone communications works under full duplex operation (simultaneous bidirectional communication). If you have to use the modem for half-duplex communication, you should turn echo off. Use the following procedure to turn echo on or off.

1. Turn on the modem and enter the command mode.

2. Enter the attention command:

   ```
   AT
   ```

3. Enter the echo command:

   ```
   E
   ```

4. Enter the off command:

   ```
   0
   ```

   or the on command:

   ```
   1
   ```

5. Press the ENTER key. The full command line is ATE0 (turn echo off) or ATE1 (turn echo on).

Half/Full Duplex

The modem normally operates in the full duplex mode since most telephone communication is bidirectional. If you need half duplex operation, turn echoing off and use the following procedure to toggle the duplex mode.

1. Turn on the modem and enter the command mode.

2. Enter the attention command:

   ```
   AT
   ```

3. Enter the duplex command:

   ```
   F
   ```

4. Enter the half-duplex command:

   ```
   0
   ```

   or the full-duplex command:

   ```
   1
   ```

5. Press the ENTER key. The full command line is ATF0 (use half-duplex operation) or ATF1 (use full-duplex operation).

Hang Up

You can command the modem to hang up (break the connection). The Hayes command set also uses the H command for special applications. To break the connection use the following procedure.

1. Enter the command mode.
2. Enter the attention command:

   ```
   AT
   ```

3. Enter the hang up command:

   ```
   H0
   ```

4. Press the ENTER key. The full command line is ATH0 (hang up).

The H1 command operates both the line and auxiliary relays. The H2 command operates the line relay only.

Speaker Volume

Depending upon the modem board, the modem will use either its own speaker or the computer's speaker. The speaker volume command (L) controls the volume. The range is 0 (off) to 7 (maximum) with 3 being the default. Use the following procedure to set the speaker volume.

1. Turn on the modem and enter the command mode.
2. Enter the attention command:

   ```
   AT
   ```

3. Enter the speaker volume command:

   ```
   L
   ```

4. Enter the volume level:

   (any number between 0 and 7)

5. Press the ENTER key. The full command line is ATL0 (speaker off), ATL1 (speaker minimum), ATL2,... ATL7 (speaker maximum). If you use 0, the speaker is off. The number 7 is full volume. The number 3 is the default.

Speaker Control

You can control when the modem uses the speaker. It can be on all the time, off all the time, or on only as long as connected to the telephone line and the modem detects no carrier. Use the following procedure to control the speaker.

1. Turn on the modem and enter the command mode.
2. Enter the attention command:

   ```
   AT
   ```

3. Enter the speaker control command:

   ```
   M
   ```

4. Enter the speaker off command:

   ```
   0
   ```

   or the speaker on command:

   ```
   2
   ```

   or the speaker on/off command:

   ```
   1
   ```

5. Press the ENTER key. The full command line is ATM0 (speaker off), ATM1 (speaker on/off), or ATM2 (speaker on). The speaker on/off command causes the modem to turn the speaker on when it places a call. Then, when the modem detects a carrier, it turns the speaker off.

On-Line

The O (letter oh) command moves operation from the command mode to the on-line mode. Use the following procedure to go on-line. The following procedure assumes you are already in the command mode.

1. Enter the attention command:

```
AT
```

1. Enter the on-line command:

```
O
```

2. Press the ENTER key. The full command line is ATO (go on-line).

The O command has two other functions, but these are reserved for test the modem's performance.

Result Code Display On/Off

The modem generates result codes for each command it receives. You may not want to receive or monitor the result codes at all times. Therefore, this command allows you to turn the display of result codes on or off. Use the following procedure to turn the result code display on or off.

1. Turn on the modem and enter the command mode.

2. Enter the attention command:

```
AT
```

3. Enter the result code display on/off command:

```
Q
```

4. Enter the result code display on command:

```
0
```

or the result code display off command:

```
1
```

5. Press the ENTER key. The full command line is ATQ0 (result code display on), ATQ1 (result code display off).

Modify Register

The modem contains 17 eight-bit registers for additional operating parameters. The modem assigns each register default values when you first turn it on. You can then use the modify register command to read or change the value of the registers. Table 5-3 describes the registers.

Table 5-3. Modem Registers

REGISTER	DESCRIPTION
0	Ring to answer on.
1	Ring count.
2	Escape code character.
3	Carriage return character.
4	Line feed character.
5	Back space character.
6	Wait before dialing.
7	Wait for carrier after dialing.
8	Comma pause time.
9	Carrier detect response time.
10	Delay between lost character and disconnect.
11	Touch-tone dialing speed.
12	Escape code guard time.
13	Parity, data bit, and overflow flag.
14	Echo, result code, pulse dialing, speaker.
15	Answer/originate, duplex, baud, carrier.
16	Self-test.

Ring to answer on register — Register 0 sets the number of rings that the modem is to detect before it answers the telephone on auto-answer. The default value is 0. The range is from 0 to 255. If the value is 0, the modem disables auto-answer and generates the RING result code.

Ring count register — Register 1 tracks both the number of times the phone rings and the quality of the connection. The default value of the register is 0. The range is from 0 to 255.

While the modem is in the command mode, this register keeps count of incoming ring signals. If no ring occurs during an 8-second interval, the register re-

verts to 0. Your software must read this register on a regular basis to monitor the number of rings on incoming calls.

Once the modem makes a connection, this register monitors the quality of the connection for 1200 baud communication. The modem does not monitor the lower baud rates and keeps the register at 0. At 1200 baud, values of 45 and above indicate good signal quality. Values of less than 35 indicate poor signal quality. Poor signal quality can introduce errors into the data being received or transmitted.

Even though you can place a value into this register, you should use it only for monitoring the number of rings and the signal quality.

Escape code character register — Register 2 contains the escape code character. The default value for this character is 43 (2BH), the plus sign. You can change this value to any other ASCII character in the range of 0 to 127. If you do, the escape from the on-line mode sequence described earlier will change.

You may want to changing the value when the mode answers a call. This is to prevent the remote modem's escape codes from placing the answering modem into the command mode. This can happen when you echo remote modem commands.

If you use an ASCII value greater than 127, you disable the escape sequence. This may be desirable for modems on bulletins boards. Then, only the modem at the other end of the line can break the connection. There are some disadvantages. For example, when you disable the escape sequence, you cannot return to the command mode. Furthermore, the modem will not recognize commands, including the hang-up command.

Carriage return character register — Register 3 contains the ASCII carriage return (CR) character. The default value is 13 (0DH). The character serves as both the command line terminator and result code terminator. The range is 0 to 127.

Line feed character register — Register 4 contains the ASCII line feed (LF) character. The default value is 10 (0AH). The range is 0 to 127. If you do not want to generate any character for a line feed, change the value to 0.

Back space character register — Register 5 contains the ASCII back space (BS) character. The default value is 8 (08H). The range is 0 to 32 and 127. You cannot use any of the printable ASCII characters for the back space character. The character serves not only as the back space character, but also for the character that moves the cursor back one position. Because some on-line equipment requires extra time to perform a back space, allow a minimum of three normal character frames each time you use the back space.

Wait before dialing register — Register 6 contains a value that determines how many seconds the modem will wait before dialing a number. The time delay allows the telephone system time to apply a dial tone to the line. Use blind dialing only when you disable the modem result code that detects the dial tone. Refer to "Result Level" later in this chapter. The range is 0 to 255 seconds. The default value is 2 seconds.

Wait for carrier after dialing register — Register 7 contains a value that determines how many seconds the modem will wait for the answering modem's carrier after dialing. If the modem does not detect a carrier within this time, it hangs up and generates the NO CARRIER result code. If the modem detects the carrier, it goes on-line. The range is 0 to 255 seconds. The default is 30 seconds.

This value also controls how many seconds the modem waits for a dial tone when you use the W modifier in the dial number command. Refer to "Dial Number" earlier in this chapter.

Comma pause time register — Register 8 contains a value that determines how many seconds the modem will pause when it encounters a comma (,) in the dial number command. Refer to "Dial Number" earlier in this chapter. The range is 0 to 255 seconds. The default is 2 seconds.

Carrier detect response time register — Register 9 contains a value that determines how long a carrier has to be present before the modem will recognize it. A higher value prevents the modem from mistaking a ring or busy signal for a carrier. The range is 1 to 255 tenths of a second. The default is .6 second (6).

Delay between lost character and disconnect register — Register 10 contains a value that determines how long the modem will wait after the remote modem's carrier has disappeared. After the time elapses, the modem will disconnect from the telephone line. The range is 1 to 255 tenths of a second. The default is .7 second (7).

The higher the value, the less change static or a poor connection will break the connection. Set this value to 255 if you want the modem to ignore actual carrier status and function as if a carrier were always present. Also, if you have call waiting, a high value will prevent the call waiting alert signal from interrupting your connection. Note however, you will lose some data during periods of heavy static or when the call waiting alert signals occur.

There is some interaction between the value in register 9 and 10. First, the modem does not recognize a carrier until it is on for the amount of time in Register 9. If the value in 10 is less than that of register 9, even momentarily losing the carrier will disconnect the signal.

Also, it takes the modem at least .05 seconds before it recognizes that the carrier is gone. The time set in this register starts after the .05-second delay. Therefore, the maximum length of time a carrier can be off is: the value in register 10 minus the value in register 9 plus .05 second.

Touch-tone dialing speed register — Register 11 contains a value that determines the duration and spacing between the tones for touch-tone dialing. The range is 20 to 255 milliseconds. The default is 70 milliseconds.

Reliable dialing is lost with values less than 50 milliseconds. This is equivalent to 10 digits per second. The maximum value is 255 milliseconds or 1.9 digits per second.

Escape code guard time register — Register 12 contains a value that determines the escape code time guard. The range is 0 to 255 fiftieths of a second (0 - 5.1 seconds). The default is 50 (1 second).

The escape code guard time is the time delay required before and after the escape code characters. The guard time also determines how quickly you must enter the escape code characters. The interval between entry of each of the three characters must be less than the guard time. If you do not meet these conditions, the modem will not recognize the escape code.

If you set the guard time to 0, timing is not a factor in detecting the escape code.

Parity, data bit, and overflow flag register — The content of register 13 determines the status of a number of functions. Refer to Table 5-4 for the function of each bit in the register.

NOTE: This register reflects the settings of the modem's communication device and does not control word length or parity. Therefore, writing to this register can produce unpredictable results.

Table 5-4. Contents of Modem Register 13

BIT	FUNCTION
0 - 1	Not used.
2	If clear (0), the modem does not use parity. If set (1), the modem uses parity.
3	If clear (0), the modem uses odd parity. If set (1), the modem uses even parity.
4	If clear (0), the modem uses 7-bit data words. If set, the modem uses 8-bit data words.
5	Not used.
6	If set (1), the modem generates the ERROR code. This is the buffer overflow flag, which has room for only 40 characters.
7	If clear (0), the parity bit will be a space (0). If set (1), the parity bit will be a mark (1). Note bit 4 must be set to 7-bit words.

Echo, result code, pulse dialing, and speaker register — The content of register 14 determines the status of a number of functions. Refer to Table 5-5 for the function of each bit in the register.

NOTE: Some bits of this register reflect settings from various commands. Therefore, writing to this register can produce unpredictable results.

Table 5-5. Contents of Modem Register 14

BIT	FUNCTION
0	Not used.
1	If clear (0), the modem has the echo feature off. If set (1), the modem has the echo feature on. Refer to "Echo "On/Off" earlier in this chapter.

Table 5-5 (Continued). Contents of Modem Register 14

BIT	FUNCTION
2	If clear (0), the display result codes feature is on. If set (1), the display result codes feature is off. Refer to "Display Result Codes On/Off" earlier in this chapter.
3	If clear (0), verbose result codes is off. If set (1), verbose result codes is on. Refer to "Verbose Result Codes On/Off" later in this chapter.
4	Clear (0). If you set this bit (1), the modem will ignore all commands.
5	If clear (0), the pulse dialing make/break ratio is 67:33. If set (1), the pulse ratio is 61:39.
6	If set (1), the modem turns the speaker on until it detects a carrier. Refer to "Speaker Control" earlier in this chapter.
7	If set (1), the speaker is always on. Refer to "Speaker Control" earlier in this chapter.

Answer/originate, duplex, baud, and carrier register — The content of register 15 determines the status of a number of functions. Refer to Table 5-6 for the function of each bit in the register.

NOTE: Some bits of this register reflect settings from various commands. Therefore, writing to this register can produce unpredictable results.

Table 5-6. Contents of Modem Register 15

BIT	FUNCTION
0 - 1	These bits are set the same as bits 4 and 5. Refer to Table 57 for the baud rate.
2	If clear (0), the modem is set to answer the call. If set (1), the modem is set to originate the call.
3	If clear (0), the modem is in half-duplex mode. If set (1), the modem is in full-duplex mode.
4 - 5	These bits determine the baud rate. Refer to Table 5-7.
6	If clear (0), the carrier is off. If set (1), the carrier in on.
7	Not used.

Table 5-7. Baud Rate

BIT 5	BIT 4	BAUD RATE
0	0	Not used.
0	1	110 bits per second.
1	0	300 bits per second.
1	1	1200 bits per second.

Self-test register — Register 16 enables the modem's self-test capabilities. The possible modes are 0, 1, 2, or 4. The default is 0. Refer to the service manual for more information on testing the modem. Table 5-8 defines the four modes of this register.

Table 5-8. Self-Test Register Modes

MODE	DESCRIPTION
0	Non-test mode. The modem operates normally.
1	On-line analog loop-back test mode.
2	Touch-tone test mode.
4	On-line digital loop-back test mode.

Operation and Examples

You can use the modify register command to read or change the values in the registers. Use the following procedure to read a register.

1. Turn on the modem and enter the command mode.
2. Enter the attention command:

   ```
   AT
   ```

3. Enter the modify register command:

   ```
   S
   ```

4. Enter the register number:

   ```
   4
   ```

This is an example. You can use any number between 0 and 16.

5. Enter the query modifier:

   ```
   ?
   ```

6. Press the ENTER key. The full command line is ATS4? The modem returns the value in the register as a decimal number:

   ```
   010
   ```

You can read more than one register with a single command line. For example, if you want to read the value of register 0 and register 7, you would enter the following command line:

```
ATS0?S7?
```

As you can see, this follows the same procedure for entering more than one command. The modem responds to the example by generating the following responses and result code:

```
001
030
OK
```

Refer to "Result Codes" later in this chapter.

To change a register, use the following procedure.

1. Turn on the modem and enter the command mode.

2. Enter the attention command:

   ```
   AT
   ```

3. Enter the modify register command:

   ```
   S
   ```

4. Enter the register number:

   ```
   2
   ```

This is an example. You can use any number between 0 and 16. Refer to the register descriptions for limitations.

5. Enter the equal modifier:

   ```
   =
   ```

6. Enter the new value as a three-digit decimal number:

   ```
   092
   ```

7. Press the ENTER key. The full command line in this example is ATS2=092. The modem changes the value in the register. This example changes the escape character to a backslash (/).

Verbose Result Codes On/Off

The modem can generate the result codes as numbers or words. The default is words. Use the following procedure to change the verbose result codes on/off command.

1. Turn on the modem and enter the command mode.

2. Enter the attention command:

   ```
   AT
   ```

3. Enter the verbose result codes on/off command:

   ```
   V
   ```

4. Enter the verbose off command:

   ```
   0
   ```

   or the verbose on command:

   ```
   1
   ```

5. Press the ENTER key. The full command line in this example is ATV0 (turn verbose off) or ATV1 (turn verbose on). Refer to "Result Codes" later in the chapter for more information.

Result Level

The result level command enables various features and their result codes. There are five levels, designated by the numbers 0 through 4. The default level is 0. Each level includes the features of the lower levels. In other words, level 2 includes all the features of level 1 and level 0. Table 5-9 describes the features enabled and the result codes that the modem can generate at each level.

Table 5-9. Result Levels

LEVEL	FEATURES ENABLED
0	Dial modifiers and 0-300, 1200, and 2400 baud transmission speed. The modem can generate result codes 0 - 4.
1	Level 0 plus transmission speed detection. The modem can generate result codes 0 - 5*.
2	Levels 0 and 1 plus dial tone detection. The modem can generate result codes 0 - 6*.
3	Levels 0 and 1 plus busy signal detection. Dial tone detection is not enabled. The modem can generate result codes 0 - 5 and code 7*.
4	Levels 0, 1, 2, and 3 are all enabled. The modem can generate result codes 0 - 7*.

*Also result code 10 on 2400 baud modems.

Connection Sequences

The result level determines the connection sequence used by the modem. The sequences show the registers that control timing in parenthesis.

Level 0 — When the modem originates the call, it:

1. connects to the telephone line.
2. waits 2 seconds (register 6).
3. dials the number.
4. waits 30 seconds (register 7) for .6 second (register 9) of answering carrier.

If the modem does not detect a carrier, it disconnects from the telephone line and generates the NO CARRIER result code.

Whether the modem originates or answers a call, when it detects a carrier, it generates the CONNECT result code.

Level 1 — When the modem originates the call, it:

1. connects to the telephone line.
2. waits two seconds (register 6).
3. dials the number.
4. waits 30 seconds (register 7) for .6 second (register 9) of an answering carrier.

If the modem does not detect a carrier, it disconnects from the telephone line and generates the NO CARRIER result code. Busy signals are not detected.

Whether the modem originates or answers the call,

- if it detects a 300 baud carrier, it establishes communication at 300 baud and generates the CONNECT result code.
- if it detects a 1200 or 2400 baud carrier, it establishes communication at 1200 or 2400 baud and generates the CONNECT 1200 or 2400 result code.

Level 2 — When the modem originates the call, it:

1. connects to the telephone line.
2. waits up to five seconds for a one-second, continuous dial tone.

If the modem does not detect the dial tone, it generates the NO DIAL TONE result code and disconnects from the telephone line.

If the modem detects the dial tone, it:

1. dials the number.
2. waits 30 seconds (register 7) for .6 second (register 9) of an answering carrier.

If the modem does not detect a carrier, it disconnects from the telephone line and generates the NO CARRIER result code. Busy signals are not detected.

Whether the modem originates or answers the call,

- if it detects a 300 baud carrier, it establishes communication at 300 baud and generates the CONNECT result code.
- if it detects a 1200 or 2400 baud carrier, it establishes communication at 1200 or 2400 baud and generates the CONNECT 1200 or 2400 result code.

Level 3 — When the modem originates the call, it:

1. connects to the telephone line.
2. waits two seconds (register 6).
3. dials the number. The modem does not wait for a dial tone.

4. looks for a busy signal. If the modem detects the busy signal, it disconnects the telephone line and generates the BUSY result code.

If the modem does not detect a busy signal, it:

1. waits 30 seconds (register 7) for .6 second (register 9) of an answering carrier.

If the modem does not detect a carrier, it disconnects from the telephone line and generates the NO CARRIER result code.

Whether the modem originates or answers the call,

- if it detects a 300 baud carrier, it establishes communication at 300 baud and generates the CONNECT result code.
- if it detects a 1200 or 2400 baud carrier, it establishes communication at 1200 or 2400 baud and generates the CONNECT 1200 or 2400 result code.

Level 4 — When the modem originates the call, it:

1. connects to the telephone line.
2. waits up to five seconds for a one-second, continuous dial tone.

If the modem does not detect the dial tone, it generates the NO DIAL TONE result code and disconnects from the telephone line.

If the modem detects the dial tone, it:

1. dials the number.
2. looks for a busy signal. If the modem detects the busy signal, it generates the BUSY result code and disconnects from the telephone line.

If the modem does not detect a busy signal, it:

1. waits 30 seconds (register 7) for .6 second (register 9) of an answering carrier.

If the modem does not detect a carrier, it disconnects from the telephone line and generates the NO CARRIER result code.

Whether the modem originates or answers the call,

- if it detects a 300 baud carrier, it establishes communication at 300 baud and generates the CONNECT result code.
- if it detects a 1200 or 2400 baud carrier, it establishes communication at 1200 or 2400 baud and generates the CONNECT 1200 or 2400 result code.

Long Disconnect

The modem does not normally respond to long disconnect signals, nor does it send any. The long disconnect command allows the modem to send a four-second break signal before it disconnects from the telephone line in response to the hang up command. Likewise, it treats a 1.6 break signal as a disconnect signal. To use the long disconnect command:

1. Turn on the modem and enter the command mode.
2. Enter the attention command:

   ```
   AT
   ```

3. Enter the long disconnect command:

   ```
   Y
   ```

4. Enter the long disconnect recognition on command:

   ```
   1
   ```

   or the long disconnect recognition off command:

   ```
   0
   ```

5. Press the ENTER key. The full command line in this example is ATY1 (turn long disconnect recognition on) or ATY0 (turn long disconnect recognition off).

Reset

There are three ways to reset the modem to its default settings: turn the computer off and then on again, perform a software reset, or perform a hardware reset. The Portable computer performs a hardware reset every time you turn on the computer. The hardware reset can also be performed by writing to one of the serial device's registers. Refer to Chapter 16 for more information.

The reset command performs a software reset. It resets the modem to its default settings, resets the registers to their default values, and breaks the telephone connection. The modem responds with the OK result code. The reset command also clears the command buffer. Therefore, any commands that follow the reset command in the current command line are ignored. Likewise, the repeat command will not work because the reset command clears the command buffer.

To use the reset command:

1. Enter the command mode.
2. Enter the attention command:

   ```
   AT
   ```

3. Enter the reset command:

   ```
   Z
   ```

4. Press the ENTER key. The full command line is ATZ.

Result Codes

After the modem receives a command, it generates one of the result codes listed in Table 5-10. Table 5-11 describes the enabled result codes based upon the result level command setting. For more information, refer to "Result Level" earlier in the chapter.

Table 5-10. Modem Command Response Codes

NUMBER	WORD	DEFINITION
0	OK	The modem executed the command properly.
1	CONNECT	The modem made connection to another modem operating at 0 to 300 baud. If no other level than level 0 is active, the modem also generates this message for connection at 1200 and 2400 baud.
2	RING	The modem detected the ring signal.
3	NO CARRIER	The modem was not able to make a connection or the connection was made and then subsequently lost.

Table 5-10 (Continued). Modem Command Response Codes

NUMBER	WORD	DEFINITION
4	ERROR	The modem detected an error. One of four can occur: the modem did not recognize the command; the command line exceeds the 40-character limit; the character format is invalid for 1200 baud; or the modem has detected an invalid checksum.
5	CONNECT 1200	The modem made connection to another modem at 1200 baud.
6	NO DIAL TONE	The modem does not detect a dial tone. It will not continue processing the command.
7	BUSY	The modem dialed the number and detected a busy signal.
8	NO ANSWER	The modem waited for silence after dialing the number but did not detect it. The modem enables this result code when it uses the @ dial number modifier.
10	CONNECT 2400	The modem connected to another modem at 2400 baud.

The verbose result code on/off and result code display on/off commands control the display of the result codes generated by the modem. The verbose result code on/off command controls the form the result codes take. The modem generates the number in the NUMBER column when the verbose result code is off. The modem generates the words in the WORD column when the verbose result code is on. Refer to "Verbose Result Codes On/Off" and "Result Code Display On/Off" earlier in this chapter.

Table 5-11. Result Codes Enabled

LEVEL	RESULT CODES ENABLED
0	0 - 4.
1	0 - 5*.
2	0 - 6*.
3	0 - 5 and 7*.
4	0 - 7*.

NOTE: The modem enables result code 8 when it uses the @ dial number modifier. The result level command does not affect this code.

*Also result code 10 on 2400 baud modems.

Chapter 6

The Monitor Program Debugger and Programming with Interrupts

This chapter provides an overview of the Monitor program Debugger and the input/output system. Each time you turn the computer on a number of system interrupts become available for use. The chapters that follow provide explanations for each interrupt.

The Monitor Program

The Monitor program, MFM-180, maintains a high level of software compatibility with desktop PC-compatible computers. This is largely done through the use of firmware-established interrupts for most control and input/output routines. In addition, the Monitor program handles initial self-tests performed each time you turn on the computer. Additional routines provided in the Monitor program are user-executed diagnostic tests, video commands, and disk boot routines. Refer to Chapter 4 for a discussion of these areas.

The Monitor program has six basic features. These features include: automatic power-up self-tests, user-executed tests, video commands, disk boot routines, machine language debugger, and basic input/output system. This portion of the manual will address the last two topics. Chapter 4 contains a discussion of the remaining features. Chapter 19 contains a discussion of the error messages generated by the self-tests and the user-executed tests.

Two methods are normally employed to gain access to the Monitor program. The first option requires that you bypass the autoboot procedure. To defeat autoboot, press the ESC key after you first turn the system on. The system will display the monitor prompt (->). The second method allows you access to the Monitor program after the computer has successfully booted. To reach the Monitor program after the operating system boots, press CTRL-ALT-INS. The Monitor program opening message, similar to the following, will appear on the display.

```
MFM-180 Monitor, version 2.0
Memory Size: 640K
Press "?" for help.
->
```

NOTE: The version number will vary from the one printed in this example.

A third, and final, method is available to access the Monitor program. This method will allow you to access the Monitor program, perform operations within the Monitor program, and return to the Operating System without the necessity of rebooting the computer. In order to access the monitor program in this manner, press the CTRL-ALT-ENTER keys. The monitor prompt will be displayed in the normal fashion. Perform any operations you may require while in the Monitor program. When you are ready to return to the operating system press the G key followed by the ENTER key. You may now proceed with normal operating system functions.

NOTE: Some programs do not follow MS-DOS conventions during execution. Using the above method to return to the Monitor program may leave certain system registers in unexpected or undefined states. This could result in strange or unexpected results when you attempt to return to the operating system. Always make sure that you save important material prior to implementing this command.

The Machine Language Debugger

The Monitor program contains a set of machine language debugging commands, listed in Table 6-1. Examples and descriptions follow the table.

The message, `^ Invalid Command`, is the Monitor program's syntax error message. The caret points to the exact spot where the first syntax error occurs. The system does not recognize additional syntax errors after it detects the first error. This error does not lock up the system. Therefore, the command can be reentered or another command entered in its place.

Display Memory

Syntax: D <address>[L<length> | <offset>]

Example:
```
D 1000:0 F ENTER
D DS:7000 L200 ENTER
```

Use this command to display the contents of the specified portion of memory. If you do not specify the length in the command line, the debugger will display 128 bytes of data, starting at the specified address. The resulting display contains three sections. The left-most column is the base address and offset of the first byte in the second part of the display. The second part uses the hexadecimal format to display 16 bytes of data. The right-most part of the display contains the ASCII characters represented by the data in the second part of the display. If any byte is outside the ASCII character range, the debugger will print a period in its place.

Table 6-1. Machine Language Debugger Commands

COMMAND	DESCRIPTION	SYNTAX
D	Display memory	D <address>[L<length>|,<offset>]
E	Examine memory	E <address>
F	Fill memory	F <start address>,<range>,<data list>
G	Go (Execute)	G [=<address>][,<breakpoint>]...
H	Hex math	H <number1>,<number2>
I	Input from port	I <port address>
M	Move memory block	M <start address>,<range>,<new address>
O	Output to port	O <port address>,<value>
R	Examine Registers	R [<register>]
S	Search memory	S <start address>,<range>,<data list>
T	Trace program	T [=<address>][,<value>]
U	Unassemble program	U [<address>][<length>|,<offset>]

If you enter the first example, as shown above, a display of 16 bytes of memory will result, starting at address 1000:0. If that portion of memory is empty (containing only nulls, or 0's), the resulting display will be:

```
1000:0000  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
```

Entering the second example will result in a display containing 200H bytes of memory. This display will start at the address offset located 7000H bytes from the base address of the data segment. The CPU DS segment register stores the base address location.

If the memory range is greater than the screen can display, the data will scroll. Use CTRL-S to stop and CTRL-Q to continue the screen action.

Examine Memory

Syntax: E <address>

Example: `E 1000:100` *ENTER*

Use the examine command to examine or change memory, one byte at a time. When this command executes, the display will show the byte at the specified memory address. The command will wait for an entry from the keyboard. Enter a hexadecimal value from 0 to FF to modify the current memory location. Press the space bar to examine the next byte of memory or the minus (hyphen) key to display the previous byte of memory. Press the ENTER key to complete this command and return to the Monitor program prompt.

CAUTION: You can examine and modify any addressable memory location, including those reserved for interrupts and interrupt vectors. Therefore, be careful that you do not accidentally modify routines that the Monitor program uses or you could lock up the computer. If this does occur, turn the computer off to reset it.

Fill Memory

Syntax: F <start address>,<range>,<data list>

Example:

```
F 1800:0,1FF,"TEST",20,54,45,53,54,20
                                    ENTER
```

Use the fill command to enter one or more bytes of data directly into memory. You can use ASCII characters, delineated by quotation marks, or hexadecimal format. Each byte and ASCII text word requires a comma for separation. Data in the list will be reused as often as necessary to fill the specified memory range. You must specify the starting address, range, and data before the command will function. In the example, 1FFH bytes, starting at memory address 1800:0, will fill with repeating strings of the data. The entire string is used, which includes the material in quotes plus the material expressed as hexadecimal values — the result of this example is the string, TEST TEST .

CAUTION: You can use the fill command to modify any addressable memory range, including those reserved for interrupts and interrupt vectors. Therefore, be careful that you do not accidentally modify routines that the Monitor program uses or you could lock up the computer. If this does occur, turn the computer off to reset it.

Go (Execute)

Syntax: G[=<address>][,<breakpoint>]...

Examples:

```
G 1000:0 ENTER
G 100 ENTER
G=1000 100 ENTER
```

You can use the go command to transfer control from the Monitor program to a machine language or user program. The transfer will occur at the address specified in the command line or in the instruction pointer and code segment of the CPU. Also, as a debugging aid, you may specify up to eight optional breakpoints (points in the program where you want the program to halt) in the command line.

In the first example, control will transfer to the address specified in the CPU registers and the debugger will set a breakpoint at memory address location 1000:0. In the second example, the monitor will transfer control to the address specified in the CPU registers and set a breakpoint at the memory address offset. This address location is 100H bytes from the location of the first byte of executable code. In the third example, the Monitor program sets the code segment register to 1000. Control then trans-

fers to this address and the Monitor program sets a breakpoint at memory offset 100H.

When the go routine encounters a breakpoint, the Monitor program will save the status and register information. This information will appear as a register dump (refer to the examine/modify registers command). The screen will display the status of the CPU's registers and flags as well as the current instruction executed just prior to the breakpoint.

You may set up to eight breakpoints and the debugger will halt when it encounters one of them, Once the debugger finds a breakpoint, the other breakpoints, stored as parameters to the command line, are forgotten.

NOTE: You must set breakpoints to addresses that contain valid instruction codes, otherwise the computer will lock up.

Hex Math

Syntax: H <number1>,<number2>

Example: `H 2a,28 ENTER`

Use the hex math command to compute the sum and difference of two hexadecimal numbers. The debugger always subtracts the second number from the first. In the example, the results will display as SUM: 0052 Diff: 0002.

Input From Port

Syntax: I <port address>

Example: `I 3F8 ENTER`

Use the input command to obtain a byte of data from a port address. In the example, a single byte from port 3F8H will display on the screen. You can address most programmable devices used in this computer through port addresses.

Move Memory Block

Syntax:

M <start address>,<range>,<new address>

Example: `M 0:1000,107F,0:2000 ENTER`

Use the move command to copy a specified block of memory to another specified location in memory. In the example, the Monitor program will copy a block of memory, 107FH bytes long, starting at memory address 0:1000 to a memory location that starts at 0:2000.

CAUTION: You can use the move command to move a block of memory from any addressable memory location to any other addressable memory location, including one reserved for interrupts and interrupt vectors. Therefore, be careful that you do not accidentally modify memory that contains routines used by the Monitor program or you could lock up the computer. If this does occur, turn the computer off to reset it.

Output To Port

Syntax: O <port address>,<data>

Example: `O 2F8,00 ENTER`

Use the output command to send a byte of data out a port address. In the example, the Monitor program will send the value of 0H out port address 2F8H. The computer can address most programmable devices through port addresses.

Examine/Modify Registers

Syntax: R [<register>]

Example:
```
R ENTER
R CS ENTER
R FL ENTER
```

Use the examine registers command to examine and alter the contents of the CPU's registers and flags. In the first example, the computer performs a register dump. The dump will display the current instruction, along with the contents of all registers and flags.

In the second example, the display will contain the contents of the CS register and the computer will wait for an entry from the keyboard. Enter a hexadecimal value to modify the contents of the current register, followed by the ENTER key. If the value you enter is not valid, the register's contents will not change. Press the ENTER key alone to leave the contents of the register unchanged.

In the third example, the display will contain the status flags. Table 6-2 contains the two-letter abbreviation that describes the status of each flag. The computer will wait for an entry from the keyboard. Enter the two-letter abbreviation to set the status of a flag, followed by a space and another two-letter abbreviation for another flag or by the ENTER key to terminate the command. If you make more than one entry for the same flag, the last value entered will be the one that remains. Press only the ENTER key to leave the flags unmodified.

Table 6-2. Processor Status Flag Codes

FLAG	ON	OFF
Overflow	OV(overflow)	NV(no overflow)
Direction	DN(down)	UP(up)
Interrupts	EI(enabled)	DI(disabled)
Sign	NG(negative)	PL(plus)
Zero	ZR(zero)	NZ(not zero)
Auxiliary Carry	AC(auxiliary carry)	NA(no auxiliary carry)
Parity	PE(even)	PO(odd)
Carry	CY(carry)	NC(no carry)

Search Memory

Syntax: S <start address>,<range>,<data list>

Example:

```
S 0:2000,1000,"TEST",20,"TEST" ENTER
```

Use the search command to search the specified area of memory for the occurrence of specific data strings or bytes of information. In the example, the search routine will examine 1000H bytes of memory, starting at memory location 0:2000, for the occurrence of TEST TEST. Notice that quotation marks enclose the ASCII text; otherwise, you must enter data in hexadecimal format. You must separate each item in the data list by a comma from the other items in the list. If you do not provide any data in the list, the Monitor program will return the ^ Invalid Command! error message.

Trace User Program

Syntax: T [=<address>][,<value>]

Example:

```
T=400:0,200 ENTER
T 5 ENTER
```

Use the trace command to single-step the execution of a program. Each time you execute the command, it executes the next instruction and performs a register dump. The dump will display the current instruction and the contents of the registers and flags. Enter T by itself to execute the instruction pointed to by the current code segment and IP register. Use the examine/modify registers command to change these values and to specify a new memory location for trace operation.

In the first example, the routine places the value following the equal sign (=) in the CS and IP registers. Trace will use this value as the starting address and will execute 200H times. Since this is a large number, the register dumps will fill and scroll the screen. Use CTRL-S to stop and CTRL-Q to start the scrolling action.

In the second example, trace will start at the current address stored in the CS and IP registers of the CPU. The command will repeat five times.

CAUTION: The trace command does not contain extensive error trapping routines to prevent accidental operation of the computer. It is possible that timing-dependent operations, such as reading or writing to a disk, could cause problems, and, in the case of disk input/output operations, damage the media itself. Therefore, you should not use trace to step through the Monitor program or interrupt service routines.

Unassemble

Syntax: U [<address>[<length>|,<offset>]

Example:

```
U ENTER
U 1000 ENTER
U 0:1000 ENTER
```

Use the unassemble command to disassemble a section of memory into assembly language mnemonic format. The resulting display resembles an assembly language source code listing, but contains no comments. The Monitor program will treat data areas that contain ASCII text as code. Therefore, it is possible to create meaningless assembly language code from raw text or other data.

In the first example, the routine will begin disassembly in the current code segment (pointed to by the CS register) at the address pointed to by the IP register. The routine will disassemble and display thirty-two

bytes (20H) of data. The number of lines displayed on the screen will vary depending upon the number of bytes generated by each instruction.

In the second example, disassembly starts at the memory location offset 1000H bytes from the value of the CS register. In the third example, disassembly starts at memory location 0:1000. In all three cases, the routine will disassemble and display 32 bytes of data.

NOTE: Although unassemble modifies the values in the CS and IP registers, the routine stores the original values in a stack. If you use the examine/modify registers command, the routine will restore the original values and subsequent use of the unassemble command will use these values.

Although not used in any example, you can use the length parameter to disassemble more or less than 32 bytes of memory. Since this may result in a display that scrolls the screen, use CTRL-S to stop and CTRL-Q to start the scrolling action.

Programming with Interrupts

The basic input/output system is the portion of the Monitor program that establishes the machine-level input/output routines. These routines provide control at the device level of the main input/output devices for the computer. Use software interrupts to call a routine and perform a function.

To use an interrupt, place a specific code in register AH of the CPU and execute an INT instruction. As a rule, the Monitor program routines will preserve the values in all CPU registers except for AX and the flags. Modifications of other registers may occur upon completion of the current task only if they are returning a value to the program that executed the function.

The firmware generates software interrupts when certain events occur within the computer. You can use these events in your programs by patching the events' interrupt vectors for your routines. If you do this and run your routines with other programs that use or modify the same interrupts or memory locations, you may experience unexpected results.

Hardware Interrupts

Although this part of the technical manual deals with the software interrupts, you need to understand the difference between interrupts generated by application software or the firmware of the computer and interrupt requests made by the hardware.

The 82C59 programmable interrupt controller manages the hardware interrupts. In the event that more than one request for an interrupt occurs at the same time, the controller will manage these requests on a priority basis.

The system assigns the highest priority to IRQ0 (interrupt request 0) and the lowest to IRQ7. By sending a bit mask to port 021H, the programmer can control whether a particular interrupt can be processed when requested. For more information on the 82C59, refer to Chapter 14.

Table 6-3 describes the usual hardware interrupt requests and the software interrupts they trigger. The normal order of precedence appears in the table. Jumpers and/or other hardware factors may change the source of the interrupt request in some computer installations (not recommended).

Table 6-3. Hardware Generated Interrupt Requests

HARDWARE INTERRUPT REQUEST	SOFTWARE INTERRUPT	USUAL SOURCE OF INTERRUPT
0	08H	Time-of-day timer
1	09H	Key pressed
2	0AH	Real-time clock (not used in this computer)
3	0BH	Communications (COM2)
4	0CH	Communications (COM1)
5	0DH	Hard Disk Drive
6	0EH	Floppy disk drive
7	0FH	Parallel printer (LPT1)

Using a Software Interrupt

The operating system initializes some of the interrupts discussed in this manual. Since MS-DOS is the standard operating system for these computers and helps maintain PC-compatibility, those interrupts considered universal (not specific to a particular version of MS-DOS) are included in the summary pre-

sented later in this chapter. The *Programmer's Utility Pack* for MS-DOS contains a discussion of those interrupts that are subject to change between versions of MS-DOS. Other operating systems and MS-DOS for other computers (not PC-compatible) may or may not initialize the same or similar interrupts.

NOTE: Any software interrupt that is not initialized will jump to a return (RET) function (no operation).

When you prepare to use or modify an interrupt routine, there are a couple of factors you should keep in mind.

- All parameters passed to and from interrupt routines must go through the CPU's registers. When a routine may perform more than one function or require additional parameters, the interrupt may use more than one register.

- Most of the interrupt routines will preserve the values in the CPU registers except when returning a value to a specific register. When you write or modify routines for these interrupts, you should plan on preserving, where possible, the values of the CPU registers.

A dedicated interrupt vector (address) points to each interrupt routine. To execute one of the interrupt routines, you must first load the specified parameters into the registers of the CPU. Then perform an INT *xx*H instruction where *xx*H is the interrupt number. For instance, the code example in Listing 6-1 executes the INT 11H interrupt and then tests for and returns the number of drives connected to the system as established by the configuration switches.

Since no parameters were required by INT 11H, none are loaded. Note however, that the results are returned in the AL register and then tested by the routine.

Modifying an Interrupt

To modify an interrupt, you must patch the interrupt vector with the address of your routine as outlined in the following procedure (MS-DOS environment). For more information, refer to the *Programmer's Utility Pack* for your version of MS-DOS.

1. Use the MS-DOS Function Request *Get Interrupt Vector* (35H) to read the current value of the interrupt vector. Save this value in your code segment.

2. Use the MS-DOS Function Request *Set Interrupt Vector* (25H) to set the address of your interrupt routine.

3. Do not execute an IRET instruction at the end of your routine. Instead, execute a JMP DWORD PTR to the address saved in step 1. This allows multiple background tasks to have a chance at the interrupt. After restoring the CPU registers the final routine executed during an interrupt sequence is an IRET.

4. Do not modify any CPU registers unless specifically stated in the discussion of that interrupt. Upon entry to an interrupt routine, the ES, DS,

Listing 6-1. Sample Code Segment

```
                        ;test for number of drives
INT     11H             ;retrieve status byte
PUSH    AL              ;temporarily store results
AND     AL,1B           ;isolate bit 0
TEST    AL,0            ;test for drives present
JZ      ZERO_DRIVES     ;if no drives, jump to routine
POP     AL              ;pop results back off stack
AND     AL,1100000B     ;isolate bits 5 and 6
MOV     CL,5            ;set for shift of 5 bit-places
SHR     AL,CL           ;shift data right 5 bits
INC     AL              ;increment results by 1 so
                        ;that true number of drives
                        ;is now in register AL
```

BX, CX, DX, AX, BP, SI, and DI registers contain information that you should save. Therefore, it is a good idea to push the information onto the stack and then restore it following the execution of the your routine.

Software Interrupt Summary

Table 6-4 summarizes the interrupts and describes the function, the initializing system (the Monitor program MFM-180, DOS, or application software), and the chapter where you can find a complete discussion. Interrupts that show dashes (--) in each column are either reserved for future use or generally not used by PC-compatible computers. Zenith Data Systems does not implement these interrupts in their computers or software.

Table 6-4. Interrupt Summary

INTERRUPT	INITIALIZED BY	CHAPTER	FUNCTION
00H	DOS	8	Divide by zero
01H	DOS	8	Single step
02H	MFM-180	8	Nonmaskable interrupt
03H	DOS	8	Software breakpoint
04H	DOS	8	Arithmetic overflow
05H	MFM-180	10	Print screen
06H	--	--	
07H	--	--	
08H	MFM-180	8	Timer (time-of-day)
09H	MFM-180	9	Key pressed
0AH	MFM-180	8	Real-time clock
0BH	Software	10	Communications (COM2)
0CH	Software	10	Communications (COM1)
0DH	Software	10	Alternate parallel printer (LPT2)
0EH	MFM-180	11	Floppy disk drive
0FH	Software	10	Parallel printer (LPT1)
10H	MFM-180	12	Video input/output
11H	MFM-180	8	Equipment configuration
12H	MFM-180	8	Memory size
13H	MFM-180	11	Disk input/output
14H	MFM-180	10	Serial input/output
15H	--	--	
16H	MFM-180	9	Keyboard input/output
17H	MFM-180	10	Printer input/output
18H	MFM-180	10	Parallel/serial configuration
19H	MFM-180	11	Booting an operating system
1AH	MFM-180	8	Set/read the time of day
1BH	MFM-180	9	Keyboard break
1CH	MFM-180	8	Tick timer
1DH	MFM-180	12	Video initialization
1EH	MFM-180	11	Disk parameters
1FH	MFM-180	12	Defining characters

System Organization

Table 6-5 provides an overall map of the system's addressable memory. Table 6-6 is a general map of the system's ports.

The information presented in these tables is subject to change as new products are introduced for this family of computers. The values represented by question marks vary, according the version of MS-DOS and the user-installed .COM and .EXE files.

NOTE: This computer does not support a light pen port. Some computer games use this feature and may not operate correctly.wc

Table 6-5. System Memory Map

ADDRESS RANGE	DESCRIPTION
00000H - 9FFFFH	System memory. This area is further divided by the Monitor program and DOS as follows:
(00000H - 003FFH)	Interrupt vector table (addresses)
(00400H - 004FFH)	Monitor program compatible data segment
(00500H - ?????H)	IO.SYS* (part of the DOS)
(?????H - ?????H)	DOS.SYS* (part of the DOS)
(?????H - ?????H)	Resident portion of COMMAND.COM (part of the DOS)
(?????H - ?????H)	User-installed .COM and .EXE files
(?????H - ?????H)	Transient portion of COMMAND.COM
(?????H - 9FFFFH)	Open for general use
A0000H - B7FFFH	Open Area
B8000H - BBFFFH	Video Ram Area

Table 6-5 (Continued). System Memory Map

ADDRESS RANGE	DESCRIPTION
BC000H - BFFFFH	Video Ram Image
C0000H - C7FFFH	Open Area
C8000H - C9FFFH	Hard Disk BIOS ROM Area
CA000H - CFFFFH	Open Area
D0000H - DFFFFH	EMS Window Area
E0000H - E7FFFH	System ROM 2
E8000H - EFFFFH	Open Area
F0000H - F07FFH	Scratchpad RAM
F8000H - FFFFFH	System ROM 1 (Monitor program)

NOTE: IO.SYS and DOS.SYS are representative of the names of hidden files that are part of MS-DOS. The actual names of these files can vary from version to version.

Table 6-6. System Port Map

PORT ADDRESSES	DESCRIPTION
000H - 00FH	DMA processor. (Gate Array 1)
020H - 021H	Interrupt controller. (82C59)
040H - 043H	System timer. (82C54)
050H - 05FH	Calendar RP5C15
060H - 063H	Peripheral interface status port. (Gate Array 1)
080H - 083H	DMA page registers. (Gate Array 1)

Table 6-6 (Continued). System Port Map

PORT ADDRESSES	DESCRIPTION
0A0H - 0AFH	Nonmaskable interrupt registers.
0C0H - 0D7H	Real-time clock.
0E0H - 0EFH	Reserved.
200H - 20FH	Game input/output port (not implemented).
258H	EMS Address.
278H - 27FH	Reserved.
2F8H - 2FFH	RS-232C serial input/output interface #2 (COM2) (used by the optional internal modem).
320H - 323H	Hard Disk Drive.
378H - 37FH	Parallel printer #1 (LPT1).
3D0H - 3DFH	Color graphics controller. (V6355)
3F0H - 3F7H	Floppy disk controller.
3F8H - 3FFH	RS-232C serial input/output interface #1 (COM1).

Monitor Program Jump Vectors

There are two jump vectors set up by the Monitor program, which can be used by applications programs. They are located at F000:FFF0, which is the power-up reset vector, and F000:FFED, which is an unconditional jump to the Monitor program (the Monitor program prompt will appear on the screen).

Chapter 7

System and CPU Interrupts

This chapter provides a description of the System and CPU interrupts, programming sound and addressing user memory.

Programming System and CPU Interrupts

Table 7-1 defines the general system interrupts used in this computer. For information pertaining to the use and programming of the software interrupts, refer to Chapter 6.

Table 7-1. System and CPU Interrupts

INTERRUPT	FUNCTION
00H	Divide by zero
01H	Single step
02H	Nonmaskable interrupt
03H	Software breakpoint
04H	Arithmetic overflow
08H	Timer (time-of-day)
0AH	Real-time clock
11H	Equipment configuration
12H	Memory size
1AH	Set/read the time of day
1CH	Tick timer

Divide by Zero (INT 00H)

INT 00H is the divide by zero interrupt. This interrupt occurs if a divide instruction produces a quotient too large to fit in the result register (such as dividing a value by 0). The operating system initializes this routine. The routine will print Divide Overflow and return control to the operating system.

You must set up a vector to intercept DIV and IDIV instructions if you do not want control returned to the operating system. That way, you can test for the error condition and prevent program control from returning to DOS. For instance, BASIC uses this method to retain control when a divide by zero condition occurs while executing BASIC functions.

Single Step (INT 01H)

INT 01H is the single step interrupt used for executing a single machine instruction at a time. The CPU calls this interrupt when an instruction executes with the trace flag (TF) set.

MS-DOS and the DEBUG command commonly use this interrupt, as does the Monitor program's trace command. The routine receives initialization instructions from the calling program (issued by the DOS or Monitor program, etc.). Because of this, you must also initialize the routines you want executed when you call this interrupt.

Nonmaskable Interrupt (INT 02H)

INT 02H is the nonmaskable interrupt. Hardware external to the 80C88 initializes this interrupt. Most Zenith Data Systems computers use this interrupt to indicate when a power-down condition has started. Also, the 8087 numeric data coprocessor uses this interrupt in its normal operation.

Normally, this interrupt is enabled. One exception however, is during the power-up sequence while the self-tests are being run. If you need to disable this interrupt (turn it off), send 00H to port A0H. To enable the NMI, send 80H to the same port (A0H).

Software Breakpoint (INT 03H)

INT 03H is the software breakpoint interrupt. When the processor encounters a breakpoint in a program, it will execute an INT 03H instruction, calling the interrupt routine.

The Monitor program's debugging routines and the MS-DOS DEBUG command allow you to set breakpoints in machine language code. Whenever the Monitor program or MS-DOS DEBUG encounter a breakpoint, control returns to the last command level.

Arithmetic Overflow (INT 04H)

INT 04H is the arithmetic overflow interrupt. The INTO instruction (interrupt on overflow) will generate this interrupt whenever the overflow flag (OF) is set. Arithmetic and Logic operations set the overflow flag.

Timer (Time-of-Day) (INT 08H)

INT 08H is the timer interrupt. The output of counter 1 of the 82C54 programmable interval timer initiates this interrupt 18.2159 times per second, or every .054897095 seconds. The CPU clock speed does not affect the operation of this timer. The timer performs such functions as keeping track of the time, timing out disk motors, and calling the timer tick interrupt (1CH).

In many computers, the timer keeps track of the time-of-day in a 32-bit word (sometimes called a double-word). When the count reaches approximately 1803D8H (1,573,848 decimal), a flag will be set to 1. This flag indicates that the timer has rolled past midnight into a new day. The value of this word is read using interrupt 1AH. The hardware establishes this interrupt as IRQ0.

In the Portable Computer, a real-time clock maintains the time of day. The real-time clock IC is described in Chapter 13.

Real-Time Clock (INT 0AH)

INT 0AH is the real-time clock alarm interrupt. The real-time clock starts when a hardware interrupt request (IRQ3) or when the alarm from the real-time clock IC takes place. Function 83H (event wait) and function 86H (wait) of INT 15 set and handle the alarm. The hardware interrupt request takes place approximately 1,024 times a second.

Most systems do not use this interrupt. The hardware within the computer essentially controls this interrupt. This makes the interrupt unique to individual machines such as this computer. Because of this fact, it should not be called by user programs.

Equipment Configuration (INT 11H)

INT 11H is the request equipment configuration interrupt. This interrupt reports the configuration of the equipment in a 16-bit word (register AX). Since this interrupt is common to PC-compatible equipment, you need to be aware of all possible responses and what they mean. Refer to Tables 7-2, 7-3, and 7-4 for a description of each bit in the data returned in register AX.

Table 7-2. Register AX Report from INT 11H

BIT	DESCRIPTION
0	If set (1), floppy disk drives are present in the system; if clear (0), no floppy disk drives are installed.
1	If set (1), the 8087 numeric data coprocessor is present; if clear (0), the 8087 is not installed.
2 - 3	These two bits indicate the device size of the RAM chips installed in the computer. Bits 2 and 3 are always set (1), indicating that either 64- or 256-kilobit devices are installed. Some early PC-type computers used 16- and 32-kilobit devices in memory.
4 - 5	These two bits indicate the initial video mode at powerup. Refer to Table 7-3.
6 - 7	These two bits report the number of floppy disk drives installed in the computer according to the hardware switch or firmware settings. Refer to Table 7-4.
8	Unused in Zenith Data Systems computers.
9 - 11	The value in these three bits indicate the number of RS-232 ports in the system. The standard input/output on this system emulates the number of IBM PC input/output ports, so the value is one.
12	If set (1), a game card is present; it is clear (0) in this computer.
13	Unused by Zenith Data Systems computers.
14 - 15	The value in these two bits is the number of printers installed.

Table 7-3. Bits 4 and 5: Video Initialization

BIT 4	BIT 5	DESCRIPTION
0	0	80 x 25 text mode on an EGA card
0	1	40 x 25 text mode on a color card
1	0	80 x 25 text mode on a color card
1	1	80 x 25 text mode on a monochrome card

NOTE: The Portable Computer emulates the color card, but does not support PC-compatible cards. Therefore, the only modes recognized are the 40 x 25 and 80 x 25 text modes on a color card.

Table 7-4. Bits 6 and 7: Disk Drive Count

BIT 6	BIT 7	DRIVE COUNT
0	0	1 Drive
0	1	2 Drives
1	0	3 Drives
1	1	4 Drives

NOTE: The third and fourth drives are external on the Portable Computer. Most PC-compatible computers do not support floppy disk drives 3 and 4. In those cases, drive C and D are usually reserved for hard disk drive partitions under MS-DOS.

Memory Size (INT 12H)

INT 12H is the memory size interrupt. It returns the number of contiguous 1K blocks of user memory in the system in register AX. For example, if the value in AX is 256 following this interrupt, then the computer has determined there is 256K of memory installed in the system.

Set/Read the Time of Day (INT 1AH)

INT 1AH is the set/read the time of day interrupt. It allows the programmer to set and/or read the internal clock's time and date, and to set the alarm on or off.

Register AH establishes the type of operation (read or write). The CX and DX registers store the value passed to or from the 32-bit counter. Register AL is serves as a flag to report that the value has rolled over into a new day since the last time the counter was read.

Most PC-compatible computers store the value that represents the time of day in a 32-bit time-of-day counter. This counter receives updates 18.2159 times a second. Therefore, 1 a.m. would be represented by a count of 65,577 and 12 noon would be 786,926. Register CX passes the high part of the count and would contain the value 1 for 1 a.m. and 12 (0CH) for 12 noon. Register DX passes the low part of the count and would contain 41 (29H) for 1 a.m. and 494 (1EEH) for 12 noon. The contents of register AL would depend upon whether the count had rolled over into a new day since the last counter read. If AL is zero, then the counter has not advanced past 24 hours. If AL is any value other than zero, then it had advanced past 24 hours.

To read the current time-of-day value from the time-of-day word, place a zero in the AH register and execute the interrupt. Upon return from the call, register CX will contain the high part of the count, and register DX will contain the low. If the count has rolled over, register AL will not contain zero. If AL contains a zero, the value has not rolled over to a new day since the last counter read.

To set the time-of-day value, place the high count in the CX register and the low count in the DX register. Place a 1 in register AH and execute the interrupt.

In the Portable Computer, a real-time clock IC keeps track of the date and time. This IC remains active even when you turn off the computer. The real-time clock uses the 32-bit time-of-day word for compatibility. When you first boot the computer with DOS, (version 3.2 or higher) it will automatically read the time of day from the clock IC. The value is then stored in the 32-bit time-of-day word. You could also use a utility program to accomplish the same thing. The end result is that software that depends on the 32-bit word will still work. In the case of the Portable Computer, a back-up real-time clock is also present.

Tick Timer (INT 1CH)

INT 1CH is the tick timer interrupt. It provides a means for you to produce CPU-independent timing loops in your programs. At powerup, this interrupt points to an IRET instruction.

The speed of the CPU clock frequency in PC-compatible computers is inconsistent from one machine to another. Furthermore, some models offer

switchable-speed CPU clock frequencies. This interrupt provides a convenient means of consistently controlling timing loops regardless of the CPU clock frequency.

Each time INT 08H (the timer interrupt) executes, the tick timer interrupt is called (18.2159 times a second). Therefore, if you use INT 1CH in a program, you must save the registers at the beginning of the routine and then restore them before returning to the main program.

Programming Sound

This computer does not contain a dedicated sound chip. However, the computer can generate tones and play them through the internal speaker. You can accomplish this in three different ways via the CPU gate array. All three of these methods may be implemented simultaneously.

1. Generate a pulse train by toggling a program control register bit.

2. Program the output of channel 2 of the timer/counter to deliver a sound waveform to the speaker.

3. Modulate the clock input to the timer/counter through a program-controlled input/output register bit.

Programming sound, particularly music, is a specialized application in itself and is beyond the intended scope of this manual. If you want to produce sounds or music, you should use one of the special application programs designed for that purpose or GW-BASIC, which contains statements to perform this type of task.

User Memory

In general, programming user memory is straight forward. Most operations can be performed directly from assembly or machine language programs and include the following:

- Reading and writing data to specific memory locations;
- Allocating or reserving specific locations in memory for specific purposes, such as for software interrupt routines or storing user-defined character fonts;
- Storing character strings and numeric values that are widely used by a number of applications;
- Rerouting interrupts to user-defined routines;
- CPU stack operations;
- Moving memory contents from one area to another; and
- Using the contents of RAM to control video graphics.

NOTE: The Portable Computer family of computers does not contain memory parity checking circuits. If you attempt to disable or enable parity generation or checking, nothing will happen.

Memory Address Format

The 80C88 uses a 2-part number to designate specific memory locations. The hexadecimal number consists of a 4-digit number to identify the segment address and a 4-digit number to identify the offset address within that segment. The format for this value is:

XXXX:YYYY

The first four digits actually represent a 5-digit hexadecimal memory address, the system performs an imaginary shift left (multiply by 16) on the value to arrive at the RAM bank and row to select. The second value selects the RAM column from the selected bank. For example, the value 3F3F:5B11 represents memory address 44F01H. 3F3F shifted left equals 3F3F0H. 5B11 is the offset added to the segment address. The result of adding 3F3F0H and 5B11H is 44F01H.

Since the least-significant digit of the segment is always zero, more than one combination of segment and offset can point to the same memory address. For instance, you can define the address from the previous example as 3F00:5F01. Either address definition will locate the same byte of user memory.

You determine the highest and lowest usable segment values by considering the memory location being defined. In the example address, the lowest usable segment value is 34F1H, since anything lower results in an offset greater than FFFFH. The highest usable segment value is 44F0H.

Chapter 8

Keyboard Interrupts and Codes

This chapter describes the keyboard interrupts and contains tables of the keyboard scan and up/down codes.

Programming Keyboard Interrupts

Table 8-1 describes the three interrupts that are applicable to the keyboard. Following a description of these interrupts are tables that describe the codes generated by the keyboard.

Table 8-1. Keyboard Interrupts

INTERRUPT	FUNCTION
09H	Key Pressed
16H	Keyboard Input/Output
1BH	Keyboard Break

Key Pressed (INT 09H)

INT 09H is the key pressed interrupt executed each time you press or release a key. The interrupt routine:

- reads the key from the keyboard register.
- encodes the key or notes the action if the key is a shift or control key.
- stores the code, if produced, in the keyboard buffer.

Applications or system software can then retrieve the code when it needs it.

Do not change the action of this interrupt routine since it directly affects the action of the keyboard.

Keyboard Input/Output (INT 16H)

INT 16H is the keyboard input/output interrupt. You can use this interrupt to perform three routine keyboard operations:

- Function code 00H retrieves a key code from the keyboard buffer.
- Function code 01H checks the keyboard buffer to see if any codes are in it.
- Function code 02H reports the status of the keyboard shift and control keys.

Function Code 00H: Get Character — This function code causes the interrupt to read a character from the keyboard buffer. When you press a key, INT 09H places the corresponding character code in the keyboard buffer. Function 00H retrieves and removes the code from the keyboard buffer and places it in the AL register. The AH register contains the scan code for the pressed key. If the keyboard buffer is empty, the routine waits until someone presses a key. The control keys do not generate codes, but affect the codes generated by the other keys.

Function Code 01H: Check Keyboard Buffer — This function code causes the interrupt to check the status of the keyboard buffer. If the buffer is empty, this function sets the zero flag (ZF).

If the buffer contains key codes, this function clears the zero flag to zero. Also, the function places the character code in the register AL. It also places the scan code in the register AH.

NOTE: This operation does not remove any codes from the keyboard buffer. Therefore, if you execute function code 01H followed by function code 00H, the interrupt returns the same key codes. Only function code 00H removes key codes from the buffer.

Function Code 02H: Get Keyboard Status — Two bytes of memory — 0040:0017 and 0040:0018 — hold the keyboard status. This function code reports the status stored at location 040:0017. Table 8-2 describes each bit in the byte stored in the AL register. If the report indicates a key was down, the key was being held down at the time the computer executed this interrupt function.

Table 8-2. Keyboard Status Report

BIT	DESCRIPTION
0	If set (1), the right SHIFT key was down.
1	If set (1), the left SHIFT key was down.
2	If set (1), the CTRL key was down.
3	If set (1), the ALT key was down.
4	If set (1), the SCROLL LOCK mode was active.
5	If set (1), the keypad lock mode was active.
6	If set (1), the CAPS LOCK mode was active.
7	If set (1), the INS (insert) mode was active.

Use your own routine to read and interpret the keyboard status byte stored at location 0040:0018. Table 8-3 describes each bit in the byte.

Table 8-3. Keyboard Status (0040:0018)

BIT	DESCRIPTION
0	Not used.
1	Not used.
2	Not used.
3	If set (1), the CTRL-NUM LOCK (pause) mode was active.
4	If set (1), the SCROLL LOCK key was down.
5	If set (1), the NUM LOCK key was down.
6	If set (1), the CAPS LOCK key was down.
7	If set (1), the INS key was down.

Keyboard Break (INT 1BH)

PC-compatible software defines the CTRL-BREAK key combination as keyboard break. INT 09H executes an INT 1BH when it detects CTRL-BREAK.

The service routine for this interrupt must end with an IRET instruction so the INT 09H service routine also ends properly. When you turn the computer on, the Monitor program provides the service routine with only the IRET instruction. That way, if you press the CTRL-BREAK during the self-tests, nothing will happen.

If you want your program recognize the CTRL-BREAK key combination, you must write the service routine for this interrupt. GW-BASIC creates an interrupt routine to halt execution of a BASIC program whenever you press the CTRL-BREAK key sequence.

Be aware that one or more of the following conditions may exist when you execute the CTRL-BREAK service routine. Your routine should accommodate the possibilities. Refer to Chapter 14 for a discussion on the 82C59 interrupt controller.

- The CTRL-BREAK may occur during interrupt processing. Then your routine must send one or more end-of-interrupt commands to the interrupt controller.

- If an input/output operation is in process when you press CTRL-BREAK, you must reset the input/output device.

- Programs that use this interrupt must not chain to the previous owner of this interrupt. Also, you must restore the interrupt when you finish processing.

Remember, your service routine for 1BH must perform an IRET to make sure that the interrupt restores properly. Do not link your service routine to the next one.

Keyboard Codes

To duplicate all the functions of the PC-compatible keyboard, the Portable computer generates a wide range of keycodes. Tables 8-4 through 8-9 summarize the various key scan codes and functions that the keyboard produces. You can use the user-executed keyboard test program to examine the following codes.

Alphabetic Keys

The PC-compatible keyboard interface provides system software with maximum flexibility for defining keyboard operation. To accomplish this the keyboard returns complex hexadecimal codes rather than simple ASCII codes. Also, most keys can generate both make and break codes, commonly called up and down codes. For example, the ESC key (defined internally as key 1) produces 01 hexadecimal when pressed (down) and 81 hexadecimal when released (up). The keyboard controller can produce codes with or without control keys pressed or under

event-driven conditions required by the application. Finally, you can produce different codes by placing the keyboard in one of the five operating modes. Chapter 2 discusses the various operating modes.

Figure 8-1 illustrates the alphabetic keys that generate the 2-byte scan codes listed in Table 8-4. The least-significant byte is the ASCII code.

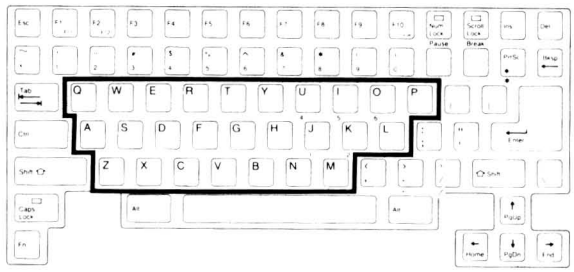


Figure 8-1. Alphabetic Keys

Table 8-4. Alphabetic Key Scan Codes

KEY	RAW CODE	WITH SHIFT KEY	WITH CTRL KEY	WITH ALT KEY	WITH FN KEY	CURSOR MODE
A	1E61H	1E41H	1E01H	1E00H	--	
B	3062H	3042H	3002H	3000H	--	
C	2E63H	2E43H	2E03H	2E00H	--	
D	2064H	2044H	2004H	2000H	--	
E	1265H	1245H	1205H	1200H	--	
F	2166H	2146H	2106H	2100H	--	
G	2267H	2247H	2207H	2200H	--	
H	2368H	2348H	2308H	2300H	--	
I/5	1769H	1749H	1709H	1700H	4C35H	
J/1	246AH	244AH	240AH	2400H	4F31H	4F00H
K/2	256BH	254BH	250BH	2500H	5032H	5000H
L/3	266CH	264CH	260CH	2600H	5133H	5100H
M/0	326DH	324DH	320DH	3200H	5230H	5200H
N	316EH	314EH	310EH	3100H	--	
O/6	186FH	184FH	180FH	1800H	4D36H	4D00H
P	1970H	1950H	1910H	1900H	--	
Q	1071H	1051H	1011H	1000H	--	
R	1372H	1352H	1312H	1300H	--	
S	1F73H	1F53H	1F13H	1F00H	--	
T	1474H	1454H	1414H	1400H	--	
U/4	1675H	1655H	1615H	1600H	4B34H	4B00H
V	2F76H	2F56H	2F16H	2F00H	--	
W	1177H	1157H	1117H	1100H	--	
X	2D78H	2D58H	2D18H	2D00H	--	
Y	1579H	1559H	1519H	1500H	--	
Z	2C7AH	2C5AH	2C1AH	2C00H	--	

NOTES:

The raw code column contains codes produced by the keyboard in its default state, with no other keys pressed.

The codes produced with the FN key are the same codes produced by those keys in the keypad locked numeric mode.

An entry that contains a dash (--) indicates that the output code is the same as the raw code. A blank indicates that no output results from this combination.

Numeric and Punctuation Keys

The numeric and punctuation keys, shown in Figure 8-2, include the following groups of keys:

- the numbers 0 - 9
- the common punctuation marks
- the special programming characters that make up the remainder of the printable ASCII character set.

Each of these keys has two distinct characters. You generate the upper character whenever you press either SHIFT key. The CAPS LOCK key does not affect the operation of these keys. Table 8-5 lists the numeric and punctuation key scan codes. The ASCII code is the least-significant byte of the scan code.

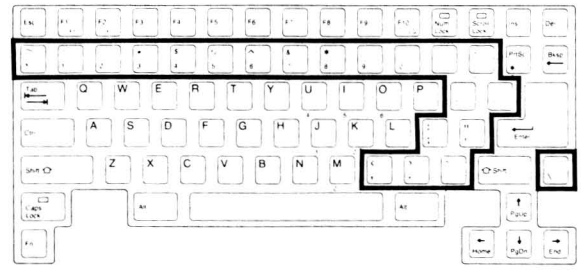


Figure 8-2. Numeric and Punctuation Keys

Table 8-5. Numeric and Punctuation Key Scan Codes

KEY	RAW CODE	WITH SHIFT KEY	WITH CTRL KEY	WITH ALT KEY	WITH FN KEY	CURSOR MODE
1/!	0231H	0221H		7800H		
2/@	0332H	0340H	0300H	7900H		
3/#	0433H	0423H		7A00H	--	
4/$	0534H	0524H		7B00H	--	
5/%	0635H	0625H		7C00H	--	
6/^	0736H	075EH	071EH	7D00H	--	
7/&/7	0837H	0826H		7E00H	4737H	4700H
8/*/8	0938H	092AH		7F00H	4838H	4800H
9/(/9	0A39H	0A28H		8000H	4939H	4900H
0/)	0B30H	0B29H		8100H	--	
-/_/-	0C2DH	0C5FH	0C1FH	8200H	4A2DH	4A2DH
=/+/+	0D3DH	0D2BH		8300H	4E2BH	4E2BH
'/~	2960H	297EH			--	
[/{/*	1A5BH	1A7BH	1A1BH		372AH	372AH
]/}	1B5DH	1B7DH	1B1DH		--	--
;/:	273BH	273AH			--	--
'/"/=	2827H	2822H			0D3DH	0D3DH
,/<	332CH	333CH			--	--
./>/.	342EH	343EH			532EH	5300H
//?//	352FH	353FH			--	--
\/|	2B5CH	2B7CH	2B1CH		--	--

NOTES: The raw code column contains codes produced by the keyboard in its default state, with no other key pressed.
The codes produced with the FN key are the same codes produced by those keys in the keypad lock numeric mode.
An entry that contains a dash (--) indicates that the output code is the same as the raw code. A blank indicates that no output results from this combination.

Control and Function Keys

The control keys, shown in Figure 8-3, do not generate their own codes, but modify the codes produced by other keys. They are described in Table 8-6 and their scan codes are listed in Table 8-7.

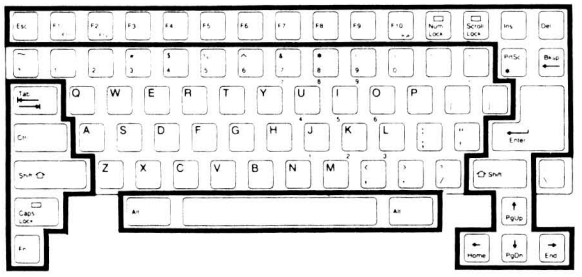


Figure 8-3. Control and Function Keys

Table 8-6. Control and Function Keys

KEY	DESCRIPTION
ALT	Alternate. Modifies the codes produced by other keys. You can also use the ALT key to generate any hexadecimal code from 0 to FFH (0 to 255 decimal). Lock the keypad, then press and hold the ALT key while you enter the decimal value through the keypad. When you release the ALT key, the keyboard controller will generate the hexadecimal value of the decimal number you entered.
BACK SPACE	Back space. Normally generates ASCII code 08H.
BREAK	Break. Normally used to break program execution.
CAPS LOCK	Caps Lock. Toggles the CAPS LOCK mode. The SHIFT keys reverse the action of the CAPS LOCK mode. An indicator, located next to the key, lights when the CAPS LOCK mode is active.
CTRL	Control. Modifies the codes produced by other keys.
(cursor keys)	Cursor/screen control. Software uses the codes produced by these keys to move the cursor up, down, left, right, and control screen movement.
DEL	Delete. Most text editing and word processing software uses the DEL key code to delete material. You can also use the DEL key the CTRL and ALT keys to reset the computer.
END	End. Most text editing and word processing software uses the END key code for screen, page, or document manipulation.

Table 8-6 (Continued). Control and Function Keys

KEY	DESCRIPTION
ESC	Escape. Normally generates ASCII code 1BH.
F1 - F12	Function keys. The codes produced by these keys work with different software products that generate "user defined" functions.
FN	Function. Modifies codes produced by other keys. This key is unique to this computer.
INS	Insert. Most text editing and word processing software uses the INS key code to insert material. You can also use the INS key the CTRL and ALT keys to reset the computer.
HOME	Home. Most text editing and word processing software uses the HOME key code for screen, page, or document manipulation.
NUM LOCK	Numbers Lock. Serves two functions. 1) When used with the FN key (FN-NUM LOCK), toggles the keypad lock. The indicator, next to the NUM LOCK key, lights when the keypad lock is on. 2) While the keypad lock is active, this key toggles the keypad between numeric and cursor control modes. The SHIFT keys reverse the numeric and cursor control modes.
PGDN	Page Down. Most text editing and word processing software uses the PGDN key code for screen, page, or document manipulation.
PGUP	Page Up. Most text editing and word processing software uses the PGUP key code for screen, page, or document manipulation.
PRTSC	Print Screen. Sends the contents of the screen to the LPT1 device.
ENTER	Enter. Normally generates ASCII code 0DH. Software usually adds ASCII code 0AH.
SCROLL LOCK	Scroll Lock. Some software uses the SCROLL LOCK mode to control screen scrolling. An indicator, next to the SCROLL LOCK key, lights when scroll lock is active.
SHIFT	Shift. Modifies the codes produced by other keys. Reverses the action of CAPS LOCK and NUM LOCK.
(space bar)	Space. Normally generates ASCII code 20H.
TAB	Tab. Normally generates ASCII code 09H.

Table 8-7. Control and Function Key Scan Codes

KEY	RAW CODE	WITH SHIFT KEY	WITH CTRL KEY	WITH ALT KEY	WITH FN KEY	CURSOR MODE
(space bar)	3920H	--	--	--		
*/PRTSC	372AH	+	7200H		--	
ALT						
BACK SPACE	0E08H	--	0E7FH	0E7FH		
CAPS LOCK						
CTRL						
DEL	5300H	--			--	--
END/-->	4D00H	--	7400H	0006H*	4F00H	--
ESC	+	+	+	+	5500H	+
F1	3B00H	5400H	5E00H	6800H	8500H	
F10/LCD/RGB	4400H	5D00H	6700H	7100H	+	
F2	3C00H	5500H	5F00H	6900H	8600H	
F3	3D00H	5600H	6000H	6A00H	--	
F4	3E00H	5700H	6100H	6B00H	--	
F5	3F00H	5800H	6200H	6C00H	--	
F6	4000H	5900H	6300H	6D00H	--	
F7	4100H	5A00H	6400H	6E00H	--	
F8	4200H	5B00H	6500H	6F00H	+	
F9	4300H	5C00H	6600H	7000H	+	
FN						
FN-F1 (F11)	8500H	8700H	8900H	8B00H		
FN-F2 (F12)	8600H	8000H	8A00H	8C00H		
HOME/<--	4B00H	--	7300H	0004H*	4700H	--
INS	5200H	+			--	--
NUM LOCK/PAUSE						
PGDN/v	5000H	--		0002H*	5100H	--
PGUP/^	4800H	--		0008H*	4900H	--
ENTER	1C0DH	--	1C0AH		--	
SCROLL LOCK/BREAK	+	+	+	+	+	
SHIFT						
TAB	0F09H	0F00H			--	

NOTES: The raw code column contains codes produced by the keyboard in its default state, with no other keys pressed.
The INT 09H routine generates the codes marked with the asterisk when you release the ALT key.
An entry that contains a plus (+) indicates that a programmed function takes place during the user-executed keyboard test program. The program does not display a code in these cases.
An entry that contains a dash (--) indicates that the output code is the same as the raw code. A blank indicates that no output results from this combination.

Up/Down Key Codes

Each time you press most keys, the keyboard produces two distinct codes. Pressing a key produces the down code. Releasing the key produces the up code. The up and down codes are distinct from the scan codes the result when you use the INT 16H function. Since INT 09H interprets the codes, you have to modify it to access the up/down codes. Do

not alter the interrupt unless you are an experienced programmer. Table 8-8 lists all the up/down codes.

Table 8-8. Up/Down Codes

KEY	DOWN	UP	FN-DOWN	FN-UP
ESC	01H	81H	55H	D5H
F1/F11	3BH	BBH	57H	D7H
F2/F12	3CH	BCH	58H	D8H
F3	3DH	BDH	3DH	BDH
F4	3EH	BEH	3EH	BEH
F5	3FH	BFH	3FH	BFH
F6	40H	C0H	40H	C0H
F7	41H	C1H	41H	C1H
F8	42H	C2H	5EH	DEH
F9	43H	C3H	5FH	DFH
F10/LCD/RGB	44H	C4H	5DH	DDH
NUM LOCK/ PAUSE	45H	C5H		
SCROLL LOCK/ BREAK	46H	C6H	2A37B7AAH	
INS	E052H	E0D2H	E052H	E0D2H
DEL	E053H	E0D3H	E053H	E0D3H
~/'	29H	A9H	29H	A9H
!/1	02H	82H		
@/2	03H	83H		
#/3	04H	84H	04H	84H
$/4	05H	85H	05H	85H
%/5	06H	86H	06H	86H
^/6	07H	87H	07H	87H
&/7/7	08H	88H	47H	C7H
*/8/8	09H	89H	48H	C8H
(/9/9	0AH	8AH	49H	C9H
)/0	0BH	8BH	0BH	8BH
_/-/-	0CH	8CH	4AH	CAH
+/=/+	0DH	8DH	4EH	CEH
PRTSC/*/*	37H	B7H	37H	B7H
BACK SPACE	0EH	8EH	E04CH	E0CCH
TAB	0FH	8FH	0FH	8FH
Q	10H	90H	10H	90H
W	11H	91H	11H	91H
E	12H	92H	12H	92H
R	13H	93H	13H	93H
T	14H	94H	14H	94H
Y	15H	95H	15H	95H
U/4	16H	96H	4BH	CBH
I/5	17H	97H	4CH	CCH
O/6	18H	98H	4DH	CDH
P	19H	99H	19H	99H
{/[	1AH	9AH	37H	B7H
}/]	1BH	9BH	1BH	9BH
CTRL	1DH	9DH	1DH	9DH
A	1EH	9EH	1EH	9EH
S	1FH	9FH	1FH	9FH
D	20H	A0H	20H	A0H
F	21H	A1H	21H	A1H
G	22H	A2H	22H	A2H
H	23H	A3H	23H	A3H
J/1	24H	A4H	4FH	CFH
K/2	25H	A5H	50H	D0H
L/3	26H	A6H	51H	D1H
:/;	27H	A7H	27H	A7H
"/'	28H	A8H	0DH	8DH
ENTER	1CH	9CH	1CH	9CH
Left SHIFT	2AH	AAH	2AH	AAH
Z	2CH	ACH	2CH	ACH
X	2DH	ADH	2DH	ADH
C	2EH	AEH	2EH	AEH
V	2FH	AFH	2FH	AFH
B	30H	B0H	30H	B0H
N	31H	B1H	31H	B1H
M/0	32H	B2H	52H	D2H
</,	33H	B3H	33H	B3H
>/./.	34H	B4H	53H	D3H
?////	35H	B5H	35H	B5H
Right SHIFT	36H	B6H	36H	B6H
|/^	2BH	ABH	2BH	ABH
CAPS LOCK	3AH	BAH	3AH	BAH
ALT	38H	B8H	38H	B8H
Space bar	39H	B9H		
ALT	38H	B8H	38H	B8H
<--/HOME	E04BH	E0CBH	E047H	E0C7H
^/PGUP	E048H	E0C8H	E049H	E0C9H
v/PGDN	E050H	E0D0H	E051H	E0D1H
-->/END	E04DH	E0CDH	E04FH	E0CFH

Chapter 9

Input/Output Interrupts

This chapter describes the interrupts used for communications, printers, and serial and parallel communications.

Programming Input/Output Interrupts

Table 9-1 defines the input/output interrupts described in this chapter. Refer to Chapter 6 for information on how to use and program these interrupts.

Table 9-1. Input/Output Interrupts

INTERRUPT	FUNCTION
05H	Print screen
0BH	Communications (COM2)
0CH	Communications (COM1)
0DH	Alternate parallel printer (LPT2)
0FH	Parallel printer (LPT1)
14H	Serial input/output
17H	Printer input/output
18H	Parallel/serial configuration

Print Screen (INT 05H)

INT 05H sends the contents of the screen to the LPT device (usually a printer). Valid characters, or those that match a valid character, are sent to the printer. All other shapes will be ignored by the default print screen routine. This interrupt routine performs the same function as when you press the SHIFT-PRTSC key.

To aid in using the print screen function, the system reserves a byte of memory at location 0050:0000 as a status byte. While the print screen routine is executing, this byte is set to 01H. This serves as a flag used by the print screen routine to prevent additional print screen requests from being processed while the routine is executing. Upon completion of the print screen routine, the status byte changes to 00H if no errors occurred, or FFH if an error did occur. The error is usually caused by a printer time-out.

Communications (INT 0BH and INT 0CH)

The INT 0BH and INT 0CH instructions provide serial communications capabilities through COM1 (INT 0CH) and COM2 (INT 0BH). The system reserves COM2 for use with the optional internal modem. These interrupts are normally reserved for use with applications software. If you want to use these interrupts for your own communication activities, you must supply your own input/output routines.

NOTE: Interrupts INT 14H and INT 18H provide support for Monitor program and MS-DOS serial communications.

Parallel Printer (INT 0DH and INT 0FH)

INT 0FH (LPT1) and INT 0DH (LPT2) are the parallel communications interrupts, generally used with printers. These interrupts are normally reserved for use with applications software. If you want to use these interrupts, you must supply your own input/output routines.

NOTE: INT 17H provides complete support for parallel communications.

Serial Input/Output (INT 14H)

INT 14H allows you to perform the serial functions described in Table 9-2 and explained in the following paragraphs. For more information, refer to "Parallel/ Serial Configuration" in this chapter.

You must place a specific function code in the AH register before you can execute this interrupt. Register DX must be loaded with 0 for COM1 or 1 for COM2.

NOTE: The addressed serial device must be connected to the system. Since this Computer has only one serial port, always set the DX register to 0 for COM1.

Table 9-2. Serial Input/Output Function Codes

CODE	DESCRIPTION
00H	Initialize the serial input/output port
01H	Send character to the serial port
02H	Receive character from the from serial port
03H	Read communications status

Function Code 00H: Initialize the Serial Input/Output Port — This function code initializes the parameters of the serial port according to the value in the AL register. Table 9-3 through Table 9-6 define the possible values for this code. Register DX must be loaded with 0 for the COM1 port.

Table 9-3. Mode-Select Byte Breakdown

BIT	DESCRIPTION
0 - 1	Sets word length (refer to Table 9-4)
2	Sets number of stop bits: 0 = 1 stop bit, 1 = 2 stop bits
3 - 4	Sets parity selection (refer to Table 9-5)
5 - 7	Sets baud rate (refer to Table 9-6).

Table 9-4. Word Length Selection

BIT 1	BIT 0	WORD LENGTH
0	0	5 bits (not supported in PC-compatible computers)
0	1	6 bits (not supported in PC-compatible computers)
1	0	7 bits
1	1	8 bits

NOTE: When you transmit or receive a word with a length of less than eight bits, the character will be converted by the asynchronous communications element. The asynchronous communications element will convert a transmitted word from a full 8-bit word to the specified length. A received word will be converted from the received length to a full 8-bit word. Extra bits will be ignored or set to 0.

Table 9-5. Parity Selection

BIT 4	BIT 3	SELECTION
0	0	No parity
0	1	Odd parity
1	0	No parity
1	1	Even parity

Table 9-6. Baud Rate Selection

BIT 7	BIT 6	BIT 5	BAUD RATE
0	0	0	110
0	0	1	150
0	1	0	300
0	1	1	600
1	0	0	1200
1	0	1	2400
1	1	0	4800
1	1	1	9600

NOTE: Although the hardware is capable of handling other baud rates, these are the only rates supported by the interrupt.

Upon return from the routine, the 16-bit AX register will contain the serial port status report. The eight bits of register AX that make up register AH will contain the line control status (refer to Table 9-7). The eight bits of register AX that make up register AL will contain the modem control status (refer to Table 9-8). Function code 03H of this interrupt will return the same report.

Function Code 01H: Send Character to the Serial Port — This function code transmits the byte stored in the AL register out the specified serial port. Register DX must contain a 0 for the COM1 port.

Upon return from the routine, the 16-bit register AX will contain the serial port status report. The eight bits of register AX that make up register AH will contain the line control status (refer to Table 9-7). The eight bits of register AX that make up register AL will contain the modem control status (refer to Table 9-8). Function code 03H of this interrupt will return the same report.

NOTE: In place of the timeout error indicated in Table 9-7, the most-significant bit of register AH will

contain the transmit status bit. If the routine cannot transmit the character placed in register AL for one reason or another, the transmit status bit will be set (1).

Function Code 02H: Receive Character from the Serial Port — This function code loads the byte at the specified serial port into the AL register. Register DX must contain a 0 for the COM1 port.

Upon return from the routine, the 16-bit register AX will contain the serial port status report. The eight bits of register AX that make up register AH will contain the line control status (refer to Table 9-7). The eight bits of register AX that make up register AL will contain the modem control status (refer to Table 9-8). Function code 03H of this interrupt will return the same report.

NOTE: Bits 7, 4, 3, 2, and 1 of register AH will contain the data transfer status (refer to Table 9-7). If the contents of register AH is 0, the routine has read the byte properly into AL. If AH is not 0, some type of error occurred. In these cases, time-out errors refer to either the absence of either the data set ready (DSR) signal or the clear to send (CTS) signal.

Function Code 03H: Read Communications Status — This function code will return a report of the status of the specified communications port. Register DX must contain a 0 for the COM1 port.

The status report returns as a bit-mapped value in register AX. The eight bits of register AX that make up register AH contain the line control status (refer to Table 9-7). The eight bits of register AX that make up register AL contain the modem control status (refer to Table 9-8).

Table 9-7. Line Control Status

BIT	DESCRIPTION
0	When set (1), the received data is ready.
1	When set (1), an overrun error has occurred. Another character arrived before the system read the previously received character.
2	When set (1), a parity error has occurred. The parity of the incoming character did not match the programmed parity selection.

Table 9-7 (Continued). Line Control Status

BIT	DESCRIPTION
3	When set (1), a framing error has occurred. The first bit of the word and the stop bit handle framing. Either the system incorrectly received the transmitted character or the total number of characters in the received word did not match the programmed selection.
4	When set (1), remote BREAK key operation was detected. Some terminals have a BREAK key that, when pressed, will hold the data line low.
5	When set (1), the transmitter holding register is empty. The serial input/output channel is ready to receive another character for transmission.
6	When set (1), the transmitter shift register is empty. The specified serial input/output channel is not currently transmitting.
7	When set (1), a time-out error occurred. The receiving device did not respond within a reasonable period of time. A printer, when off line, will typically return this type of error.

Table 9-8. Modem Control Status

BIT	DESCRIPTION
0	When set (1), the clear to send (CTS) line has changed state.
1	When set (1), the data set ready (DSR) status has changed.
2	When set (1), the tailing edge of the ring signal has been detected.
3	When set (1), the carrier detect (CD) signal has changed state.
4	This bit reflects the current state (high or low) of the clear to send line.
5	This bit reflects the current state (high or low) of the data set ready line.
6	This bit reflects the current state (high or low) of the ring indicator line.
7	This bit reflects the current state (high or low) of the carrier detect line.

Printer Input/Output (INT 17H)

INT 17H performs input/output functions for the parallel port on the computer. Before executing the interrupt, load the AH register with one of the function codes described in Table 9-9. This interrupt uses the parallel configuration table, set up by interrupt 18H, to complete its operation. Refer to "Parallel/Serial Configuration" in this chapter.

Table 9-9. Printer Input/Output Function Codes

FUNCTION CODE	DESCRIPTION
00H	Print the next character. Send the character in the AL register to the port. If the parallel device does not return a ready status within a reasonable amount of time, set bit 0 of the AH register (1), otherwise the register will return the status report (refer to Table 9-10). Load the DX register with the port number to be used (00H for LPT1, 01H for LPT2, etc.)
01H	Initialize the printer port. Register AH will return the status report following execution of the interrupt (refer to Table 9-10).
02H	Read status of printer port. This function returns the status report (refer to Table 9-10) in register AH.

Table 9-10. Parallel Printer Status Report

BIT	DESCRIPTION
0	A timeout error has occurred. The parallel device is probably not ready, perhaps off line.
1	Not used.
2	Not used.
3	An input or output error has occurred.
4	The device is on line.
5	The out of paper signal is active.
6	The receiver acknowledges the transmitted character.
7	The device is busy or in an error state.

Parallel/Serial Configuration (INT 18H)

INT 18H serves as a pointer to the parallel and serial channel device tables. These are the tables used by MS-DOS and many application packages to configure the computer's serial and parallel ports. It can also route the parallel printer output to a serial communications channel.

NOTE: In IBM PC computers that contain BASIC in ROM, this interrupt points to the BASIC program. Programs that make a specific call to BASIC ROM routines will not work on Zenith Data Systems computers, including the Portable Computer.

In order to maintain compatibility between various versions of MS-DOS, the Monitor program loads the default configuration tables. MS-DOS and application programs can either modify these tables or install new tables and modify the pointers accordingly.

Figure 9-1 illustrates the format of the serial and parallel device parameter mapping table.

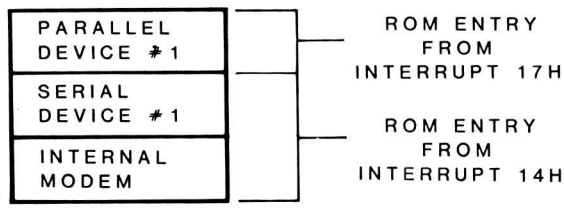


Figure 9-1. Serial and Parallel Device Layout

Parallel Format

Table 9-11 shows the format for each of the three, 4-byte parallel maps.

Table 9-11. Parallel Map Format

BYTE	PARAMETER DESCRIPTION
1	Performs remapping, parity, and uppercase/lowercase handling as described in Table 9-12.
2	Contains the ASCII value of the pad character used after a CR has been sent.
3	Contains the number of pad characters to send.
4	Contains the timeout value for the parallel device.
NOTE:	You can modify these parameters by using the MS-DOS CONFIGUR utility supplied with Zenith Data Systems versions of MS-DOS for this computer.

Table 9-12. Parallel Map Byte #1

BIT	DESCRIPTION
0 - 3	Remapping. The value stored in these bits determines the parallel device mapping. If the value is 0, no mapping takes place. If the value is 1, the output is remapped to COM1. If the value is 2, the output is remapped to COM2 (Since the Portable computer has only one COM port, do not use this option). Values of 3 or greater are not used.
4	If set (1), strip parity on output.
5	Not used.
6	If set (1), convert (map) lowercase output to uppercase.
7	Not used.

Serial Format

Table 9-13 shows the format for the two, 8-byte serial maps.

Table 9-13. Serial Map Format

BYTE	DESCRIPTION
1	Bit-mapped to indicate the type of handshaking that will be used by this channel (refer to Table 9-14).
2	Bit-mapped to indicate the attributes used by this channel for code transmission (refer to Table 9-15).
3	Contains the ASCII code for the pad character sent following a CR.
4	Contains the number of pad characters to send.
5	Used as a flag for burst transmission with ETX/ACK handshaking.
6	Used as a counter for ETX/ACK handshaking.
7	Bit-mapped to indicate the parameters used during initialization of this channel. (refer to Table 9-16).
8	Reserved.

Table 9-14, 9-15, and 9-16 contain the bit-mapped information for byte #1 (handshaking), #2 (attributes), and #7 (initialization parameters), respectively of the serial map.

Table 9-14. Byte #1 (Handshaking)

BIT	DESCRIPTION
0	Compatibility bit. If set (1), use IBM-compatible DTR and RTS signals for handshaking when active high. If clear (0), bit 1 determines the type of handshaking protocol used (hardware or software).
1	Hardware/software protocol bit. If set (1), use software handshaking ; if clear (0), use hardware handshaking.
2	DC1/DC3, EXT/ACK, DTR handshaking, based on the condition of bit 1. With bit 1 set, a set condition (1) on this bit indicates DC1/DC3 protocol. If bit 2 is clear (0), use ETX/ACK protocol. With bit 1 clear, a set condition (1) on bit 2 indicates DTR handshaking protocol. If bit 2 is clear (0), DTR is not in use (see bit 4).
3	DTR Polarity bit, based on the condition of bits 1 and 2. If bit 1 is clear (0) (hardware handshaking is in effect) and bit 2 set (1) (DTR handshaking is used), this bit determines the polarity of DTR handshaking. If set (1), DTR handshaking is active high; if clear (0), DTR handshaking is active low. If bit 1 is clear, software handshaking is in effect and this bit indicates if software is waiting for a handshake signal. If set (1), it is waiting; if clear (0), it is not.
4	RTS handshaking bit, based on the condition of bit 2. If bit 1 is clear (0), hardware handshaking is in effect. If set (1), RTS handshaking is used; if clear (0), RTS handshaking is not used.
5	RTS polarity bit, based on the condition of bit 1 and bit 4. If bit 1 is clear (0), hardware handshaking is in effect and this bit indicates the polarity of RTS handshaking if bit 4 is set (1). If set (1), RTS handshaking is active high; if clear (0), RTS handshaking is active low.

Table 9-15. Byte #2 (Parity and Case Mapping)

BIT	DESCRIPTION
0	If clear (0), do not strip the parity bit on input; if set (1), strip the parity bit on input.
1	If clear (0), do not strip the parity bit on output; if set (1), strip the parity bit on output.
	If set (1), convert (map) lowercase alphabetic characters to uppercase on input; if clear (0), do not convert lowercase alphabetic characters to uppercase on input.
3	If set (1), convert (map)lowercase alphabetic characters to uppercase on output; if clear (0), do not convert lowercase alphabetic characters to uppercase on output.

Table 9-16. Byte #7 (Word Length, Stop Bits, Parity, and Baud Rate)

BIT	DESCRIPTION
0 - 1	Determines the number of bits in a word. If the value is 0 (00), use five bits (not used in most PC-compatible applications). If the value is 1 (01), use six bits (not used in most PC-compatible applications). If the value is 2 (10), use seven bits. If the value is 3 (11), use eight bits.
2	Determines the number of stop bits. If set (1), use two stop bits. If clear (0), then use 1 stop bit. Even though the hardware is capable of 1-1/2 stop bits, you cannot program this interrupt to use that value. You must program the hardware directly.
3 - 4	Determines the parity used. If the value is 0 (00) or 2 (10), no parity; a 1 (01), indicates odd parity; a 3 (11), indicates even parity.
5 - 7	Determines the baud rate used (refer to Table 9-6).

Chapter 10

Disk Drive Interrupts

This chapter describes the interrupts used for disk input, output, and setup.

Programming Disk Drive Interrupts

Table 10-1 defines the disk drive interrupts described in this chapter. Refer to Chapter 6 for information on how to use and program these interrupts.

NOTE: Any interrupt that is not initialized will jump to a return (RET) function (no operation).

Table 10-1. Disk Drive Interrupts

INTERRUPT	FUNCTION
0EH	Floppy disk drive
13H	Disk input/output
19H	Booting an operating system
1EH	Disk parameters

Floppy Disk Drive (INT 0EH)

INT 0EH provides communication with floppy disk drives. The Monitor program uses this interrupt to provide disk input/output at the hardware level. User programs should not modify this interrupt routine. INT 13H is more useful for most disk communication.

Disk Input/Output (INT 13H)

INT 13H provides communications with the system's disk drives. Table 10-2 lists the port addresses associated with the disk drives.

Table 10-2. Disk Drive Port Addresses

PORT	REGISTER
3F2H	Floppy disk digital output port
3F4H	Floppy disk controller status port
3F5H	Floppy disk controller data port
320H	Hard disk drive #1 port
322H	Hard disk drive #2 port

To access a particular drive, you must use one of the drive codes defined in Table 10-3 to identify it.

Table 10-3. Drive Identification Codes

CODE	DRIVE
00H	Floppy disk drive 1
01H	Floppy disk drive 2
02H	Floppy disk drive 3
03H	Floppy disk drive 4
80H	Hard disk drive 1
81H	Hard disk drive 2
82H	Hard disk drive 3
83H	Hard disk drive 4

Function Codes

To use the INT 13H instruction, place one of the function codes described in Table 10-4 in the AH register. Function codes 06H and higher are only for hard disk operation. A description of each function code appears on the following pages.

Table 10-4. Disk Drive Function Codes

CODE	FUNCTION
00H	Reset disk system. Each disk in the system will move its head(s) to track 0.
01H	Read disk status. Return the status in the AH register.
02H	Read specified sector(s) into a buffer.
03H	Write buffer contents to specified sector(s).
04H	Read a specified sector, test and verify that the specified sector(s) was read without error.
05H	Format the specified track.
06H	Flag the specified track as bad.
07H	Format the remainder of the specified drive starting at a specified track.
08H	Return current hard drive parameters.
09H	Initialize hard drive characteristics.
0AH	Read the specified sector(s) and ECC bytes.
0BH	Write the specified sector(s) and ECC bytes.
0CH	Seek to the specified track.

Table 10-4 (Continued). Disk Drive Function Codes

CODE	FUNCTION
0DH	Reset hard drive controller for the specified drive.
0EH	Read hard drive controller's sector buffer for the specified drive.
0FH	Write hard drive controller's sector buffer for the specified drive.
10H	Test the status of the specified drive.
11H	Recalibrate the specified drive. Move the head(s) to cylinder 0.
12H	Execute controller diagnostics on the memory (buffer) of the specified drive.
13H	Execute internal diagnostics on the specified drive and drive to controller interface.
14H	Execute the controller's self-tests for the specified drive.

Common Function Codes

Function Code 00H: Reset Disk System — This function code causes the interrupt to reset each floppy disk in the system. The reset sequence moves the read/write head(s) to track 0. This function is similar to function code 11H, which affects the hard drives. Before executing this interrupt, interrupt vector 1EH should point to the drive parameter block for the floppy disk drives. Table 10-5 describes the contents of each byte in the block. For more information, refer to Chapter 17.

Table 10-5. Drive Parameter Block

BYTE	PARAMETER
00H	This is the first disk controller command byte.
01H	This is the second disk controller command byte.
02H	This byte contains the number of timer ticks to wait before turning the drive motor off following any disk operation.
03H	This byte contains a value that corresponds to the number of data bytes in each sector: 0 = 128, 1 = 256, 2 = 512, and 3 = 1024.
04H	This byte contains the number for the last sector of a track.
05H	This byte contains the number of bytes that make up the gap length between sectors.

Table 10-5 (Continued). Drive Parameter Block

BYTE	PARAMETER
06H	This byte will contain 80H if the data length is 128 bytes per sector, or FFH if it is not.
07H	This byte contains the number of bytes to use for the gap length between sectors when formatting.
08H	This byte contains the data pattern to use during formatting.
09H	This byte contains the time required to settle the heads in milliseconds.
0AH	This byte contains the amount of time, in 1/4-second increments, required to start the motor of the drive.

Function Code 01H: Read Disk Status — This function code will cause the interrupt to return the disk status code in register AL. Use this call when an applications program needs to see the disk status generated by the last disk operation. Table 10-1 describes the status codes returned by this operation.

Function Code 02H: Read Sector(s) — This function code will cause the interrupt routine to read the specified number of sectors into the disk buffer. Refer to Table 10-6 for the parameters required for this operation.

Function Code 03H: Write Sector(s) — This function code will cause the interrupt routine to write the contents of the disk buffer into the specified number of sectors on the disk. Refer to Table 10-6 for the parameters required for this operation.

Function Code 04H: Verify Sector(s) — This function code will cause the interrupt routine to test the specified sector(s) to see if a data read can be accomplished without any errors. No transfer of data takes place during this procedure. Refer to Table 10-6 for the parameters required for this operation.

Function Code 05H: Format a Track — This function code will cause the interrupt routine to format the specified track. You must place the sector header information for the track in a block in memory. The buffer pointer, ES:BX, must point to the beginning of that memory block. Each sector on the track must have its own sector header. Each header must contain four bytes of information in the following order: track number, head (disk side) number, sector number, and bytes per sector (refer to function code

00H). Place the number of sectors per track in register AL before you execute the interrupt instruction. The data stored in the disk buffer will become the filler for each sector's data field.

If the drive is a hard disk, you must first load the controller's sector buffer with the data you wish to write to the disk using this function code.

Table 10-6. Required Parameters for Function Codes 02H - 05H

REGISTER	DESCRIPTION
AH	This register must contain the function code (02H - 05H).
DL	This register must contain the drive identification code described in Table 10-3.
DH	This register must contain the head (disk side) number; 0 or 1 for floppy disk drives, or 0 - 7 for hard disk drives.
CH	This register must contain the track number.
CL	This register must contain the sector number. For hard disk operation, the two most significant bits of register CL must contain the two most significant bits of the track number. AL This register must contain the number of sectors you want to transfer.
ES:BX	These two registers must contain the pointer to (address of) the disk buffer; the segment number in register ES and the offset address in register BX. The offset address is not required for function code 4 since no data is being transferred.

Hard Drive Function Codes

Use the following function codes only for hard disk drive operation. Table 10-7 describes the register requirements for function codes 06H, 07H, and 0CH.

Table 10-7. Register Requirements for Function Codes 06H, 07H, and 0CH

REGISTER	DESCRIPTION
AH	This register must contain the function code (06H, 07H, or 0CH).
AL	This register must contain the interleave factor (01H - 10H) (not required for function code 0CH).

Table 10-7 (Continued). Register Requirements for Function Codes 06H, 07H, and 0CH

REGISTER	DESCRIPTION
CH	This register must contain the cylinder number (00H - 03FFH).
CL	This register must contain the sector number to read. The two most significant bits of register CL must contain the two most significant bits of the track number.
DH	This register must contain the head number (00H - 07H).
DL	This register must contain the drive number (80H - 87H).

Function Code 06H: Flag Bad Track -- This function code will cause the interrupt routine to flag the track, specified by the head number recorded in register DH and cylinder number recorded in register CH, as a bad track.

Function Code 07H: Format Drive Starting at Specified Track — This function code will cause the interrupt routine to format the remainder of the drive, starting at the track specified by the head number recorded in register DH and the cylinder number recorded in register CH. The data in the disk buffer will become the filler in the data field of each sector.

Function Code 08H: Return Current Drive Parameters — When completed, register DL will contain the number of active hard disk drives. Register DH will contain the maximum number of usable heads and register CH will contain the maximum number of usable cylinders. Register CL will contain the maximum usable sector number with the two most significant bits representing the two most significant bits of the maximum usable cylinder number.

Function Code 09H: Initialize Drive Type Characteristics — Interrupt vector 41H should point to the drive parameter table. The table is organized as four sets (corresponding to the four possible drives) of information formatted as described in Table 10-8.

Table 10-8. Drive Type Characteristics

DATA TYPE	DESCRIPTION
Word	This word must contain the number of tracks on the drive.

Table 10-8 (Continued). Drive Type Characteristics

DATA TYPE	DESCRIPTION
Byte	This byte must contain the number of heads on the drive.
Word	This word must contain the number of the start cylinder for reduced write current. Toward the hub of a hard disk, where the circumference of each cylinder is smaller, some disk drives require the application of less electrical current to the write head to prevent the magnetic pulses from overlapping into adjacent tracks.
Word	This word must contain the number of the start cylinder for write precompensation. Disk storage devices operate by creating magnetically polarized areas on the disk surface that correspond to the presence or absence of bits in the information being stored. It is possible, if the recorded data is too closely spaced, for the charged areas on the disk surface, representing data, to shift slightly. If such a shift occurs, reduced reliability results. Precompensation refers to the intentional shifting of data during the recording process so the combined result of precompensation and the natural shift (which we cannot prevent) places the data in the desired position. We do not require prcompensation on the outer tracks of the disk since the spacing between bits is great enough that no significant shifting occurs. Since the inner tracks have a smaller circumference, the bits are closer together requiring precompensation to maintain acceptable reliability.
Byte	This byte must contain the maximum number of bits to use to correct ECC errors.
Byte	This bit-mapped byte must contain the several controller parameters described in Table 10-9.
Byte	The byte must contain the timeout value for read or write operations.
Byte	This byte must contain the timeout value for format operations.
Byte	This byte must contain the timeout value for disk diagnostic operations.
Byte	Reserved (four bytes).

Table 10-9. Controller Parameters

BIT	DESCRIPTION
7	When set (1), disable retries on disk operations.
6	When set (1), disable ECC operation.
3 - 5	Not used.
0 - 2	Determines the step rate to be used by the disk drive: 000 = 3 microseconds 100 = 200 microseconds 101 = about 60 microseconds 110 = 3 milliseconds 111 = 3 milliseconds

Function Code 0AH: Read Sectors and ECC Bytes — This function code will cause the interrupt routine to read the specified sectors into memory including the error correction code bytes. Refer to Table 10-10 for the parameters required for this operation.

Function Code 0BH: Write Sectors and ECC Bytes — This function code will cause the interrupt routine to write memory into the specified sectors including the error correction code bytes. Refer to Table 10-10 for the parameters required for this operation.

Table 10-10. Register Requirements for Function Codes 0AH and 0BH

REGISTER	CONTENTS
AH	This register must contain the function code (0AH or 0BH).
AL	The register must contain the number of sectors involved in the read or write operation (01H - 4FH).
ES:BX	These registers must contain the memory address of the disk buffer.
CH	This register must contain the cylinder number (00H - 3FFH).
CL	This register must contain the starting sector number of the operation. Place the two most significant bits of the cylinder number in the two most significant bits of the CL register.

Table 10-10 (Continued). Register Requirements for Function Codes 0AH and 0BH

REGISTER	CONTENTS
DH	This register must contain the head number (00H - 07H.
DL	This register must contain the hard drive number (80H - 87H).

Function Code 0CH: Seek a Track — This function code will cause the interrupt routine to seek (move the heads) to the specified track. Except for register AL, which is empty or may contain any value, refer to Table 10-7 for the CPU register requirements.

Function Code 0DH: Reset Controller — This function code will cause the interrupt routine to reset the hard drive's controller values to the default values for the specified drive. Register DL must contain the drive number, a value from 80H to 87H.

Function Code 0EH: Read Sector Buffer — This function code will cause the interrupt routine to read the specified hard drive's controller sector buffer into memory. You must set the AL register to a value of 1. The ES and BX registers must contain the starting transfer address in memory for the sector buffer data. The DL register must contain the drive number, a value from 80H to 87H.

Function Code 0FH: Write Sector Buffer — This function code will cause the interrupt routine to write the contents of memory into the specified hard drive's controller's sector buffer. You must set the AL register to a value of 1. The ES and BX registers must contain the starting transfer address in memory for the data. The DL register must contain the drive number, a value from 80H to 87H.

Function Code 10H: Test Drive Ready — This function code will cause the interrupt routine to check the status of the specified drive. Register DL must contain the drive number being tested, a value from 80H to 87H. The AH register will contain the resulting status. If the drive is ready, the status will be 00H.

Function Code 11H: Recalibrate Drive — This function code will cause the interrupt routine to recalibrate the specified drive (move the read/write heads to cylinder 0). Register DL must contain the drive number, a value from 80H to 87H.

Function Code 12H: Execute Controller Memory Diagnostic — This function code will cause the interrupt routine to run a data-pattern test on the controller's internal memory buffer for the drive specified in the DL register.

Function Code 13H: Execute Internal Diagnostics — This function code will cause the interrupt routine to run a test of the specified drive and drive-to-controller interface. These tests consist of a recalibrate command, followed by a series of seek commands. The routine tests sector 0 of head 0 to see if it can read data successfully. These tests are not destructive;they do not execute write commands. Register DL must contain the specified drive number, a value from 80H to 87H.

Function Code 14H: Controller Self-Test — This function code will cause the interrupt routine to run the controller's self tests for the specified drive. These tests include a check of the controller's processor, the data buffer, ECC circuits, and the checksum of the ROM. Register DL must contain the specified drive number, a value from 80H to 87H.

Error Status Codes

All of the hard drive function calls will return a status code in register AH upon completion of the routine. Register AH and the carry flag CF will contain a 0 if the function was successful. If the function failed, the carry flag will set (1) and register AH will contain a value representing a specific error. Table 10-11 defines the error codes.

Table 10-11. Disk Drive Error Codes

VALUE	DESCRIPTION
00H	This code indicates no error has occurred.
01H	This code indicates issuance of a bad (invalid) command.
02H	This code indicates a hard disk controller could not find an address mark on the specified disk.
03H	This code indicates an attempted write (or format) operation on a write-protected disk.
04H	This code indicates the specified record (or sector) could not be found.

Table 10-11 (Continued). Disk Drive Error Codes

VALUE	DESCRIPTION
05H	This code indicates a hard disk controller reset failed.
07H	This code indicates the controller would not accept drive parameters.
08H	This code indicates a DMA overflow took place. This occurs when the DMA controller cannot keep up with the disk data transfer rate and loses information. It usually results from excessive DMA action from other devices on the data bus or else a hardware failure occurred within the bus interface.
09H	This code indicates a DMA boundary error has occurred. The hardware is incapable of transferring sector buffer information across 64K memory boundaries. You can avoid this error in one of two ways. Try reducing the sector count in the AL register or changing the buffer pointer so that the entire transfer area resides within a single 64K memory segment.
10H	This code indicates a bad CRC on a disk read operation has occurred. The routine located the specified record, but the cyclic redundancy check (CRC) for the data did not match the value calculated by the controller. On hard disk drives, this error flags any error that the ECC circuitry could not correct.
11H	On hard disk drives this code indicates an ECC error has occurred, but the controller was unable to reconstruct the lost data.
20H	This code indicates the disk controller IC has failed.
40H	This code indicates the disk controller attempted to move the read/write head to a specified track, but could not find a matching sector header on that track (bad seek).
80H	This code indicates a timeout error. The routine issued a command to the controller, but it was not completed within a specified amount of time by the hardware.

Table 10-11 (Continued). Disk Drive Error Codes

VALUE	DESCRIPTION
BBH	This code indicates an undefined error has occurred. Usually, this is the result of a bad controller IC.
FFH	On hard disk drives, this code indicates a sense drive status operation has failed.

Booting an Operating System (INT 19H)

INT 19H attempts to boot an operating system from the specified disk drive. The DL register must contain the drive number of the intended boot drive.If it is the hard disk drive, AL must contain a partition number between 0 and 3 expressed in ASCII format (that is, a value from 30H to 33H). If the value is 0, the boot track of the default partition will be selected.

The routine attempts to read from track 0, sector 1 of the specified device. The code loacted at this position is loaded into memory and executed. If the boot routine fails because the drive did not exist or because of a hardware failure, an error message will appear and control passes to the Monitor program.

Disk Parameters (INT 1EH)

INT 1EH points to an area in memory that contains the disk parameters for the system. When you turn the computer on this area of memory initializes and provides support for loading the operating system from the disk. Table 10-12 describes the data stored in this area of memory.

NOTE: You may need to modify the parameter tables if you use different disk drives than those supplied by Zenith Data Systems.

Table 10-12. Disk Parameter Table

BYTE	DESCRIPTION
1	Bits 0 - 3 contain the head unload time (the range is 16 - 240 milliseconds in 16-millisecond increments). Bits 4 - 7 contain the head step rate for the drive (the range is 1 - 16 milliseconds in 1-millisecond increments, in reverse order — 0FH = 1 millisecond, 0EH = 2 milliseconds, and so on).

Table 10-12 (Continued). Disk Parameter Table

BYTE	DESCRIPTION
2	Bits 1 - 7 contain the head load time (the range is 2 - 254 milliseconds in 2-millisecond increments). Bit 0 contains the DMA flag (if bit 0 is clear (0), the DMA mode in use).
3	This is the motor on timeout value (the amount of time the motor will remain on after the last disk operation, measured in timer ticks).
4	This contains the number of data bytes per sector. Refer to Table 11-13.
5	This contains the number of sectors per track.
6	This contains the gap length of gap 3 for read and write operations. The value in byte 4 determines the value of this byte. Refer to Table 10-13.

Table 10-12 (Continued). Disk Parameter Table

BYTE	DESCRIPTION
7	If byte 4 (sector length) is 0, then this value is the length of the data read from or written to each sector. If byte 4 is not 0, set this value to FFH; the controller will ignore it.
8	This contains the gap length of gap 3 for format operations (the value in byte 4 determines this value). Refer to Table 10-13.
9	This contains the format data pattern (the bit pattern used by the format command to fill and test the sector).
10	This contains the head settle time, measured in 1-millisecond increments.
11	This contains the motor start time, measured in 125-millisecond increments.

Table 10-13. Sector and Gap Lengths

BYTE 4 VALUE	BYTES PER SECTOR	SECTORS PER TRACK	WRITE GAP LENGTH	FORMAT GAP LENGTH
FM Mode				
00H	128	12H	07H	09H
00H	128	10H	10H	19H
01H	256	08H	18H	30H
02H	512	04H	46H	87H
03H	1024	02H	C8H	FFH
04H	2048	01H	C8H	FFH
MFM Mode				
01H	256	12H	0AH	0CH
01H	256	10H	20H	32H
02H	512	08H	2AH	50H
03H	1024	04H	80H	F0H
04H	2048	02H	C8H	FFH
05H	4096	01H	C8H	FFH

NOTE: MS-DOS version 3 uses MFM Mode, 512 bytes per sector, and nine sectors per track. The UCSD p-system can use MFM mode, 512 bytes per sector, and ten sectors per track. Neither of these formats is represented in this table.

Chapter 11

Video Interrupts

This chapter describes the video interrupts and how to define characters.

Programming the Video Interrupts

Although the Portable Computer drives an LCD display, the video circuits support the color graphics capability of PC-compatible computers by using a VLSI chip, the V6355. This integrated circuit not only emulates the 6845 CRT controller's capabilities, but also support the LCD display in a unique manner. The sixteen colors of the PC-compatible display are selectively sent on the RGB video connector or displayed on the LCD as a 16-level gray scale. Table 11-1 contains descriptions of the normal video interrupts.

Refer to Chapter 6 for information on how to use and program these interrupts and to Chapter 15 for detailed information on the V6355 video controller.

Table 11-1. Video Interrupts

INTERRUPT	FUNCTION
10H	Video input/output
1DH	Video initialization
1FH	Defining characters

Video Input/Output (INT 10H)

INT 10H provides communications with the video logic section of the computer. To use this interrupt, place one of the function codes described in Table 11-2 in register AH. The following pages contain descriptions of each of the function codes.

Function Code 00H: Set Video Mode — This function code will cause the interupt to set the video mode according to the value placed in register AL and described in Table 11-3.

Table 11-2. Video Input/Output Function Codes

CODE	FUNCTION
00H	Set video mode
01H	Set cursor type
02H	Set cursor position
03H	Read cursor position
04H	Read light pen position — not used
05H	Select active display page
06H	Scroll an area of screen up
07H	Scroll an area of screen down
08H	Read cursor position
09H	Send character/attribute to screen
0AH	Send character to screen
0BH	Set graphics foreground color
0CH	Write graphics pixel
0DH	Read graphics pixel
0EH	Dumb terminal display]
0FH	Return video state
64H	Set scrolling mode

Table 11-3. Video Modes

MODE	DESCRIPTION
0	40 characters by 25 rows, monochrome text display on the LCD screen or RGB output.
1	40 characters by 25 rows, color text display at the RGB output or as a gray-scale display on the LCD screen.
2	80 characters by 25 rows, monochrome text display on the LCD screen or RGB output. Individual video pages may be scrolled without affecting other video pages.
3	80 characters by 25 rows, color text display at the RGB output or as a gray-scale display on the LCD screen. Individual text pages may be scrolled.

Table 11-3 (Continued). Video Modes

MODE	DESCRIPTION
4	320 x 200 pixel resolution, color graphics display at the RGB output or as a gray-scale display on the LCD screen. Text displays appear as 40 characters by 25 rows.
5	320 x 200 pixel resolution, monochrome graphics display on the LCD screen or RGB output. Text displays appear as 40 characters by 25 rows.
6	640 x 200 pixel resolution, monochrome graphics display on the LCD screen or RGB output. All three scrolling modes are available. Text displays appear as 80 characters by 25 rows.
7	80 characters by 25 rows, monochrome text display. This mode requires a TTL-compatible monochrome graphics adapter, which you cannot implement in the Portable Computer. Instead, this computer enters a standby state and disables the video display to preserve battery power.

Function Code 01H: Set Cursor Size — This function code causes the interrupt to set the cursor size. Load bits 0 - 4 of the CH register with the starting scan line number of the cursor and bits 0 - 4 of the CL register with the ending scan line number. While it is possible to load any value between 0 and 31, characters in the Portable computer contain only eight scan lines; use only values 0 - 7 for the starting and ending scan line.

NOTE: The starting scan value may not be greater then the ending scan value, or no cursor will appear.

Function Code 02H: Set Cursor Position — This function code causes the interrupt to place the cursor at the specified row and column location on the screen. Load register DH with the cursor row number and register DL with the column number. The row numbers extend from 0 to 24. The column numbers extend from 0 to 79 (80 characters per line) or 39 (40 characters per line). Therefore, when you load a value of 0 in the DH and DL registers, the cursor appears in the upper left-hand corner of the screen. Load register BH with the video page number; this must be consistent with the video mode selected (only page 0 is valid when you are in the graphics mode).

Function Code 03H: Read Cursor Position — This function code causes the interrupt to return the current cursor information. The row number will be in register DH, the column number will be in register DL. The cursor's starting scan line number will be in register CH, and the cursor's ending scan line number will be in register CL. Before executing the interrupt, register BH must contain the page number.

Function Code 04H: Read Light Pen Position — The Portable Computer does not support a light pen option, even though this is a valid function code for PC-compatible computers.

The function code causes the interrupt to attempt to obtain the light pen's position. After the interrupt has been executed, register AH will contain the light pen trigger/switch status (0 or 1). A value of 1 in the AH register indicates an active switch closure (the light pen is on). If the register contains a 0 instead, the switch is not activated (the light pen is off). If the AH register contains a 1, then a number of other registers contain related position information. The DH register will contain the tow number, register DL will contain the column number. The CH register will contain the column number. The CH register will contain the scan line row number (0 - 199), and register BX will contain the pixel column number. The pixel column number can vary between 0 - 310 or 0 - 639, depending upon the graphics mode.

Function Code 05H: Select Active Display Page — This function code causes the interrupt to display the page specified in register AL. In the text modes, unused portions of video memory can serve as additional video pages. In video modes 0 and 1, the valid page numbers are 0 to 7; in video modes 2 and 3, they are 0 - 3. In the graphics modes, only page 0 is valid.

Function Code 06H: Scroll an Area of the Screen Up — This function code causes the interrupt to scroll the specified area of the screen up a specified number of lines. Prior to executing this interrupt, you must load certain registers with specific data. The CX register must contain the upper left-hand coordinates (place the row number in register CH and the column number in register CL) of the scroll area. The DX register must contain the lower right-hand coordinates (place the row number in register DH and the column number in register DL) of the same area. The

BH register must contain the attribute byte for the blank lines, and the AL register contains the number of lines to scroll. If the AL register contains a 0, the entire window will be cleared.

NOTE; If a hardware or smooth scrolling mode is currently active when this interrupt executes, they will function only if the entire screen is scrolled. Hardware and software scrolling operations affect the entire screen and not just a portion of the screen.

Function Code 07H: Scroll an Area of the Screen Down — This function code causes the interrupt to scroll the specified area of the screen down the specified number of lines. Before executing this interrupt, you must load the following register information. The CX register must contain the upper left-hand coordinates (place the row number in register CH and the column number in register CL) of the scroll area. The DX register must contain the lower right-hand coordinates (place the row number in register DH and the column number in register DL) for the same area. The BH register must contain the attribute byte for the blank lines, and the AL register contains the number of lines to scroll. If the AL register contains a 0, the entire window will be cleared.

Function Code 08H: Read Character and Attribute — This function code causes the interrupt to return the character and attribute codes for the character that resides at the current cursor position. In text modes, register BH must contain the video page number. Upon completion of the routine, the character code will be in register AL and the attribute code will be in register AH.

Function Code 09H: Write Character and Attribute to Screen — This function code causes the interrupt to write the specified character and attribute codes to the cursor location. Place the character code in register AL, and the attribute code in the BL register. The CX register contains the number of times the character is to repeat, and in text modes, the page number is in the BX register.

Function Code 0AH: Write Character to Screen — This function code causes the interrupt to write the specified character to the screen, but not the attribute byte. Place the character code in register AL, the number of times to repeat the character in register CX, and in text modes, the page number in register BH.

Function Code 0BH: Set Color Palette — This function code is only available in mode 4, the 320 x 200 graphics mode. Place a value from 0 to 127 in register BH (see the following text), and a value from 0 to 4 in register BL.

If the value placed in register BH is even, then the current background color will become the foreground color (normally, 0 - 31). Values above 15 will select the intensified level of the 16 colors.

If the value placed in register BH is odd, the value in the BL register will determine which one of two available palettes is chosen. The palette and pixel color number will determine the foreground color. A value of 0 will select palette 0, likewise a value of 1 will select palette 1. Refer to Table 11-4 for the palette number and color number matrix.

Table 11-4. Palette and Pixel Colors

COLOR NUMBER	PALETTE 0	PALETTE 1
1	Green	Cyan
2	Red	Magenta
3	Yellow	White

NOTE: The V6355 in this computer can modify the actual color display. Refer to chapter 14 for more information.

Function Code 0CH: Write Graphics Pixel — This function code causes the interrupt to light a single pixel at the specified location on the screen. Place the pixel row number in register DX, the pixel column number in register CX, and the color in the AL register. Row numbering extends from 0 - 19 and column numbering from 0 -319 or 0 - 639. Color values range from 0 - 3 in medium-resolution (320 x 200) mode or 0 - 1 in high-resolution (640 x 200). In all cases, 0 is the background color (usually black). In the high-resolution mode, 1 is the foreground color. In the medium-resolution mode, the palette and the color number determine the color, as described in Table 11-4.

If the most significant bit (bit 7) of AL is set, the color is XORed with the current color permitting simple animation. For more information on animation techniques, see the discussion on graphics in the GW-BASIC manual.

Function Code 0DH: Read Graphics Pixel — This function code returns the color of the pixel at the specified location. Place the pixel row number in register DX, and the pixel column number in register CX. The returned color value will appear in the AL register.

Function Code 0EH: Dumb Terminal Display — This function code causes the interrupt to treat the character as if it were sent to a dumb terminal. The routine will treat the back space (08H), carriage return (DH), line feed (0AH), and bell (07H) as console commands rather than screen formatting characters. If a character should print at the end of a screen line, the cursor will position at the start of the next line. If you perform a line feed on the last display line of the screen or if a character prints at the last position of the last line, the screen will scroll up one line. When scrolling, the attribute for the new row (when in text mode) will be the same as the attribute of the character at the cursor position on the line when the scrolling takes place.

Place the character in register AL, the foreground color in register BL (for graphics modes), and the display page number in register BH.

Function Code 0FH: Return Video State — This function code returns the current video state. Register AL will contain the current video mode (refer to Function Code 00H). The AH register will contain the screen width in columns, and register BH will contain the active video page number.

Function Code 64H: Set Scrolling Mode — This function code causes the interrupt to select one of three scrolling modes. Place the scrolling mode value in register AL: mode 0 is software scrolling, mode 1 is hardware jump scrolling, and mode 2 is hardware smooth scrolling. Keep in mind the following limitations.

- Hardware scrolling will not work in the 40 x 25 text modes (0 and 1).
- Hardware smooth scrolling works only in the high-resolution graphics mode (6).
- Hardware jump scrolling works only in the graphics modes (4, 5, and 6) and 80 x 25 text mode (3).
- Software scrolling will work in all modes. If you write software that bypasses the Monitor program, use software scrolling.

Video Initialization (1DH)

INT 1DH, unless otherwise programmed by an application program, initialized the video section parameters according to the information stored in the Monitor program ROM. This is the same data used to initialize the video section of the computer when you first turn the system on.

Four tables are necessary to properly define the video parameters. Keep in mind the fact that the Portable Computer does not use video mode 7, the monochrome text (TTL) video mode. Table 11-5 defines the values for each register in the 6845 register set. Refer to Chapter 14 for a discussion of the 6845 register set in the V6355 video controller.

Table 11-5. Video Initialization Default Values

REGISTER NUMBER	TEXT 40x25	TEXT 80x25	GRAPHICS	MONOCHROME TEXT (TTL)
R0	38	71	38	61
R1	28	50	28	50
R2	2D	5A	2D	52
R3	0A	0A	0A	0F
R4	1F	1F	7F	19
R5	06	06	06	06
R6	19	19	64	19
R7	1C	1C	70	19
R8	02	02	02	02
R9	07	07	01	0D
R10	06	06	06	0B
R11	07	07	07	0C
R12	00	00	00	xx
R13	00	00	00	xx
R14	xx	xx	xx	xx
R15	xx	xx	xx	xx
R16	xx	xx	xx	xx
R17	xx	xx	xx	xx

NOTE: All values are expressed in hexadecimal. xx represents any value between 00H and FFH.

Defining Graphics Characters (1FH)

INT 1FH functions as a pointer to an extended graphics character set. It is not a true interrupt, since it points to the start of a user-defined character set. A true interrupt points to the beginning of the interrupt service routine. If you define an extended character set, INT 10F will use it to display a medium- or high-resolution graphics character.

Normally, the character generator ROM defines the first 128 characters (00H - 7FH) used in the graphics modes. You can create a custom character set of 128 additional characters (80H - FFH) using the following procedure:

1. Assign a 1K section of memory (not video memory) to hold the character set. You will need eight bytes for each character you create.

2. Define each character in an 8 x 8 matrix as illustrated by the example in Figure 11-1. Be sure to allow for ascenders, descenders, and space between characters.

3. Identify the pixels you want lit in each line of the matrix.

4. Refer to the example in Figure 11-1. Using the first pixel as the most-significant bit, assign binary-based hexadecimal weights to each column in the matrix. Refer to the hexadecimal values across the top of the matrix.

5. For each row in the matrix, add the values of the lit pixels to produce a hexadecimal value.

6. Load the resulting eight values into the first eight bytes of the memory you assigned for the character set.

7. Repeat this procedure for each of the 128 characters in the set.

8. Set the pointer for interrupt 1FH to the start of the 1K memory block. Interrupt 1FH's pointer is at memory location 0000:007C.

Now, whenever you use INT 10H to display a character code between 80 and FF in graphics mode, the computer will use the corresponding character from your newly created set.

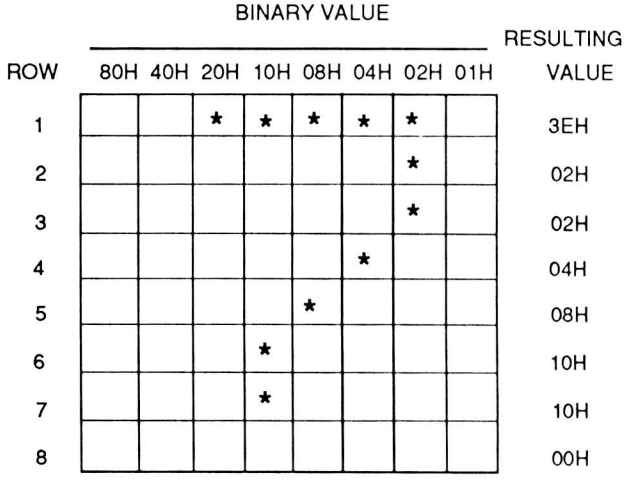


Figure 11-1. Character Design Matrix

Part III

Hardware

Chapter 12

Introduction to the Hardware

This section of the manual provides an overview of the major sections that make up the Portable Computer. This chapter describes the basic circuit groups, the external connectors pinouts, and the power system.

Related Hardware Publications

Intel iAPX 88 Book — This manual, published by Intel, provides information on the architecture and programming of the 80C88 CPU.

Intel iAPX 86/88, 186/188 User's Manual — This manual, published by Intel, provides programmer's reference material for the 80C88 and 8087 processors used in this computer. It also briefly discusses the architecture of each integrated circuit.

8087 Applications and Programming for the IBM PC and Other PCs — This manual, published by Robert J. Brady Co., provides application programming information for the 8087 numeric data coprocessor used in this computer.

MS-DOS Programmer's Utility Pack — This software package, produced by Zenith Data Systems, contains information and software to write and assemble MS-DOS and 8088 assembly language programs. Separate packages are available for MS-DOS version 2 and MS-DOS version 3.

Major Circuits

The computer contains four major sections as shown in Figure 12-1 (control, processor, memory, and peripheral support) and the power supply. Each of these sections contain a number of major integrated circuits.

The control section contains three major integrated circuits that control the operation of the computer. These integrated circuits are the ROM, the gate array, and the decoder. The ROM contains the Monitor program which provides a number of capabilities discussed in a later chapter. The gate array contains a large portion of the hardware logic for the computer. The decoder is a programmed logic array that provides most of the chip select signals for the rest of the system.

The processor section consists of the CPU (80C88) and an optional numeric processor extension (8087-2), and the support integrated circuits. The CPU is the heart of the computer, although it can't function without the control section (the ROM) to guide its first few steps. Between the CPU and numeric data coprocessor, all the calculations and sophisticated processing takes place. The support ICs take the signals generated by the CPU and coprocessor and cause other sections of the computer to function. Some of these devices, such as the interrupt controller, act analogous to a secretary and feed the CPU only the most important messages, which causes it to interrupt its normal operation to expedite a more important process.

The memory section contains most of the read/write memory in the computer. This memory is sometimes called user memory, although it is more commonly called RAM. There are two distinct sections of memory: scratchpad memory, used by the CPU for intermediate calculations and for temporary storage space (buffers); and contiguous memory, providing a total of 640K of available user memory.

The peripheral support section consists of the video controller and memory, the floppy disk controller, and the input/output sections. These include the keyboard, the internal modem, the parallel printer port, and the serial input/output port.

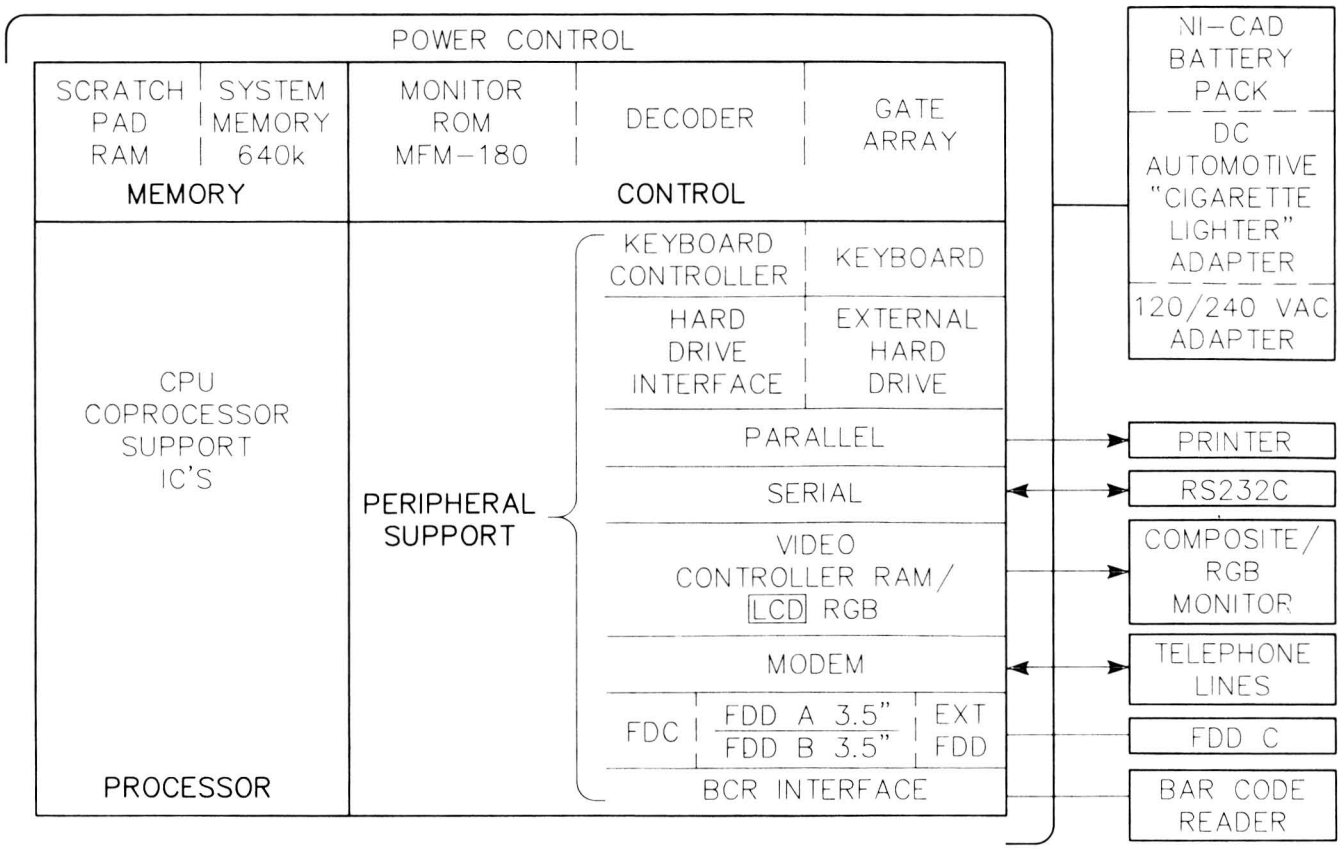


Figure 12-1. Major Circuit Groups

External Cable Connections

The following sections describe each external connector and provide a description of the pins used.

Back Panel of the Computer

The external cable connectors are on the back of the computer. Refer to Figure 12-2 for their locations.

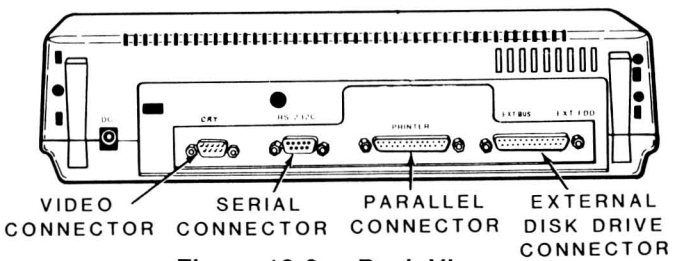


Figure 12-2. Back View

Video Connector — This connector supplies RGB video output for an external color video monitor and a composite monochrome signal for a monochrome video monitor. Refer to Figure 12-3 and Table 12-1 for the pinout.

Serial Connector (COM1) — This connector is an RS-232C DTE input/output port for use with a serial printer or peripheral at a software programmable baud rate. The operating system contains the necessary software to properly configure the connector for use with most serial devices. Refer to Figure 12-4 and Table 12-2 for the pinout.

Parallel Connector (LPT1) — This connector provides Centronics-type output signals for a parallel printer or peripheral. The operating system contains the necessary software to configure this connector for use with most parallel devices. Refer to Figure 12-5 and Table 12-3 for the pinout.

External Disk Drive Connector — This connector provides the signals necessary to connect an external disk drive to the computer. Refer to Figure 12-6 and Table 12-4 for the pinout.

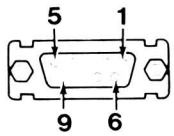


Figure 12-3. Video Connector

Table 12-1. Video Connector

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1 - 2	GND	Ground.
3	R	Red. This line carries the red video signal.
4	G	Green. This line carries the green video signal.
5	B	Blue. This line carries the blue video signal.
6	I	Intensity. This line carries the intensity signal.
7	C	Composite. This line carries the composite monochrome video signal.
8	HSYNC	Horizontal sync. This line carries the horizontal sync signal for RGB monitors.
9	VSYNC	Vertical sync. This line carries the vertical sync signal for RGB monitors.

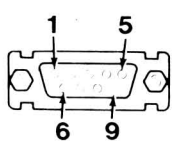


Figure 12-4. Serial Connector

Table 12-2. Serial Connector

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	CD	Carrier detect. This modem generated signal indicates the detection of a carrier signal.
2	DO	Data out. This signal is the transmitted serial data.

Table 12-2 (Continued). Serial Connector

PIN NUMBER	SIGNAL NAME	DESCRIPTION
3	DI	Data in. This signal is the received serial data.
4	DTR	Data terminal ready. This signal indicates to the remote hardware that the computer is ready to transmit data. Hardware handshaking uses this signal.
5	SG	Signal Ground.
6	DSR	Data set ready. This signal indicates that the remote equipment has a data block ready to transmit. Hardware handshaking uses this signal.
7	RTS	Request to send. This signal requests permission to transmit data. Hardware handshaking uses this signal.
8	CTS	Clear to send. This signal indicates that the remote equipment is ready to receive data. Hardware handshaking uses this signal.
9	RD	Ring detect. This modem generated signal indicates the detection of an incoming call.

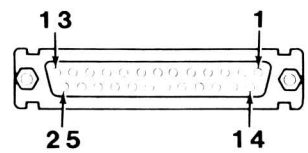


Figure 12-5. Parallel Connector

Table 12-3. Parallel Connector

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	STROBE	Strobe bit. This active low pulse indicates that the computer is transmitting parallel data. This signal times the the data being sent to the peripheral device.
2	PDATA0	Data bit 0. This signal is the buffered and latched data bit 0.
3	PDATA1	Data bit 1. This signal is the buffered and latched data bit 1.

Table 12-3 (Continued). Parallel Connector

PIN NUMBER	SIGNAL NAME	DESCRIPTION
4	PDATA2	Data bit 2. This signal is the buff
5	PDATA3	Data bit 3. This signal is the buffered and latched data bit 3.
6	PDATA4	Data bit 4. This signal is the buffered and latched data bit 4.
7	PDATA5	Data bit 5. This signal is the buffered and latched data bit 5.
8	PDATA6	Data bit 6. This signal is the buffered and latched data bit 6.
9	PDATA7	Data bit 7. This signal is the buffered and latched data bit 7.
10	ACK	Acknowledge. This active low signal indicates that the peripheral device has received the data. It can be used for hardware handshaking.
11	BUSY	Busy. This signal indicates that the peripheral device is busy and not ready to receive data. It can be used for hardware handshaking.
12	PE	Paper end (out). This signal indicates a peripheral fault. This signal indicates that the printer is out of paper.
13	SLCT	Select. This signal is active when a peripheral device has been selected.
14	AUTO FDXT	Automatic feed. This active low signal requests a paper feed by the peripheral.
15	ERROR	Printer fault. This active low signal indicates that an error condition exists in the peripheral.
16	INIT	Initialize. This active low signal initializes the peripheral.
17	SLCT IN	Select in. This active low signal is used by the peripheral to select the computer.
18 - 25	GND	Ground.

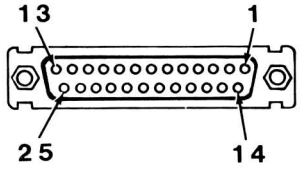


Figure 12-6. External Disk Drive Connector

Table 12-4. External Disk Drive Connector

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	$\overline{\text{WRITE GATE}}$	Write gate. This active low signal enables writing to the disk and is used to prevent accidentally writing to a disk during the power-up or power-down sequence.
2	$\overline{\text{DS3}}$	Drive select 3. This active low signal selects disk drive D.
3	$\overline{\text{DS2}}$	Drive select 2. This active low signal selects disk drive C.
4	$\overline{\text{WRDATA}}$	Write data. This active low signal is the serial data that is to be written to the disk.
5	$\overline{\text{DIR}}$	Direction. This active low signal determines the direction the head moves when it is stepped.
6	$\overline{\text{STEP}}$	Step. This active low signal is pulsed each time the head is to move from one track to another.
7-13	$\overline{\text{GND}}$	Ground.
14	$\overline{\text{READY}}$	Ready. This active low signal informs the system that the disk drive is ready for a command.
15	$\overline{\text{SIDE1}}$	Side 1 select. This active low signal determines the side (or head) that is selected for reading or writing data.
16	$\overline{\text{READ DATA}}$	Read data. This active low signal is the serial data that has been read from the disk.
17	$\overline{\text{WRITE PROTECT}}$	Write protect. This active low signal informs the system that the disk in the selected drive is write-protected.
18	$\overline{\text{TRACK0}}$	Track 0. This active low signal informs the system that the heads in the selected disk drive are at track 0.
19	$\overline{\text{INDEX}}$	Index. This active low signal informs the system that the selected disk drive has detected the timing hole (index mark) on the disk.
20	$\overline{\text{MOTOR}}$	Motor on. This active low signal turns on the motors in the external disk drive system.
21-24	—	Not connected.
25	$\overline{\text{EXT ON}}$	External on. This active low signal informs the computer that the external disk drive system is turned on.

You will find two connectors on the left side of the computer as shown in Figure 12-7. A brief description of each follows.

Line Connector — This connector supplies the necessary signals for telephone computer-to-computer communication. It connects the internal modem to the telephone company line. Refer to Figure 12-8 and Table 12-5 for the pinout.

Telephone Connector — This connector supplies the necessary signals for a standard telephone when the telephone company line is connected to the line connector. Refer to Figure 12-8 and Table 12-5 for the pinout.

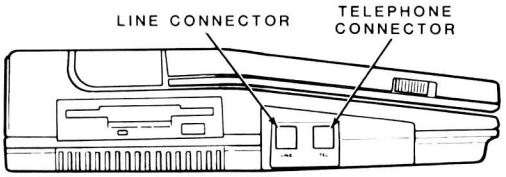


Figure 12-7. Left Side View

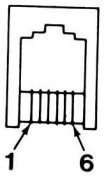


Figure 12-8. Line and Telephone Connectors

Table 12-5. Line and Telephone Connector

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1 - 2	—	Not connected.
3	RING	Ring. This line carries the ring signal.
4	TIP	Tip. This line carries the voice or data signal.
5 - 6	—	Not connected.

External Keypad Connector — This connector is located on the right side of the computer, as shown in Figure 12-9. The external keypad connector supplies the necessary power and signal lines for an external numeric keypad. Refer to Figure 12-10 and Table 12-6 for the pinout.

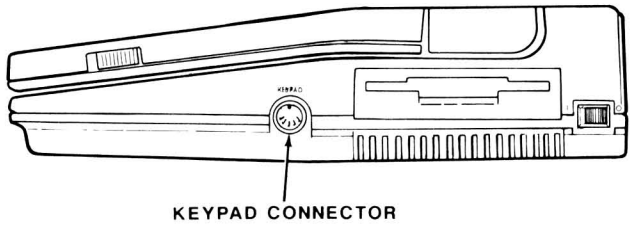


Figure 12-9. Right Side View

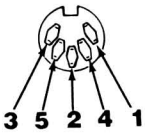


Figure 12-10. External Keypad Connector

Table 12-6. External Keypad Connector

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	KYBCLK	Keypad clock
2	KYBDATA	Keypad data
3	KYBRST	Keypad reset
4	KYBGND	Ground
5	KYBPWR	+5 volts DC

Configuration Switch — The configuration DIP switch is behind a removable access cover. Chapter 2 contains a description of this switch. Refer to Figure 12-11.

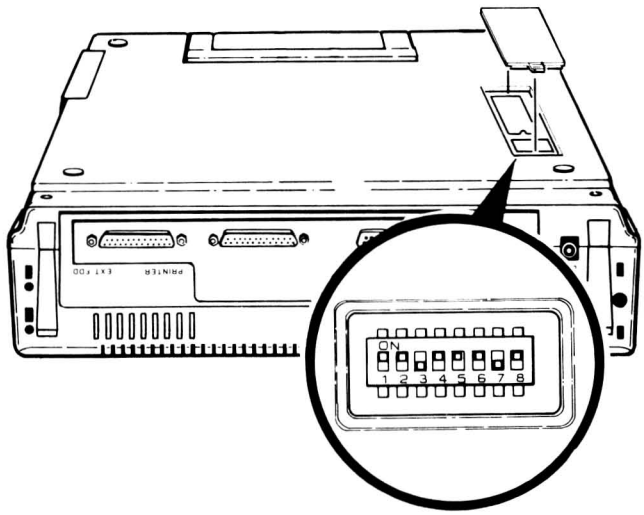


Figure 12-11. Bottom of the Computer

Power System

The DC power supply provides the voltages used by the various circuits in the computer. Power for the supply comes from an external adapter (120-volt AC line, 240-volt AC line, or 12-volt DC cigarette lighter) or an external battery pack.

External AC or DC Voltage Source — The AC adapter provides approximately +18 volts of regulated DC to the system. The DC cigarette lighter adapter cord supplies unchanged DC power from an automotive-type 12-volt system.

Low voltage detection — The output voltage from the external battery controls the input to the low voltage warning circuit. As long as the DC adapter is supplying a charge potential to the battery, the voltage level will remain relatively high, even with a discharged battery.

When the output voltage of the battery drops below a preselected level the voltage detector will trigger the low power warning circuit on the main board, causing the low-power indicator to flash.

+/–5 VDC regulated supply — The computer requires a regulated +5 VDC to operate. In addition, RS-232C operation requires a regulated –5 VDC. These supplies are developed from the DC input voltage. These voltages can vary from slightly under +12 VDC with the system running from the battery to as much as +18 VDC when using an external AC line adapter.

Disk drive power — The power supply provides a regulated +5 VDC and +12 VDC for the disk drive circuits. In systems using a hard disk drive the power levels are monitored by the low voltage detection circuits. If the voltage drops below a pre-determined point it will not be possible to access the hard disk system. If this happens you must connect an external adapter to the computer and recharge the battery.

Backlight power supply — The backlight power supply output is the only AC output of the power supply module. The backlight requires an AC signal of about 750 Hz with a voltage range from about 20 VAC (low backlight) to about 80 VAC (bright backlight). The backlight circuit consists of three parts: the drive, which includes the brightness control; the on/off, which is controlled by the ELON signal from the computer; and the oscillator, which produces the output to the backlight.

ELG circuit — The ELG circuit provides the operator with a control to manually shut the backlight off by sliding the contrast control to the the minimum position.

LCD driver circuits — The LCD circuits display information on the liquid crystal display of the computer. The LCD driver circuit supplies the power to the LCD driver board. If no power is available, no information will appear on the display.

Chapter 13
The Processors

There are four major circuit groups in the processor and control section of the computer. These major groups are:

- The clock circuits, which are necessary to run the CPU and numeric data coprocessor and perform system timing functions.
- The control circuits, which include the interrupt signals and circuit selection signals.
- The addressing circuits, so that memory and various input/output devices and ports may be accessed.
- The data circuits, to pass data to and from the various parts of the computer.

Because of the complexity of these circuits, a thorough discussion of each is provided in this manual. This chapter discusses the processors, the 80C88 CPU and the 8087 Numerics Coprocessor. A discussion of the other circuits can be found in Chapter 14. For an overview of the operation of the computer, refer to Chapter 12.

Circuit Overview

The simplified block diagram in Figure 13-1 shows how the four major circuits relate to each other.

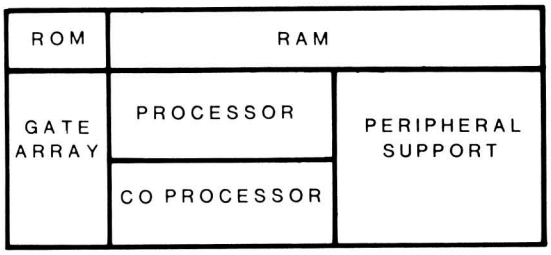


Figure 13-1. System Block Diagram

The clock circuits provide the processor block with the basic timing signals for the operation of the computer. The CPU produces the S0 - S2 timing signals, which feeds the gate array. The timer block develops the TRM1 and TRM2 signals used with the S0, S1, and S2 signals to produce the control signals for the rest of the system. The decoder uses the control signals to produce the chip select signals, which make up part of the control bus. The gate array supplies the remainder of the signals that make up the control bus.

The processor block, consisting of the 80C88 CPU the 8087 numeric data coprocessor, and the gate array, is the source for all addressing within the system. The first eight lines are multiplexed with the data lines and are bidirectional. They are split into the appropriate address and data lines by the bidirectional data buffer and unidirectional address buffer. The gate array uses the DMA address buffers to set up DMA addressing over the address bus.

The interrupt controller block uses one of the outputs of the timer block to initiate an INT 08H instruction in the CPU 18.2159 times per second or every .0549254 seconds as described in Chapter 7. The interrupt controller handles all hardware initiated interrupts and routes them to the CPU for processing.

The other interrupt line is the NMI (nonmaskable interrupt) line. The gate array usually initiates this interrupt to indicate that a power-down condition has started.

Major Integrated Circuits

The major ICs in the processor section of the computer include the 80C88 CPU and the optional 8087 numeric data coprocessor. The following section describes the operation of these integrated circuits.

The 80C88 CPU

The 80C88 is a third-generation microprocessor with a 16-bit internal architecture and an 8-bit external data path to memory and input/output ports. This processor is suitable for a wide range of computer applications. The processor is designed to operate with the 8087 numeric data coprocessor. The design is substantially more powerful than previous 8-bit microprocessors. When compared to the older 8080,

it represents an improvement of four to six times, depending upon the application. The higher performance is due to a pipeline architecture where instructions are prefetched.

Microprocessors usually execute programs by repeatedly cycling through a series of steps known as the execution cycle. The 80C88 execution cycle follows:

1. Fetch the next instruction from memory.
2. Read the operand, if required by the instruction.
3. Execute the instruction.
4. Write the result of the instruction, if required.

Second-generation CPUs that do not use a pipeline design usually perform these steps in a serial manner or with a single bus cycle fetch overlap. The 80C88 design contains two separate, independent, internal processing units (the execution unit and the bus interface unit), which share the previously-listed steps. The execution unit executes the instructions, while the bus interface unit fetches instructions, reads operands, and writes the results of the instruction. This mode of operation results in overlapping fetch and execution steps, decreasing the amount of time required to perform a series of instructions.

The execution unit is a 16-bit arithmetic logic unit (ALU). It maintains the CPU status and control flags, and manipulates the general registers and instruction operands. All internal register and data paths are 16 bits wide. Instructions waiting execution transfer from a queue maintained by the bus interface unit. Accessing memory or an input/output port is accomplished by initiating a request through the bus interface unit.

The bus interface unit performs all bus operations for the execution unit. During periods when the execution unit is busy, the bus interface unit (BIU) "looks ahead" and fetches (retrieves) instructions from memory. The BIU stores these instructions in an internal RAM array called the instruction stream queue. This queue can store up to four bytes in the 80C88.

There are eight 16-bit general purpose registers, four segment registers, and an instruction pointer inside the CPU. The general purpose registers consist of two sets of four registers each. One set is the data registers, sometimes called the H & L group for "high" and "low." The other set is called the pointer and index registers, sometimes called the P & I group. Refer to Table 13-1 for a description of each register.

Table 13-1. 80C88 Internal Registers

MNEMONIC	DESCRIPTION
AX	Accumulator register. This register consists of the 8-bit AH (high) and AL (low) registers. Specifically used by word multiply, word divide, and word input/output instructions.
AH	The high eight bits of register AX. Specifically used by byte multiply and byte divide instructions.
AL	The low eight bits of register AX. Specifically used by byte multiply, byte divide, byte input/output, translate, and decimal arithmetic instructions.
BX	Base register. This register consists of the eight-bit BH (high) and BL (low) registers. Specifically used by translate instructions.
BH	The high eight bits of register BX.
BL	The low eight bits of register BX.
CX	Count register. This register consists of the eight-bit CH (high) and CL (low) registers. Specifically used by string operations and loops.
CH	The high eight bits of register CX.
CL	The low eight bits of register CX. Specifically used by variable shift and rotate instructions.
DX	Data register. This register consists of the 8-bit DH (high) and DL (low) registers. Specifically used by word multiply, word divide, and indirect input/output instructions.
DH	The high eight bits of register DX.
DL	The low eight bits of register DX.
SP	Stack pointer register. The processor uses this 16-bit register for stack operations.

Table 13-1 (Continued). 80C88 Internal Registers

MNEMONIC	DESCRIPTION
	The register contains the current stack address.
BP	Base pointer register. Although the processor does not use this 16-bit register for any specific operation, it does come into use when certain addressing modes are used.
SI	Source index register. The processor uses this 16-bit register for string operations.
DI	Destination index register. The processor also uses this 16-bit register for string operations.
CS	Code segment register. This 16-bit register points to the base address of the current code segment. The CPU fetches instructions from this segment.
DS	Data segment register. This 16-bit register points to the base address of the current data segment. Normally this register contains which usually contains variables for the program being executed.
SS	Stack segment register. This 16-bit register points to the base address of the current stack segment. This segment holds the stack for the CPU.
ES	Extra segment register. This 16-bit register points to the base address of the current extra segment, which is usually used for data storage.
IP	Instruction pointer. This 16-bit register is similar in purpose to the program counter in the 8-bit CPU designs. The contents of this register point to the next instruction location. This information is updated and the next instruction fetched by the bus interface unit. If the instruction pointer was saved on the stack, This information is kept current.

NOTE: All eight general purpose registers fit the definition of accumulator as defined for 8-bit CPUs. Programs can be written to use Many registers other than those specifically mentioned in this table.

You may address the four 16-bit data registers as either one 16-bit register or as two separate 8-bit registers. You may use them freely in most arithmetic and logic operations, but some instructions use specific registers. You may also use the four 16-bit pointer and index registers for arithmetic and logic operations. Except for the base pointer register, some instructions specify the other three registers during normal operations. Refer to Figure 13-2 and Table 13-2.

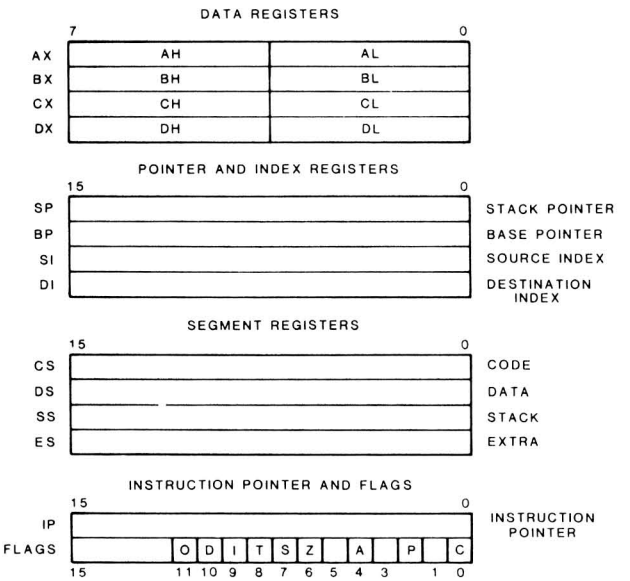


Figure 13-2. 80C88 Internal Register Structure

Table 13-2. Instructions that Use Specific Registers

REGISTER	INSTRUCTIONS
AX	Word multiply, word divide, word input/output
AL	Byte multiply, byte divide, byte input/output, translate, decimal arithmetic
AH	Byte multiply, byte divide
BX	Translate
CX	String operations, loops
CL	Variable shift and rotate
DX	Word multiply, word divide, indirect input/output
SP	Stack operations
SI	String operations
DI	String operations

The 80C88 can address a maximum of one megabyte of memory, but not all at the same time. To make the most effective use of system memory, the addressable memory consists of 64K sections, called

segments. A segment may start on any 16-byte memory boundary, called a paragraph, and contain up to 64 kilobytes. The CPU has direct access to four segments at any one time. The base (starting) address of each segment is stored in its corresponding segment address register in the CPU. The segments may be adjacent, disjointed, partially overlapping, or fully overlapping. A physical memory location may be addressed by one or more of the segments, each having the same or a different value, depending upon the value of the base address for that segment.
Finally, there are six 1-bit status flags and three 1-bit control flags. Table 13-3 contains descriptions of these flags.

Table 13-3. CPU Flags

MNEMONIC	DESCRIPTION
CF	The carry flag, when set (1), indicates that a carry out of or a borrow into the high-order bit of the result has taken place. Instructions that add or subtract multiple-byte values use this flag. Rotate instructions also can isolate a bit from a memory location or a register by rotating it into the carry flag.
AF	The auxiliary carry flag, when set (1), indicates that a carry or a borrow has taken place. Decimal arithmetic instructions use this flag. The flag can indicate a carry out of the low nibble into the high nibble or a borrow from the high nibble into the low nibble of an 8-bit value.
OF	If set (1), the overflow status flag indicates an arithmetic overflow has occurred. The CPU supports an interrupt on overflow instruction so the user can recognize when an overflow takes place.
SF	The sign status flag indicates the sign of the result. If set (1), the result is negative (negative binary numbers are represented in standard two's complement notation). If this flag is clear (0), the result is positive.
OF	The parity flag, when set (1), indicates that the result has even parity (an even number of set bits). You can use this flag to check for data transmission errors.
ZF	The zero status flag, when set (1), indicates that the result of the operation is 0.

Table 13-3 (continued). CPU Flags

MNEMONIC	DESCRIPTION
DF	The direction flag is a control flag. Setting it to 1 causes string instructions to automatically decrement so string processing moves from right to left. If the flag is clear (0), string instructions are automatically incremented so string processing moves from left to right.
IF	The interrupt-enable flag is a control flag. Setting it to 1 allows the CPU to recognize external maskable interrupts. If this flag is clear (0), the processor will ignore interrupts. This flag does not affect non-maskable interrupts.
TF	The trap flag is a control flag. Setting it to 1 puts the processor into a single step mode. When in this mode, the CPU automatically generates an internal interrupt after each instruction executes. This allows the user to inspect the program after each instruction on a step-by-step basis.

The 80C88 can operate in two modes: minimum mode and maximum mode. Most dedicated, turn-key applications use the processor in the minimum mode configuration. Most computer applications, including this computer, use the processor in its maximum mode configuration.

The 80C88 communicates with the rest of the computer by using three types of signals. The signals are the address, data, and control signals. Each group of common signals make up a bus. The purpose of referring to these signals as a bus is to simplify the schematics and circuit descriptions used in the design and maintenance of the computer. Several single lines are multiplexed and require latches to separate the information.

Address Signals — Twenty address lines make up the address bus allowing the CPU to address one megabyte of memory. Sixteen of the twenty address lines provide access to the input/output ports. The eight low order address lines provide both address and data information by sharing multiplexed signals. Refer to Figure 13-3.

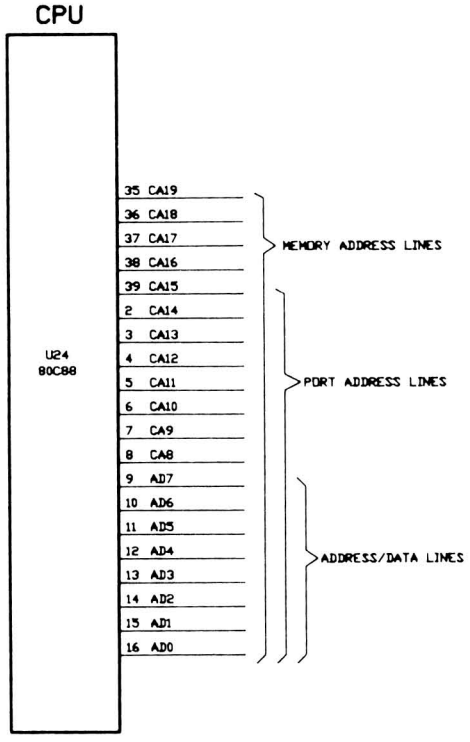


Figure 13-3. Address Line Organization

Data Signals — Eight data lines make up the data bus. The 80C88 communicates with the rest of the computer over this bus. These signal lines are multiplexed with the eight low-order address lines before being separated by bidirectional buffers.

Control Signals — The numerous control signals generated in the 80C88, gate array, and decoder, are the commands that the rest of the computer carries out. These include various read, write, and timing signals.

The CPU carries out its instructions by performing one or more machine operations. The operations that the CPU uses are: fetch (obtaining information over the data lines), execute (manipulating the data in some manner), write (sending data to memory or a peripheral device), or read (receiving data from memory or a peripheral device). You may want to draw a parallel between the read and fetch functions. However, their purposes are slightly different: the fetch function obtains the next instruction where the read function fetches the actual data.

The instruction set for this processor exists at two levels: assembly language level the and machine language level. At the assembly language level, there are about 100 instructions. At the machine language level, there are about 300 instructions. The assembly language compiler takes the various instructions and translates them into the necessary machine language instructions for the CPU.

As an example, there is one basic MOV instruction in assembly language. It can transfer a byte or word from a register, a memory location, or an immediate value, to either a register or memory location. At the machine language level, there are 28 different MOV instructions that accomplish these various tasks. It is much easier for a programmer to remember one general instruction and how to use it.

For a complete discussion of the instruction sets of the 80C88 processor, refer to one of the manuals published by Intel, listed in Chapter 12.

The CPU is a very complex device and must start in an orderly manner. The RESET signal is the one that starts the CPU initialization sequence. When the CPU detects a positive-going edge on the RESET line, all CPU activities stop. When the line goes low, the CPU initializes and the internal flags are cleared. The instruction pointer, DS, SS, and ES registers have a value of 0000H; the CS register has a value of FFFFH; and the queue is empty.

Since the instruction pointer contains 00H and the code segment (CS) register contains FFFFH, the processor fetches its first instruction from memory location FFFF0H. This location is in the multifunction monitor ROM and contains a JMP instruction to the start of the initialization routines. For a complete map of the computer's memory and port addresses, refer to Chapter 6.

The first thing the fetch operation does is to load the contents of the instruction pointer into the address lines through the bus interface unit. As the address output lines become stable, the S0, S1, and S2 lines transition to their correct state (S0 and S1 are high and S2 is active-low). This combination of signals is interpreted by the gate array to produce the AEN1 signal (it goes active-high), which latches the address into the address buffers. The bus interface unit next increments the instruction pointer so that it contains the location of the next instruction.

After the address latches into the bus interface unit, RD and S0 go active-low. This new combination of S0 and S2 low and S1 high causes the gate array to

assert (make active) DT/R low. When this happens, the data from the addressed memory passes through the data buffer back to the CPU where the bus interface unit receives it.

The bus interface unit places the received information in the instruction queue where the execution unit can retrieve it. If it is an instruction, the CPU will execute it. If it is data, the CPU will manipulate it.

The JMP instruction in the multifunction monitor ROM is a JMPF (jump far) instruction to the beginning of the multifunction monitor ROM routines. Therefore, the processor will need to obtain the next four bytes of data to determine the location in memory. Some instructions require two or more bytes to operate, as is the case here. The CPU will repeat the fetch cycle as often as needed to load the remaining parts of the instruction.

The instruction cycle determines what the CPU must do, in addition to fetching instructions. Operations may involve read from or writing to memory, similar to the fetch operation, or they may cause some internal operation to take place.

The S0, S1, and S2 lines carry status signals that identify the type of bus cycle the CPU will require. Table 13-4 describes the types of bus signals as determined by the states of these three lines.

Table 13-4. CPU Bus Cycle Type as Determined by S0, S1, and S2

S0	S1	S2	BUS CYCLE TYPE
L	L	L	Interrupt acknowledge
L	L	H	Read input/output
L	H	L	Write input/output
L	H	H	Processor halt
H	L	L	Instruction fetch
H	L	H	Read memory
H	H	L	Write memory
H	H	H	Passive, no bus cycle required

The data in Table 13-5 is a description of the function of each pin in the 80C88. Refer to Figure 13-4 for the pin locations.

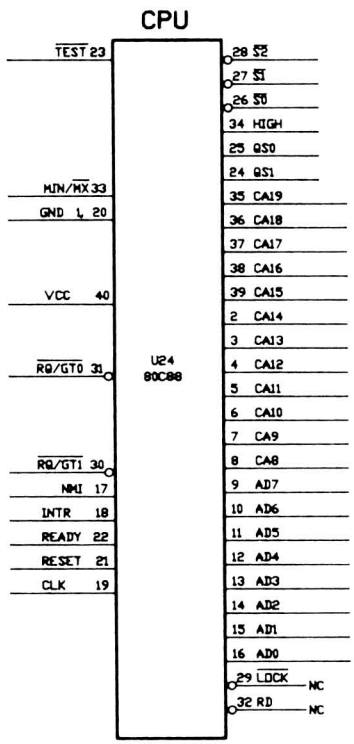


Figure 13-4. 80C88 Pinout

Table 13-5. CPU Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	GND	Ground.
2 - 7	A8 - A14	Address line 8 - address line 14. These bidirectional, three-state address lines carry bits 8 through 14 of the address and are used only for addressing purposes.
9 - 16	AD0-AD7	Address/data line 0 - address/data line 7. These bidirectional, three-state lines are multiplexed to carry bits 0 through 7 for both data and address. At the beginning of a computer cycle they become the low-order address lines; later in the cycle, a program can use them to transfer data to or from the CPU if required by the operation.

Table 13-5 (Continued). CPU Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
17	NMI	Non-maskable interrupt. A low-to-high transition on this input indicates to the CPU that a non-maskable interrupt request is being made.
18	INTR	Interrupt request. A high on this input pin indicates that a maskable interrupt request is being made.
19	CLK	Clock. The input pin receives the system clock signal from the gate array. The CPU uses the clock signal to control all system timing.
20	GND	Ground.
21	RESET	Reset. This input pin receives the system reset signal. Refer to the text for a discussion of the results of this signal.
22	READY	Ready. This input is an acknowledgment from the addressed device (memory or input/output) that it can complete the requested data transfer. Low-speed devices that cannot complete a data transfer in one machine cycle use this signal.
23	$\overline{\text{TEST}}$	Test. Used to synchronize the CPU with an event that is external to the processor. It is used in conjunction with the optional 8087 numeric data coprocessor. The pin is tied high.
24 - 25	QS1-QS0	Queue status 0 - queue status 1. These two output signals provide the optional 8087 with the instruction queue status, specifically the queue operation in the last CLK cycle. Refer to Table 13-7 for the definition of these two signals.

Table 13-5 (Continued). CPU Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
26 - 28	$\overline{\text{S0}}$ - $\overline{\text{S2}}$	Status line 0 - status line 2. The gate array decodes these three, three-state output lines to provide the system with the type of bus cycle being performed by the CPU, as shown in Table 13-4, or the 8087, as shown in Table 13-8, when it has control of the bus.
29	$\overline{\text{LOCK}}$	DMA lockout. This three-state output signal, when low, prevents the DMA circuits in the gate array from gaining control of the system busses.
30 - 31	$\overline{\text{RQ}}$/$\overline{\text{GT0}}$ - $\overline{\text{RQ}}$/$\overline{\text{GT1}}$	Request/grant 0 - request/grant 1. These bidirectional three-state pins are not used in this computer but can provide the system with request (input) grant (output) bus access control for DMA operations.
32	RD	Read. This three-state output pin is not used in this computer. In minimum mode systems, it carries a read pulse. This computer uses the S0, S1, and S2 signals in the gate array to generate equivalent signals.
33	MN/$\overline{\text{MX}}$	Minimum mode/maximum mode. This input pin determines the mode of operation for the CPU. When grounded, the CPU works in maximum mode.
34	HIGH	High. This three-state output pin goes to a high impedance state when the CPU grants a request (GT1). At all other times it is at a logic high.
35 - 39	A15-A19	Address line 15 - address line 19. These three-state output pins carry bits 15 through 19 of the address lines. These pins carry the segment portion of the address.

Table 13-5 (Continued). CPU Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
40	Vcc	+5 VDC power supply input.

Table 13-6. Queue Status Decoding

QS0	QS1	DESCRIPTION
L	L	During the last clock cycle, nothing was taken from the queue.
L	H	The byte removed from the queue was the first byte of the instruction.
H	L	The queue is empty. It was reinitialized by a transfer instruction.
H	H	The byte removed from the queue was the second or subsequent byte of the instruction.

The Optional 8087-2 Numeric Data Coprocessor

You may have an optional 8087 numeric data coprocessor installed on the main board next to the CPU. The 8087 is an arithmetic logic unit (ALU) processor that contains expanded 80-bit registers, an expanded instruction set, and internal logic to perform complex mathematical operations with less programming than required to perform the same task with only the 80C88 CPU.

Because the 80C88 requires more program steps to perform the same functions as the 8087, it takes longer. In some real-time operations, the 80C88 alone may not be able to keep up with the application. The 8087, assuming the software uses it, dramatically increases system throughput involving programmed mathematical functions.

The 8087 shares many of the input, output, and bidirectional signals of the 80C88. This allows it to monitor the same instructions fetched by the CPU. Since the two integrated circuits are designed to work side by side, the CPU ignores 8087 instructions and the 8087 ignores CPU instructions; each processes only its own instructions.

Like the 80C88, the 8087 is internally divided into two processing units. These are called the control unit and the numeric execution unit. The control unit is responsible for fetching instructions, reading values from and writing values to memory, and executing control instructions while the numeric execution unit performs all numeric instructions. The control unit also maintains synchronization with the 80C88 CPU.

Because the 8087 is considered an extension of the 80C88, it does not have registers in the sense that the 80C88 has registers. For a complete description of the operation of the 8087 and its instruction set refer to the publications listed in Chapter 12. The pinout for the 8087 is described in Table 13-7 and illustrated in Figure 13-5.

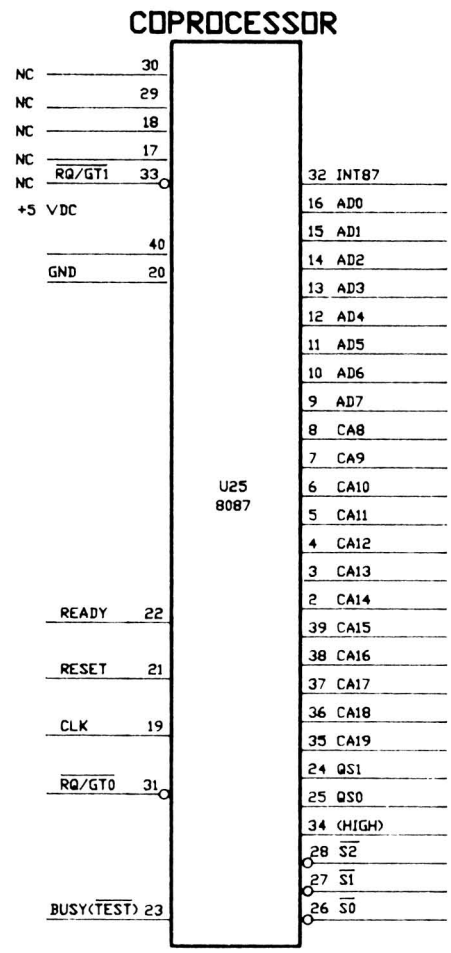


Figure 13-5. 8087 Pinout

Table 13-7. 8087 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	GND	Ground.
2 - 8	A8 - A14	Address line 8 - address line 14. These bidirectional, three-state address lines carry bits 8 through 14 of the address and are used only for addressing purposes.
9 - 16	AD7-AD0	Address/data line 0 - address/data line 7. These bidirectional, three-state lines are multiplexed to carry bits 0 through 7 for both data and address. At the beginning of a CPU cycle, they are used for the low-order address lines; 8later in the cycle, they can be used to transfer data to or from the CPU or 8087, if required by the operation.
17 - 18	—	Not connected.
19	CLK	Clock. This input pin receives the 4.77 MHz or 8 MHz clock from the gate array.
20	GND	Ground.
21	RESET	Reset. The input pin receives the system reset signal. In the 8087, it clears the internal stack.
22	READY	Ready. This signal is developed by the gate array to let the rest of the system know that the addressed device (either memory or an input/output port) can complete the requested data transfer. This signal is normally used with low-speed devices that cannot complete a data transfer in one machine cycle. It is also used for wait-state control.
23	BUSY	Busy. This output is used with the CPU to synchronize it and the 8087.

Table 13-7 (Continued). 8087 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
24 - 25	QS0-QS1	Queue status 0 - queue status 1. These two input signals provide the 8087 with the instruction queue status of the CPU. They tell the 8087 what the last queue operation was. Refer to Table 13-6 for a description of the QS0 and QS1 status lines.
26 - 28	$\overline{S0}$ - $\overline{S2}$	Status line 0 - status line 2. These three, three-state output lines are decoded by the gate array to provide the system with the type of bus cycle being performed by the CPU, as shown in Table 13-4, or the 8087, as shown in Table 13-8, when it has control of the bus.
29 - 30	—	Not connected.
31	RQ/GT0	Request/grant 0. This signal is used by the CPU to provide the 8087 access to the address and data lines.
32	INT	Interrupt. This output is normally routed to the CPU's INTR input via the 8259 programmable interrupt controller. It is used by the 8087 to request an interrupt of the CPU and is routed through the gate array to produce IRQ1.
33	RQ/GT1	Request/grant 1. This signal is not used in this computer. It may be connected to a local 8089, an input/output processor (not used by Zenith Data Systems computers).
34	BHE/S7	Bus high enable/status line 7. This line is tied to the always high HIGH signal of the CPU. It is used for the high eight bits of 16-bit data transfers when the CPU is an 8086 device (but the CPU is an 8088 device in this computer, so this line is not used in this manner).

Table 13-7 (Continued). 8087 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
35 - 39	A19 - A15	Address line 15 - address line 19. These bidirectional, three-state address lines carry bits 15 through 19 of the address and are used only for addressing purposes.
40	Vcc	+5 VDC power supply input.

Table 13-8. 8087 Bus Cycle Type as Determined by S0, S1, and S2

S0	S1	S2	BUS CYCLE TYPE
H	L	H	Read memory
L	H	H	Write memory
H	H	H	Passive; no bus cycle

Chapter 14
Support Circuits

This chapter picks up where Chapter 13 ends. In this chapter you will find information detailing the operation of the CPU support circuitry. This information includes the system memory and the keyboard circuits.

CPU Support Circuitry

A number of integrated circuits make up the majority of the CPU support circuitry. Among these devices are the RP5C15 Real-time Clock, the 82C59 Interrupt Controller, the 82C54 Programmable Interval Timer, the CPU Gate Array, the System Decoder and the Bus Controller Gate Array. Some additional latches and buffers are also used as part of this circuitry.

The RP5C15 Real-Time Clock

The Portable Computer has a real-time clock IC that is connected directly to the address and unbuffered data bus of the system. Data can be written to and read from the IC in the same manner as RAM is written to and read from. A 1 farad "super-capacitor" maintains the time and 26, 4-bit words of non-volatile memory over several days while power is turned off.

Internal counters track time in hours, minutes, and seconds, and the date in days, months, and years. The day of the week is calculated, and leap years, occurring every four years are taken into account. There is no logic to handle the omission of leap years every 100 years, except that the year 2000 is a leap year. Therefore, the calendar chip will be accurate up through the year 2099. Since the year 2100 is not a leap year, the clock chip will not be accurate and in that year will produce a February 29th.

The chip contains 51, 4-bit internal registers that are used as counters, control registers, and non-volatile memory. Four modes control the access to these registers. All data is stored in BCD (Binary Coded Decimal) format since the clock is set up to function as the clock for the computer system. Reading and writing to the clock should be handled through the normal BIOS interrupts, described in Part III of this manual.

Registers 0 - D are read/write registers. Register E and F are write-only registers and will return a zero (0) when you read either one. Registers D - F are common to all four modes. The function of each register is described in the following four tables. Refer to Table 14-1 for mode 0; Table 14-2 for mode 1; Table 14-3 for mode 2; and Table 14-4 for mode 3.

An "x" in the bit column of a table means that the bit may be set (1) or clear (0). This bit determines the value in the counter or register taken in conjunction with the other similarly-marked bits for the same register. A " - " means that the value may be written to, but will always produce a 0 (not set) when read by the system.

Table 14-1. Real-Time Clock Registers in Mode 0

REG. NO.	BIT D3	D2	D1	D0	DESCRIPTION
0	x	x	x	x	Ones of seconds counter. The range is 0 - 9.
1	-	x	x	x	Tens of seconds counter. The range is 0 - 5.
2	x	x	x	x	Ones of minutes counter. The range is 0 - 9.
3	-	x	x	x	Tens of minutes counter. The range is 0 - 5.
4	x	x	x	x	Ones of hours counter. The range is 0 to 9.
5	-	-	x	x	Tens of hours counter. When the clock is operating in the twelve-hour mode, the range is 0 - 1. If bit D1 is set (1), then the time is PM; if D1 is clear (0), then the time is AM. When the clock is operating in the twenty-four hour mode, the range is 0 - 2.
6	-	x	x	x	Day of the week counter. The range is 0 - 6.
7	x	x	x	x	Ones of days counter. The range is 0 - 9.
8	-	-	x	x	Tens of days counter. The range is 0 - 3.
9	x	x	x	x	Ones of months counter. The range is 0 - 9.
A	-	-	-	x	Tens of months counter. The range is 0 - 1.

Table 14-1 (Continued). Real-Time Clock Registers in Mode 0

REG. NO.	BIT D3	D2	D1	D0	DESCRIPTION
B	x	x	x	x	Ones of years counter. The range is 0 - 9.
C	x	x	x	x	Tens of years counter. The range is 0 - 9.
D	x	x	x	x	Clock enable, alarm enable, and mode selection register. If bit D3 is set (1), the clock is allowed to run; if clear (0), the clock will not run. If bit D2 is set (1), the alarm is enabled; if clear (0), the alarm is disabled. If bits D1 and D0 are clear (0), mode 0 will be selected; if bit D1 is clear (0) and bit D0 is set (1), mode 1 will be selected; if bit D1 is set (1) and bit D0 is clear (0), mode 2 will be selected; and if bits D1 and D0 are both set (1), mode 3 will be selected. For normal operation in this computer, the clock is enabled and the alarm is disabled.
E	x	x	x	x	Test register. This register is used during testing by the chip manufacturer for high-speed function testing. If this register is not clear (0), the clock will not operate properly.
F	x	x	x	x	Reset controller. Bit D3 must be clear (0) to enable the 1 Hz signal out the ALARM pin. Bit D2 must be clear (0) to enable a 16 Hz signal. Bit D1, when set (1) will reset all the clock's internal counters. Bit D0, when set (1) will reset the alarm's internal counters (refer to Table 14-2). For normal operation in this computer, only bit D2 of this register is set (1).

Table 14-2. Real-Time Clock Registers in Mode 1

REG. NO.	BIT 3	2	1	0	DESCRIPTION
0	-	-	-	-	Not used.
1	-	-	-	-	Not used.

Table 14-2 (Continued). Real-Time Clock Registers in Mode 1

REG. NO.	BIT 3	2	1	0	DESCRIPTION
2	x	x	x	x	Alarm ones of minutes counter. The range is 0 - 9.
3	-	x	x	x	Alarm tens of minutes counter. The range is 0 - 5.
4	x	x	x	x	Alarm ones of hours counter. The range is 0 - 9.
5	-	-	x	x	Alarm tens of hours counter. The range is 0 - 1 when the clock is in the twelve hour mode and the range is 0 - 2 when the clock is in the twenty-four hour mode.
6	-	x	x	x	Alarm day of the week register. The range is 0 - 6.
7	x	x	x	x	Alarm ones of days register. The range is 0 - 9.
8	-	-	x	x	Alarm tens of days register. The range is 0 - 3.
9	-	-	-	-	Not used.
A	-	-	-	x	Twelve/twenty-four hour mode. If bit D0 is set (1), the clock is in the twenty-four hour mode. If bit D0 is clear (0), the alarm is in the twelve hour mode.
B	-	-	x	x	Leap year counter. This counter follows the ones of years counter and when bits D1 and D0 are both clear (0), the clock logic will generate a nine in the tens of days counter following an eight, when the ones of days and the ones of months are both set (1) to 2, February 29 is to follow February 28. Note that because there is no century counter, the clock cannot determine when the century year is not to be a leap year.
C	-	-	-	-	Not used.
D	x	x	x	x	Clock enable, alarm enable, and mode selection register. If bit D3 is set (1), the clock is allowed to run; if clear (0), the clock will not run. If bit D2 is set (1), the alarm is enabled; if clear (0), the alarm is disabled. If bits D1 and D0 are clear (0), mode 0 will be selected; if bit D1 is clear (0) and bit D0 is set (1),

Table 14-2 (Continued). Real-Time Clock Registers in Mode 1

REG. NO.	BIT 3	2	1	0	DESCRIPTION
					mode 1 will be selected; if bit D1 is set (1) and bit D0 is clear (0), mode 2 will be selected; and if bits D1 and D0 are both set (1), mode 3 will be selected. For normal operation in this computer, the clock is enabled and the alarm is disabled.
E	x	x	x	x	Test register. This register is used during testing by the chip manufacturer for high-speed function testing. If this register is not clear (0), the clock will not operate properly.
F	x	x	x	x	Reset controller. Bit D3 must be clear (0) to enable the 1 Hz signal out the ALARM pin. Bit D2 must be clear (0) to enable a 16 Hz signal. Bit D1, when set (1) will reset all the clock's internal counters (refer to Table 14-1). Bit D0, when set (1) will reset the alarm's internal counters. For normal operation in this computer, only bit D2 of this register is set (1).

Table 14-3. Real-Time Clock Registers in Mode 2

REG. NO.	BIT 3	2	1	0	DESCRIPTION
0	x	x	x	x	Four bits of non-volatile RAM.
1	x	x	x	x	Four bits of non-volatile RAM.
2	x	x	x	x	Four bits of non-volatile RAM.
3	x	x	x	x	Four bits of non-volatile RAM.
4	x	x	x	x	Four bits of non-volatile RAM.
5	x	x	x	x	Four bits of non-volatile RAM.
6	x	x	x	x	Four bits of non-volatile RAM.
7	x	x	x	x	Four bits of non-volatile RAM.
8	x	x	x	x	Four bits of non-volatile RAM.
9	x	x	x	x	Four bits of non-volatile RAM.
A	x	x	x	x	Four bits of non-volatile RAM.
B	x	x	x	x	Four bits of non-volatile RAM.
C	x	x	x	x	Four bits of non-volatile RAM.

Table 14-3 (Continued). Real-Time Clock Registers in Mode 2

REG. NO.	BIT 3	2	1	0	DESCRIPTION
D	x	x	x	x	Clock enable, alarm enable, and mode selection register. If bit D3 is set (1), the clock is allowed to run; if clear (0), the clock will not run. If bit D2 is set (1), the alarm is enabled; if clear (0), the alarm is disabled. If bits D1 and D0 are clear (0), mode 0 will be selected; if bit D1 is clear (0) and bit D0 is set (1), mode 1 will be selected; if bit D1 is set (1) and bit D0 is clear (0), mode 2 will be selected; and if bits D1 and D0 are both set (1), mode 3 will be selected. For normal operation in this computer, the clock is enabled and the alarm is disabled.
E	x	x	x	x	Test register. This register is used during testing by the chip manufacturer for high-speed function testing. If this register is not clear (0), the clock will not operate properly.
F	x	x	x	x	Reset controller. Bit D3 must be clear (0) to enable the 1 Hz signal out the ALARM pin. Bit D2 must be clear (0) to enable a 16 Hz signal. Bit D1, when set (1) will reset all the clock's internal counters (refer to Table 14-1). Bit D0, when set (1) will reset the alarm's internal counters (refer to Table 14-2). For normal operation in this computer, only bit D2 of this register is set (1).

Table 14-4. Real-Time Clock Registers in Mode 3

REG. NO.	BIT 3	2	1	0	DESCRIPTION
0	x	x	x	x	Four bits of non-volatile RAM.
1	x	x	x	x	Four bits of non-volatile RAM.

Table 14-4 (Continued). Real-Time Clock Registers in Mode 3

REG. NO.	BIT 3	2	1	0	DESCRIPTION
2	x	x	x	x	Four bits of non-volatile RAM.
3	x	x	x	x	Four bits of non-volatile RAM.
4	x	x	x	x	Four bits of non-volatile RAM.
5	x	x	x	x	Four bits of non-volatile RAM.
6	x	x	x	x	Four bits of non-volatile RAM.
7	x	x	x	x	Four bits of non-volatile RAM.
8	x	x	x	x	Four bits of non-volatile RAM.
9	x	x	x	x	Four bits of non-volatile RAM.
A	x	x	x	x	Four bits of non-volatile RAM.
B	x	x	x	x	Four bits of non-volatile RAM.
C	x	x	x	x	Four bits of non-volatile RAM.
D	x	x	x	x	Clock enable, alarm enable, and mode selection register. If bit D3 is set (1), the clock is allowed to run; if clear (0), the clock will not run. If bit D2 is set (1), the alarm is enabled; if clear (0), the alarm is disabled. If bits D1 and D0 are clear (0), mode 0 will be selected; if bit D1 is clear (0) and bit D0 is set (1), mode 1 will be selected; if bit D1 is set (1) and bit D0 is clear (0), mode 2 will be selected; and if bits D1 and D0 are both set (1), mode 3 will be selected. For normal operation in this computer, the clock is enabled and the alarm is disabled.
E	x	x	x	x	Test register. This register is used during testing by the chip manufacturer for high-speed function testing. If this register is not clear (0), the clock will not operate properly.
F	x	x	x	x	Reset controller. Bit D3 must be clear (0) to enable the 1 Hz signal out the ALARM pin. Bit D2 must be clear (0) to enable a 16 Hz signal. Bit D1, when set (1) will reset all the clock's internal counters (refer to Table 14-1). Bit D0, when set (1) will reset the alarm's internal counters (refer to Table 14-2). For normal operation in this computer, on bit D2 of this register is set (1).

Table 14-5 describes the pin functions of the real-time clock IC. Refer to Figure 14-1 for an illustration of the pinout.

Table 14-5. Real-Time Clock Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	$\overline{CS}$	Chip select. This active-low line is used for normal chip selection by the system. CS (pin 2) must also be high for the chip to be selected.
2	CS	Chip select. This active-high line is used to disable chip select when a power failure or low voltage condition is detected. CS must also be high for the chip to be selected.
3	ADJ	Adjust. In some applications (not in this computer) this pin is used to adjust the time in the registers when the chip is not connected to a supported CPU. Since the 80C88 CPU of this computer supports the chip, this pin is connected to ground and is not used.
4 - 7	A0 - A3	Address line 0 - address line 3. These four input pins bring the four low-order address lines to the real-time clock IC. They are connected directly to the address bus.
8	$\overline{RD}$	Read. This active low line is asserted when data is requested by the system. $\overline{CS}$ must also be low and CS must be high.
9	GND	Ground.
10	$\overline{WR}$	Write. This active low line is asserted when data is to be written into the chip by the system. $\overline{CS}$ must also be low and CS must be high.
11 - 14	D0 - D3	Data bit 0 - data bit 3. These four pins carry the four low-order bidirectional data signals and are connected directly to the unbuffered system data bus.
15	$\overline{ALARM}$	Alarm. This active-low signal is programmable and may carry an alarm signal, a 1 Hz signal, or a 16 Hz signal. In this computer, this pin carries the 1 Hz signal that is used for the low-power.

Table 14-5 (Continued). Real-Time Clock Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
		indicator. When not low, the output will float, requiring a pull up voltage to go high.
16	OSCIN	Oscillator in. This pin is connected to one side of the 32.768 kHz crystal oscillator.
17	OSCOUT	Oscillator out. This pin is connected, through a resistor, to the other side of the 32.768 kHz crystal oscillator.
18	Vcc	+5 VDC power supply.

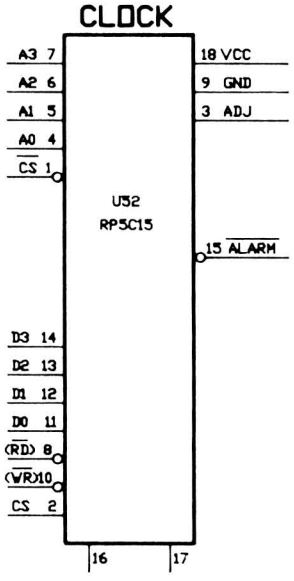


Figure 14-1. Real-Time Clock Pinout

The 82C59 Interrupt Controller

The interrupt controller monitors various devices in the system and notifies the CPU if any of them require immediate attention. These include the timer, keyboard, floppy disk controller, and the input/output ports.

There are two types of interrupts in this computer: maskable and nonmaskable interrupts. Maskable interrupts are handled by the interrupt controller while the non-maskable interrupt goes directly to the CPU and is reserved for serious error conditions and those circuits that have critical (tight) timing requirements. The 82C59 is designed to be used in a interrupt driven environment, such as the one in this computer. It manages up to eight request lines and is programmed as an input/output device. The user may select from several priority modes that can be used to configure the 82C59 to match the requirements of the system. Furthermore, this configuration may be changed "on the fly," while the system is running. This feature allows the system to be configured as various requirements for interrupts arise during operation of the system.

Individual interrupt lines can be masked without affecting those with either higher or lower priority. Of the eight maskable interrupts, three are used by this computer: the system timer (IRQ0 - interrupt request 0) and the keyboard (IRQ1 - interrupt request 1), and the floppy disk controller (IRQ6 - interrupt request 6).

The system timer interrupt, which has the highest priority of the maskable interrupts, is a clock signal generated by the 82C54 and is used by the computer for disk drive timing and to generate certain clock "ticks" for the operating system. The keyboard interrupt has the next highest priority and is generated whenever a key is pressed on the keyboard. The floppy disk interrupt is from the floppy disk controller board, which generates the interrupt at the end of a data transfer cycle.

Programming Considerations

The programmable interrupt controller accepts two types of commands: initialization and operation. Initialization commands must be issued before normal operation can begin. Operation commands allow the 82C59 to operate in one of four interrupt modes: fully nested mode, rotating priority mode, special mask mode, and polled mode.

Since both initialization and operation of the 82C59 are handled through the system BIOS routines, we do not recommend programming the chip directly. For a complete discussion of the interrupt controller and its characteristics, refer to the *iAPX 86, 88 User's Manual.* The following material summarizes that information.

Internal Structure of the Controller

Refer to the block diagram in Figure 14-2 while reading the following material. There are eight major blocks in the chip, seven of which are used in the computer. Lines and arrows determine the interrelationship between these blocks.

The data buffer is an 8-line, bidirectional, three-state buffer that interfaces the address/data bus to the CPU. Instructions, status information, and interrupt vector data are transferred through this buffer.

The control logic is responsible for sending the interrupt signal to the CPU and receiving the interrupt acknowledge back from the CPU. It communicates with the priority resolver and receives information from the in-service register. During initialization and programming, the control logic receives instructions from the read/write logic.

The read/write logic is responsible for handling the programming of the chip, both during initialization and during operation. It also is responsible for releasing status information onto the data bus when requested by the CPU. The controlling lines, treated by the CPU as an input/output port, are the read, write, chip select, and address line 0 lines.

The in-service register is one of three internal registers in the IC. This register is used to store all the interrupt requests that have been acknowledged by the CPU and are currently being serviced. The priority resolver looks at this data to see if an incoming interrupt request is already being serviced by the CPU.

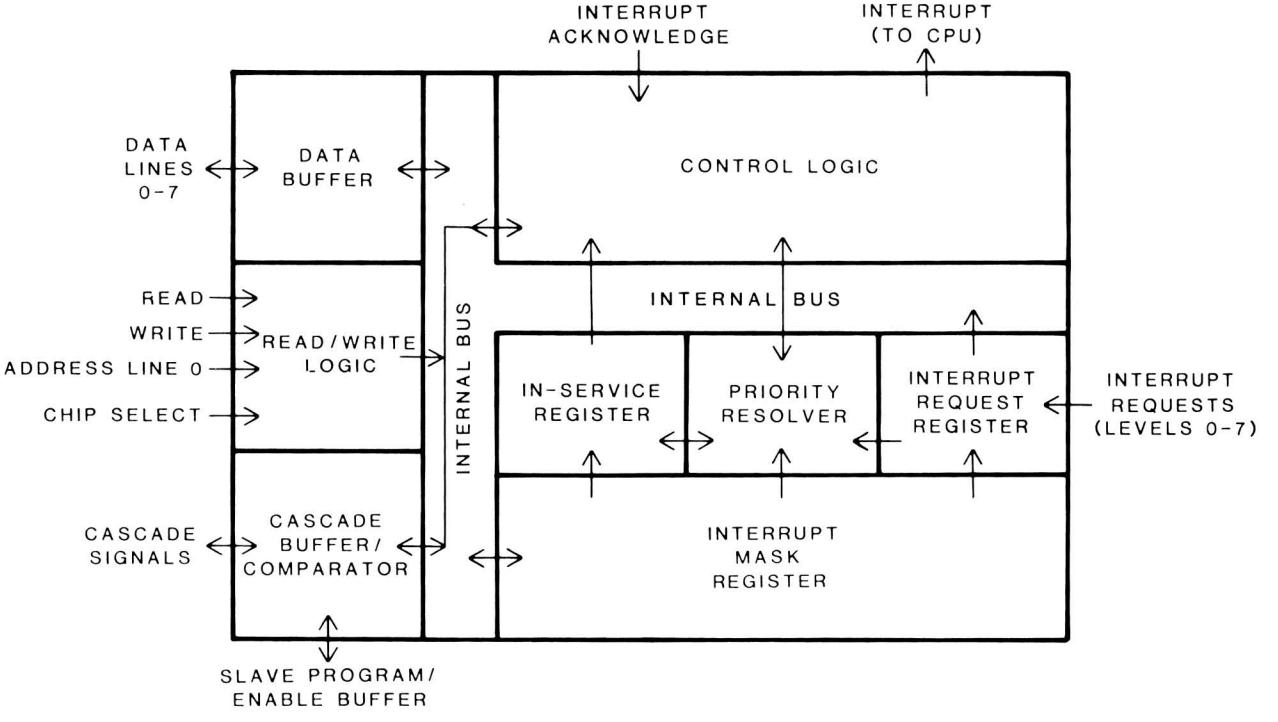


Figure 14-2. Interrupt Controller Block Diagram

The priority resolver determines the priorities of the interrupt requests in the interrupt request register. It does this by first determining if the interrupt request line has been masked by the interrupt mask register and then if the interrupt is in the in-service register, which tells the resolver that the interrupt is already being serviced.

The interrupt request register receives all the incoming interrupt request lines and latches them until they can be processed by the CPU. This register stores all interrupts that are requesting service. Once the interrupt request has been acknowledged by the CPU, it is transferred to the in-service register. However, before the interrupt request register will recognize another interrupt request on the same line, that line must go inactive. If the request has been transferred to the in-service register, the interrupt request register will clear (which will also clear the in-service register) and will recognize a subsequent interrupt request that is received on the same line. This action prevents an interrupt request that has been serviced by the CPU from reactivating the service routine by going through the system a second or subsequent time.

The cascade/buffer comparator is not used in this computer. It is used only in those installations where more than one 8259 is used for a wider interrupt level structure. The PC-compatible design does not require or recognize more than eight interrupt levels.

The interrupt mask register is used to disable selected interrupt request lines. It operates in conjunction with the priority resolver to mask out incoming interrupt requests. Masking one priority level will not affect the operation of higher or lower priority interrupts.

Initialization

As part of initializing the computer system, the CPU must initialize certain internal registers and this programs the interrupt controller by sending it a series of commands. Here is the procedure.

The system will first initialize the SS (stack segment) register of the CPU. This register points to the start of the stack, which is located near the top of scratchpad memory between addresses F0000H and F3FFFH. The exact address depends on the version of the MFM monitor program installed in your computer. Note that this stack location is only used by the MFM monitor program. Under DOS operation, the stack segment register may point elsewhere.

Next, the interrupt vector table in low memory will be initialized. This table contains the addresses of the service routines used by each interrupt. Each interrupt vector address is four bytes long: two bytes are used for the base address for the CS (code segment) register, while the other two are used for the offset address for the IP (instruction pointer).

Finally, the CPU will initialize the interrupt controller with a series of initialization commands. These commands set up the controller for normal operation. Tables 14-6 through 14-9 describe the contents of the four command words that may be sent to the controller. Note that either three or four commands may be sent in sequence. The A0 line, when low, identifies the first command word. Then the A0 line is made high and the following command words are sent to the controller in sequence.

Table 14-6. Initialization Command Word 1

BIT	DESCRIPTION
0	Informs the controller whether there will be three or four command words in the sequence. If the bit is high, there will be four command words in the sequence. If the bit is low, there will be three command words in the sequence.
1	Tells the controller whether it is the only controller in the system or not. If this bit is set (1), only one controller is in the system and no command word 3 will be issued.
2	Tells the controller whether the vector addresses are four bytes long or eight bytes long. The computer uses four byte vector addresses, so this bit will be set (1).
3	Tells the controller to recognize interrupt requests on the leading edge of the interrupt line transition or when the level is high. If this bit is set (1), the interrupt request will be recognized on the edge of the transition.
4	Always set (1).
5 - 7	Not used in this computer. If the 8259 is used in a computer with an 8-bit processor, such as the 8080 or 8085, then these three bits are programmed with address bits A5 - A7 of the vector table address. If the vector addresses are four bytes long, the interrupt controller will automatically supply address bits A0 - A4, depending upon the interrupt requested. If the vector addresses are eight bytes long, the interrupt controller will automatically supply address bits A0 - A5 and bit 5 in this command word will be ignored.

Table 14-7. Initialization Command Word 2

BIT	DESCRIPTION
0 - 2	These three bits are not used in this computer. Refer to the explanation for the bit 0 - 7 configuration.
3 - 7	These five bits are programmed with address bits A3 - A7 in this computer. In computers with 8-bit CPUs, such as the 8080 or 8085, these bits are programmed with address bits A11 through A15. Refer to the explanation for the bit 0 - 7 configuration.
0 - 7	This configuration is not used in this computer. In computers with an 8-bit CPU, such as the 8080 or 8085, these bits are programmed with address bits A8 - A15.

Table 14-8. Initialization Command Word 3

BIT	DESCRIPTION
0 - 7	If this interrupt controller is the master device, these bits let the controller know which interrupt request line has a slave interrupt controller attached to it. Each bit corresponds to an interrupt request line.
0 - 7	If this interrupt controller is a slave device, the three low-order bits are programmed in binary with the value that corresponds to the interrupt line of the master interrupt controller to which this controller is attached. For instance, if this controller is programmed with a six (bits 1 and 2 are set), then this controller would be connected to interrupt request line 6 of the master interrupt controller and bit six in command word 3 for the master interrupt controller would be set.

Table 14-9. Initialization Command Word 4

BIT	DESCRIPTION
0	Tells the interrupt controller whether it is in the 8088/8086 (16-bit CPU) mode or the 8080/8085 (8-bit CPU) mode. If this bit is set (1), the controller is in the 8086/8088 mode.
1	Tells the controller whether to program for automatic end-of-interrupt mode or not. If this bit is set (1), then the controller will be in the automatic end-of-interrupt mode. In the automatic end-of-interrupt mode and in the 16-bit mode the interrupt controller will perform a non-specific end-of-interrupt at the trailing edge of the second interrupt acknowledge received from the CPU. In the 8-bit mode, this operation is performed at the end of the third interrupt acknowledge received from the CPU.
2	If bit 3 of this word is set (1), then this bit determines the whether the interrupt controller is a master or slave controller. If this bit is set (1), the controller is a master controller. If bit 3 of this word is clear (0), then this bit will be ignored.
3	Determines whether the controller is in the buffered mode or not and controls the output of the SP/EN line and the status of bit 2 of this word. If set (1), buffered mode is selected and the SP/EN line becomes an enable output line; bit 2 of this command word will be checked for the master/slave option.
4	Determines whether the interrupt controller works in the fully nested mode where the controller supports more than one level of interrupt and the priority order of the eight lines is arranged from highest (IRQ0) to lowest (IRQ7). If the bit is clear (0), then the controller will not operate in the fully nested mode of operation, then one interrupt will not interrupt the operation of another, but will operate in a sequential manner. Note that the priority order may be changed by software.
5 - 7	Not used.

In this computer, the interrupt controller will be set as follows.

- The interrupt request inputs, IRQ0 - IRQ7, are set to be edge-sensitive. The controller will send an interrupt to the CPU on the positive-going transition of one of these lines.
- Interrupt priority is in reverse numerical order. IRQ0 has the highest priority, IRQ7 the lowest.
- All interrupt masks are cleared.
- If the interrupt controller sends an interrupt to the CPU, the controller will not send any additional interrupts to the CPU until the CPU executes a specific end-of-interrupt instruction.

- The interrupt controller uses vector addresses that are spaced every four bytes in a 32-byte table in memory.

Further Programming

Once programmed, the interrupt controller is ready to accept interrupt requests through its input lines. However, three commands may be sent to the controller that cause it to operate in different modes. These three commands, unlike the initialization command words, do not have to be in sequential order, but may be sent to the controller as needed.

Command word 1 is used to place a mask into the controller's interrupt mask register. Each of the data bits sent to the controller during the write operation correspond to one of the interrupt request lines: bit 0 corresponds to interrupt request line 0, bit 1 to line 1, and so on. If a bit is set (1), the line is masked. If a bit is clear (0), the line is not masked. Address line 0 must be set prior to executing this write operation to the controller.

Command word 2 is used for end of interrupt and rotation commands. It is summarized in Table 14-10. The address line 0 must be clear (0) for the write operation of this command word.

Table 14-10. Operation Command Word 2

BIT	DESCRIPTION
0 - 2	These three bits, coded in a binary format, determine the interrupt level (request line) that is to be acted upon by the remainder of this word. If all three bits are clear (0), interrupt request 0 is programmed; bit 0 set (1) and bits 1 and 2 clear (0) will program interrupt request 1; and so on. Note that bit 6 of the command word can disable the function of these three bits.
3 - 4	These two bits are clear (0), signifying that is command word 2.
5	Determines if an end-of-interrupt command will be issued by the controller. If the bit is clear (0), no end-of-interrupt command will be executed. If the bit is set (1), bit 6 and bit 7 determine the type of end-of-command operation that is executed.
6	Enables or disables the use of bits 0 - 2. If this bit is set (1), the first three bits of the command word are enabled. If this bit is clear (0), they are disabled.
7	Controls all interrupt controller rotation operations. If this bit is set (1), rotation will be performed, depending upon the state of bits 6 and 7 of this command word. If bit 5 and bit 6 are both set (1), then the interrupt request line specified by bits 0 - 2 will be set to the lowest priority and the remainder of the interrupt request levels will conform to the nested mode based upon this interrupt. If bit 5 is set (1) and bit 6 is clear (0), then bits 0 - 2 will be ignored. If bit 5 and bit 6 are both clear (0), then no end-of-interrupt command will be issued by the controller and bits 0 - 2 are ignored. If bit 7 is clear (0), rotation will not be performed.

Command word 3 is used for a number of purposes. The functions performed by command word 3 are summarized in Table 14-11.

Table 14-11. Operation Command Word 3

BIT	DESCRIPTION
0	This bit determines which register (the in-service register or the interrupt request register) is to be read by a read register status operation. If this bit is set (1), the in-service register will be read. If this bit is clear (0), the interrupt request register will be read.
1	This bit determines whether a read register command has been issued or not. If this bit is set (1), then a read register command has been issued. Unless bit 2 is set (1), bit 0 will determine which register is to be read.
2	This bit determines whether a poll command has been issued or not. If this bit is set (1), it overrides bit 1 and and poll command has been issued (bit 1 will be ignored). If this bit is clear (0), a poll command has not been issued.

Table 14-11 (Continued). Operation Command Word 3

BIT	DESCRIPTION
3 - 4	Bit 3 is set (1) and bit 4 is clear (0), signifying that is command word 3.
5	This bit determines whether the special mask mode is to be enabled or disabled. Bit 6 controls the validity of this bit.
6	If this bit is set (1), then bit 5 will be honored and the special mask mode will be enabled or disabled, depending upon the status of bit 5. If this bit is clear (0), bit 5 will be ignored.
7	Not used.

Operation

Operation of the interrupt controller falls into five areas: CPU mode, priority handling, interrupt trigger, status, and cascading.

CPU Mode — The controller is designed to operate with two types of CPUs: eight-bit devices, such as the 8080 and 8085; and sixteen-bit devices, such as the 8088 and 8086. The CPU mode determines the type of device the controller operates with and is programmed during the initialization command word sequence. In this computer, the interrupt controller is always operated in the mode for sixteen-bit CPUs.

When an interrupt takes place in the sixteen-bit mode, the controller will place a single interrupt-vector address byte on the data bus in response to two interrupt acknowledge signals from the CPU.

The first signal is used by the controller to resolve interrupt priorities. The second signal times the placing of the interrupt address byte on the data bus. Bits 0 - 2 are determined by the interrupt request being serviced and bits 3 - 7 are determined by the pattern programmed into the controller during initialization command word 2.

The eight bits placed on the data bus are not used as a direct address, but rather, as the call number for the INT command, executed by the CPU. These INT commands are the same commands used by software, which are explained in Part II of this manual.

When the CPU receives the eight bits from the interrupt controller, the processor multiples the value by four to obtain the address of the interrupt vector address. Then program execution is transferred to the routine at the interrupt vector address.

Priority handling — Priorities fall into two categories: fully nested and masked. The controller handles them in several different modes. These modes are dynamically programmable; the way priorities are handled may be changed during the operation of a program.

The fully nested mode is a general purpose mode that supports multiple levels of interrupt priority. The eight interrupt request lines are handled in a highest-to-lowest manner, usually with interrupt request line 0 as the highest priority. This is the default mode that is set up during the initialization of the controller.

Programming can change which interrupt request line has the highest priority. For example, you can make interrupt request line 4 the highest priority. Interrupt request line 5 then will be the next highest, interrupt request line 6 the next, and so on until you reach interrupt request line 3, which will be the lowest.

Once an interrupt is acknowledged by the CPU, the highest priority request is determined from the interrupt request register. The vector value then is placed on the data bus and the corresponding bit in the in-service register is set. The in-service register bit will remain set until an end-of-interrupt command is executed and sent to the interrupt controller.

In the fully nested mode, once the in-service register bit has been set, all subsequent requests by the same or lower-priority interrupt request line will not generate an interrupt to the CPU. However, a higher-priority interrupt request will be honored and will interrupt the execution of the service routine for the lower-priority interrupt. Since the interrupt request pin on the CPU is disabled when the interrupt acknowledge is sent back to the interrupt controller, the CPU must have an enable interrupt instruction executed before a higher-priority interrupt request can be acknowledged.

Consider the following example. While the main program is being executed, the in-service register will be clear, since no interrupts are being serviced.

Suppose interrupt request line 3 becomes asserted. The interrupt controller notifies the CPU of this and is sent an interrupt acknowledge signal. The controller places the vector byte on the data bus and receives

the second interrupt acknowledge signal. At that point, bit three of the in-service register is set and the CPU starts executing the service routine for interrupt line 3. One of the first instructions to be executed in this routine is the enable interrupts instruction that allows the CPU to receive further interrupt requests from the controller.

Now suppose that while the CPU is executing this service routine, interrupt line 1 is asserted. Again, the interrupt controller notifies the CPU and is sent an interrupt acknowledge signal. The controller places the vector byte on the data bus and receives the second interrupt acknowledge signal. At this point, bit 1 of the in-service register is set and the CPU starts executing the service routine for interrupt line 1. Now both bit 1 and bit 3 are set in the in-service register because the service routine for interrupt line 3 has not been completed.

Now, after the CPU once again receives an enable interrupt command, the controller will act only on interrupt request line 0.

When the service routine for interrupt line 1 has finished, it must inform the controller by executing an end-of-interrupt command. This will clear bit 1 in the in-service register. Next a return instruction must be executed. This will return control to the service routine for interrupt line 3 and will allow any interrupt line from 0 to 2 to be serviced by the system.

When the service routine for interrupt line 3 has finished, the end-of-interrupt command will reset bit 3 in the in-service register and the return will transfer control back to the main program.

The interrupt controller in this computer is almost always in the fully nested mode of operation. Only two programming conditions can disturb this mode: the automatic end-of-interrupt mode and the special mask mode.

Three different end-of-interrupt formats may be programmed: the non-specific end-of-interrupt command, the specific end-of-interrupt command, and the automatic end-of-interrupt command.

The non-specific end-of-interrupt command, while letting the controller know that an interrupt service routine has been completed, does not inform the controller which level of interrupt is involved. By being in a mode where the controller can determine service routine levels, it can determine that the interrupt level that applies to the routine just completed corresponds to the highest interrupt level bit set in the in-service register. The non-specific end-of-interrupt command will reset this bit.

There are two conditions that may cause the non-specific end-of-interrupt routine to fail: when the service routine reset interrupt priorities and when the special mask mode is in use. In both cases, the controller may not be able to determine the routine's interrupt level.

The specific end-of-interrupt command must include the in-service bit to be reset. This allows the programmer the latitude to change interrupt priorities with the servicing routine or perform other functions that might make it vague to the controller which interrupt routine was being serviced, particularly if other service routines were being executed at the same time.

The automatic end-of-interrupt mode eliminates the need for the CPU to issue an end-of-interrupt command to notify the controller that an interrupt service routine has been completed. While in this mode, the controller will perform a non-specific end-of-interrupt at the trailing edge of the second interrupt acknowledge signal from the CPU.

This mode disturbs the fully nested mode because the in-service register bit is reset right after it was acknowledged, leaving no sign that the service routine is being executed. Therefore, any interrupt request (when interrupts are enabled) will get serviced, regardless of its priority, making it possible for an interrupt request to interrupt its own service routine.

It is considered good programming practice not to use the automatic end-of-interrupt mode unless the CPU's interrupt input will be kept disabled while interrupt routines are being serviced.

Rotation of the interrupt priorities is available under several conditions and is provided in two modes: automatic rotation and specific rotation.

Automatic rotation is desirable when the interrupts are equal in nature, for example, when a series of communication channels handle peripheral devices. Once a peripheral is serviced, all other equal-priority peripherals should be given the opportunity of being serviced before the first is again serviced. To accomplish this, the automatic rotation mode assigns the just-serviced line the lowest priority. Automatic rota-

tion may be implemented with the rotate on non-specific end-of-interrupt command or in the rotate on automatic end-of-interrupt mode.

When the rotate on non-specific end-of-interrupt command is given, the in-service register bit being serviced is reset and its corresponding interrupt request line is assigned lowest priority. The other lines' priorities are then rotated to conform to the fully nested format, based on the bit that has been assigned the lowest priority. For example, if bit 3 was just serviced, it is assigned lowest priority, bit 4 the highest, bit 5 the second highest, and so on around to bit 2 which is second lowest.

The automatic end-of-interrupt mode works much the same way, except that the reassignment of priorities takes place at the falling edge of the second interrupt acknowledge received from the CPU.

Specific rotation is completely controllable by the programmer. Through this operation, the specific interrupt request line is selected to receive the highest or lowest priority. Two commands allow this: the set priority command and the rotate on specific end-of-interrupt command.

The set priority command is used to assign an interrupt request line to the lowest priority. The other lines' priorities then are rotated to conform to the fully nested format, based on the bit that has been assigned the lowest priority.

If the set priority command is used during a service routine, then you must use either a specific end-of-interrupt command or the automatic end-of-interrupt mode to end the routine. The non-specific end-of-interrupt resets the highest in-service register bit, which may not represent the service routine that issued the set priority command. If the automatic mode is used, there is no problem because it performs the non-specific end-of-interrupt before the set priority command can be issued. It is still the best practice to use the specific end-of-interrupt command to eliminate any possible confusion.

The rotate on specific end-of-interrupt command is a combination of the set priority command and the specific end-of-interrupt command. With this command, you specify which interrupt request line is assigned to the lowest priority and issue the end-of-interrupt command at the same time. The other lines' priorities are rotated to conform to the fully nested format, based on the bit that has been assigned the lowest priority.

Masking interrupts allows the programmer to enable interrupt request lines that are at a lower priority than the one being serviced.

As an example, suppose interrupt request line 4 has triggered its interrupt service routine, but you want to allow lower-level interrupt request lines to interrupt the service routine. Inside the service routine for interrupt request line 4, you would first mask interrupt request line 4 and then issue a special mask mode command. This disables normal nested mode priority operation and enables all interrupt request lines except those being masked. To leave this mode, the sequence is executed in reverse order.

There is one problem, however, using the mode. You cannot use a non-specific end-of-interrupt command because all masked interrupt request line bits are hidden and are not clearable from the in-service register. Only if the special mask mode has been exited can you use the non-specific end-of-interrupt command to clear masked bits from the in-service register. Therefore, it is the best policy to issue a specific end-of-interrupt command when using this mode.

Interrupt trigger — Two traditional means are used to sense an interrupt: level-sensitive and edge-sensitive.

In the level-sensitive interrupt mode, the interrupt controller will recognize an active high on any of its interrupt request lines. If the interrupt request line remains active after the end-of-interrupt command is issued, another interrupt will be generated if the CPU has been told to recognize interrupts. In this mode, the interrupt must remain active until the first interrupt acknowledge from the CPU has been received. Otherwise, the controller will act as if interrupt request line 7 had been active.

The edge-sensitive mode is used as the default in this computer. In this mode, the interrupt controller recognizes the interrupt on the rising edge as a interrupt request line goes active. If the interrupt request line remains active after the service routine has been completed and the processor set to recognize interrupts, it will not trigger a subsequent interrupt. In this mode, the interrupt must also remain active until the first interrupt acknowledge from the CPU has been received. Otherwise, the controller will act as if interrupt request line 7 had been active.

Status — There may be occasions where the status of the three internal registers needs to be known,

particularly by an interrupt service routine. Polling of interrupts is also possible, though it is not needed in this computer.

The three internal registers of the interrupt controller, the in-service register, the interrupt request register, and the interrupt mask register, can be read by software. The interrupt request register specifies all interrupt request lines that are currently requesting service. The in-service register specifies all interrupt request lines that are currently being serviced by routines. The interrupt mask register specifies all interrupt request lines that are currently masked.

Cascading — Interrupt cascading is not possible in this computer because there is only one interrupt controller installed.

Interrupt Controller Port Address

The interrupt controller is treated as an input/output device for programming purposes. Its addresses are 020H (A0 = 0) and 021H (A0 = 1). In cases where it has been indicated that A0 needs to be cleared, use OUT or IN commands to port address 020H. If A0 needs to be set, use OUT or IN commands to port address 021H.

Sequence of Operation

Once the interrupt controller has been programmed, the CPU must be instructed to set a bit in its flag register to enable interrupts. If one or more circuits generate an interrupt, the system will respond as follows.

- One or more of the interrupt request lines, labeled IRQ0 to IRQ7 for interrupt request, will go high.
- The controller will check the priorities of the incoming interrupts and compare them to others that may also be waiting to be serviced.
- The controller will assert the the INTRQ line to the CPU.
- The CPU recognizes the interrupt request, completes its current instruction, and places the interrupt-acknowledge code on the status bus.
- The interrupt controller will resolve the priority of the interrupts that are requesting service and prepare the vector address value of the selected interrupt line for placement on the data bus during the next interrupt acknowledge cycle.
- The CPU next generates the second interrupt acknowledge signal.
- Upon receipt of the second interrupt acknowledge signal from the CPU, the controller will place the 8-bit interrupt vector address value on the address/data bus.
- The CPU then will multiply the 8-bit value by four to determine the actual address in the interrupt pointer (vector) table for the interrupt service routine.
- The CPU also disables interrupts by clearing the flag bit in its flag register. Then it pushes the flag register, IP (instruction pointer), and CS (code segment register) values onto the CPU stack.
- The new IP and CS values are fetched from the interrupt pointer table and control is jumped to the routine at that address.
- The interrupt-handling subroutine should push any other CPU registers to be used during the service routine onto the stack.
- Typically, at the end of the service routine, an end-of-interrupt command is sent to the controller. This resets the in-service register bit for the service routine, indicating that the routine has finished.
- Before the routine is exited, the appropriate CPU registers are returned from the stack, interrupts are enabled, and control is returned to the instruction following the one the CPU completed when the interrupt service routine was called.

There will be variations in this sequence depending upon a number of factors, the least of which is the possibility that alternate modes can be used by the interrupt controller under program control. However, this should provide you an idea of how a typical operation would take place.

Pinout

Table 14-12 describes the pin functions of the programmable interrupt controller IC. Refer to Figure 14-3 for an illustration of the pinout.

Table 14-12. Programmable Interrupt Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	$\overline{CS}$	Chip select. This active low input signal enables read and write communication (programming) between the CPU and the interrupt controller.
2	$\overline{WR}$	Write. This active low signal allows the interrupt controller to accept command words from the CPU.
3	$\overline{RD}$	Read. This active low signal allows the CPU to read the status of the internal registers of the interrupt controller.
4 - 11	D0 - D7	Data line 0 - data line 7. In this computer, the data lines are connected directly to address/data line 0 - address/data line 7, the bidirectional address/data bus. Command words are sent to, and vector and status data is received from the interrupt controller over these lines.
12 - 13	CAS0 - CAS1	Cascade line 0 - cascade line 1. These lines are not used in this computer. They are used with pin 15 in installations where more than one interrupt controller is cascaded together, allowing control of more than eight interrupt request lines. NOTE: Do not confuse these signal names with the column address strobe signal lines (also identified as CAS lines).
14	GND	Ground.
15	CAS2	Cascade line 2. This line is not used in this computer. It is used with pins 12 and 13 in installations where more than one interrupt controller is cascaded together, allowing control of more than eight interrupt request lines. NOTE: Do not confuse this signal name with the column address strobe signal line 2 (also identified as CAS2).
16	$\overline{SP}/\overline{EN}$	Slave program/enable buffer. This line is not used in this computer. It is used in installations where more than one interrupt controller is cascaded together as an input to designate master controller (when tied high) or slave controller (when low). In that type of installation, this line acts as an output to control buffer transceivers under program control.
17	INT	Interrupt. This pin is asserted high when an interrupt request line is recognized and the service routine is requested. It is used to notify the CPU of the request.
18 - 25	IRQ0 - IRQ7	Interrupt request line 0 - interrupt request line 7. These asynchronous input lines request an interrupt when they are asserted high and held in that state until the first interrupt acknowledge signal is received from the CPU.
26	$\overline{INTA}$	Interrupt acknowledge. This input from the CPU is used to carry the interrupt acknowledge signal to the controller. Two pulses from the CPU are required to complete an interrupt acknowledge sequence.
27	A0	Address line 0. This is used as an input control line, along with the CS, WR, and RD lines.
28	Vcc	+5 VDC power supply.

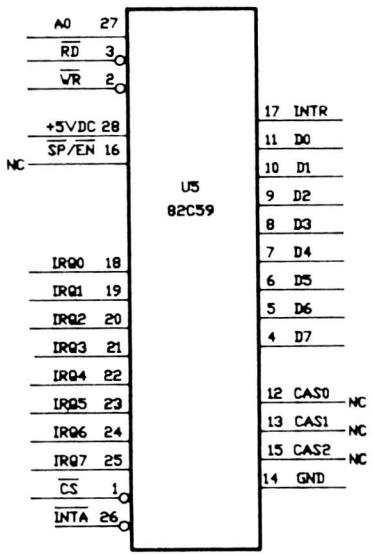


Figure 14-3. Interrupt Controller Pinout

The 82C54 Programmable Interval Timer

The programmable interval timer, used to generate time delays of varying lengths, eliminates the need for software timing loops that are subject to such variables as CPU clock speed and interrupt service routines. Instead, the programmer can configure the timer to match his requirements and initialize one or more of the three counters in the timer with the desired values. Then, upon command, the timer will count to the value and issue an interrupt request upon completion of the task.

The timer also can be used as a programmable rate generator, event counter, binary rate multiplier, real-time clock, digital "one-shot", or complex motor controller. Not all of these functions are used in this computer.

The timer can operate in one of six modes: interrupt on terminal count, programmable one-shot, rate generator, square-wave rate generator, software-triggered strobe, and hardware-triggered strobe.

Refer to Figure 14-4, the block diagram of the timer, for the following discussion.

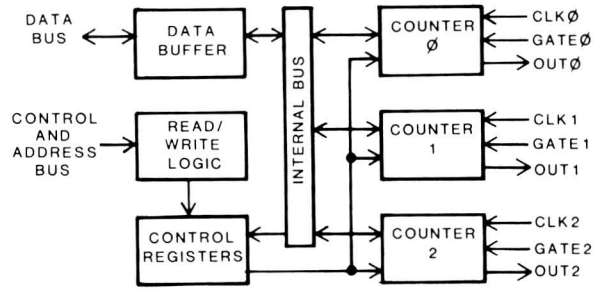


Figure 14-4. Programmable Interval Timer Block Diagram

The timer is divided into six logical sections: data buffer, read/write logic, control register, and three counters.

The data buffer is a three-state, bidirectional buffer used to interface the timer with the address/data bus. Data is sent to or received from the bus via IN or OUT commands to ports 040H - 043H.

The read/write logic accepts the IOR, IOW, CS54, A0, and A1 signal lines as inputs to determine the type of operation to be performed inside the timer. CS53 controls chip access, while address lines 0 and 1 determine the type of read or write operation requested.

The control registers receive the instructions that program the counters. The control word for each counter is selected by the value in bits 5 and 6. Table 14-13 describes the contents of the registers. Since all three registers (one for each counter) are identical, only one register is described.

Table 14-13. Timer Control Registers

BIT	DESCRIPTION
0	Establishes the counter format: 16-bit binary or 4-digit binary-coded decimal. If the bit is set (1), the counter will be in the 4-digit binary-coded decimal format. If the bit is clear (0), the counter will be placed in the 16-bit binary format.

Table 14-13 (Continued). Timer Control Registers

BIT	DESCRIPTION
1 - 3	Establishes the mode for the selected counter. If they equal zero, mode 0 is selected; 1 selects mode 1; 2 or 6 selects mode 2; 3 or 7 selects mode 3; 4 selects mode 4; and 5 selects mode 5. Refer to the text for a description of each mode.
4 - 5	Controls the loading (writing) of the count into the specified counter. If the value placed in these bits is 0, the counters are latched; if the value is 1, only the most significant byte will be read or loaded; if the value is 2, only the least significant byte will be read or loaded; if the value is 3, the least significant byte will be read or loaded first, followed by the most significant byte. In all cases, the number of bytes (one or two) must be read or loaded before a different operation can be applied to the specified counter.
6 - 7	Specifies the counter to receive the operation. If the value is 0, counter 0 is specified; a value of 1 specifies counter 1; and a value of 2 specifies counter 2. If the value is 3, no counter is selected; the value is illegal.

Each of the three counters contain two bytes and may be configured as either a 16-bit binary counter or a 4-digit binary-coded decimal counter. The input, gate, and output lines are configured by the modes programmed through the control register during programming. In addition, each counter may be read selectively without first inhibiting the clock input for the counter being read.

Mode Definitions

Many of the modes require that specific hardware conditions do not exist. For example, the gates, clocks, and outputs must be free; not tied to specific circuits. While the programming is fully explained in this section, many of the functions are not available, since the timer is more or less dedicated to specific tasks as proscribed by PC compatibility. For example, gates 0 and 1 are tied high in this computer and cannot be used to control counters 0 and 1. Gate 2 is tied to the PB0 signal, which is generated by the gate array whenever a key is pressed. Counter 2's output is sent back to the gate array where it is processed and eventually feeds the speaker to produce the audible key click heard whenever a key is pressed.

The other two counters are equally committed to specific tasks: counter 0 is used to provide the CPU with memory refresh timing and counter 1 is used with DMA requests. So, none of the counters can be controlled fully to produce special timing signals beyond those allowed within the confines of the computer's design.

Interrupt on Terminal Count (Mode 0) — Figure 14-5 illustrates Mode 0. This straight-forward mode will decrement the counter until it reaches 0 and then assert its output line high.

The upper set of lines illustrate the counter without receiving any low signals on the gate input line. Once the counter has been loaded, in this case, with a value of 4, the decrementing of the counter starts. When the counter reaches 0, the output will go high. Counting will not resume until the counter is reloaded.

The lower set of lines illustrate the counter being affected by the gate line. In this case, the counter is loaded with a value of 5. As long as the gate line remains high, the counter will decrement. When the gate line goes low, decrementing the counter will halt, but it is not reset to the original value. When the gate line goes high again, decrementing the counter will resume. When the counter reaches 0, the output will go high. Counting will not resume until the counter is reloaded.

In this mode, the output of the counter should be tied to an interrupt line of the CPU.

The gate signal will disable counting when low and enable counting when high.

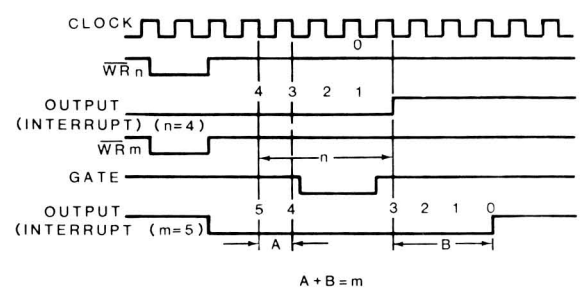


Figure 14-5. Timing Mode 0

Programmable One-Shot (Mode 1) — Figure 14-6 illustrates Mode 1. In this mode, the counter acts like a programmable one-shot. The output will go low the following clock cycle after the gate is asserted high (acting as a trigger). The output will remain low until the counter has decremented to 0.

If the trigger (the gate input line) goes low and then high again, the counter will be reset and will be held low until the counter decrements to zero.

The counter, when acting as a one-shot, is retriggerable. Therefore, the output will remain low until the counter has decremented from the original value to zero following the rising edge of the gate. The rising edge of the gate reloads and reinitiates counting.

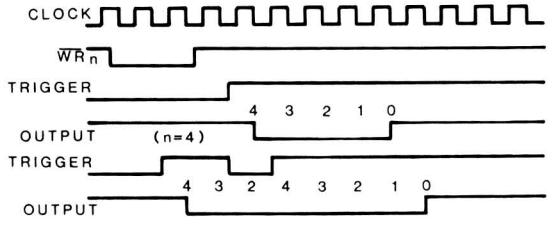


Figure 14-6. Timing Mode 1

Rate Generator (Mode 2) — Figure 14-7 illustrates Mode 2. In this mode, the counter will produce a low for one clock cycle on its output pin every n clock cycles. As long as the gate remains high, the counter will repeatedly decrement the value to zero and start over. If the counter's value is changed, the new value will take effect for the next counting cycle and not affect the current cycle. This action is illustrated in the upper set of lines.

The gate can be used to synchronize the counter. While the gate input is low, the counter will not decrement. When the gate goes high, the counter will reloaded with its programmed value and start decrementing. This action is illustrated in the lower set of lines.

When the gate is low, counting is disabled and the output is made high. The rising edge of the gate reloads and reinitiates counting. When the gate is high, counting is enabled.

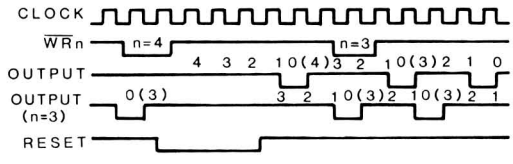


Figure 14-7. Timing Mode 2

Square Wave Rate Generator (Mode 3) — Figure 14-8 illustrates Mode 3. In this mode, the counter is decremented by two each clock cycle, rather than one, effectively cutting the time it takes to reach zero in half.

If the programmed value is even, the counter is always decremented by two. When the counter reaches zero, the state of the output will change (for example, from high to low), the counter is reloaded, and the decrementing by two starts over. When the counter reaches zero, the output state again changes (for example, back to high), the counter is reloaded, and the process starts over again. This produces an even square wave.

If the value is odd and the output is high, the first clock cycle will decrement the counter by one and then by twos until it reaches zero. Then the clock will change state to low, the counter will be reloaded with the programmed value, and the first clock pulse will decrement it by three, then by twos until it reaches zero. This produces a signal where the output is high for $(n+1)/2$ counts and low for $(n-1)/2$ counts.

When the gate is low, counting is disabled, and the output is high. The rising edge of the gate initiates counting. When the gate is high, counting is enabled.

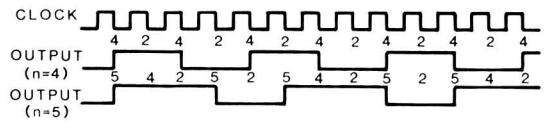


Figure 14-8. Timing Mode 3

Software Triggered Strobe (Mode 4) — Figure 14-9 illustrates Mode 4. In this mode, the counter, when it reaches zero, will place a single pulse that lasts for one clock cycle on its output. Counting will begin when the counter is programmed with a value.

When the gate is low, counting is disabled; when the gate is high, counting is enabled.

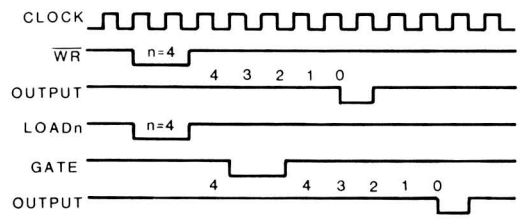


Figure 14-9. Timing Mode 4

Hardware Triggered Strobe (Mode 5) — Figure 14-10 illustrates Mode 5. The counter, when it reaches zero in this mode, will place a single pulse that lasts for one clock cycle on its output. The counter will not start decrementing its value until it senses the rising edge of the gate input. The counter is retriggerable and will reload after it reaches zero.

The rising edge of the gate signal initiates counting.

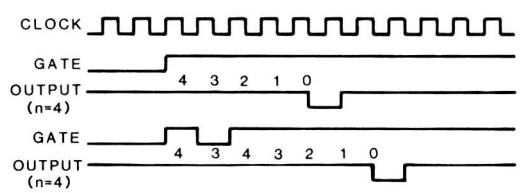


Figure 14-10. Timing Mode 5

Programming Considerations

The monitor ROM is used to program the mode and initial value in each of the three counters in the timer. In each case a control word is placed in the selected counter's register and then the programmed number of bytes (one or two) before the counter is started.

There is no special sequence in which the timer must be programmed. For example, you can write a command word to any of the three counters without affecting the operation or programming of the other two. Suppose you want to load counter 0 with the least significant byte only, counter 1 with both bytes, and counter 2 with the most significant byte only. The following are two possible sequences that accomplish this task.

Load Sequence 1

1. Load command word 0, 1, and 2.
2. Load the most significant byte in counter 2.
3. Load the least significant byte in counter 0.
4. Load the least significant byte in counter 1.
5. Load the most significant byte in counter 1.

Load Sequence 2

1. Load command word 2.
2. Load the most significant byte in counter 2.
3. Load command word 0.
4. Load command word 1.
5. Load the least significant byte in counter 1.
6. Load the least significant byte in counter 0.
7. Load the most significant byte in counter 1.

NOTE: The only sequence that has to be followed is in loading two bytes into a counter. In both examples, the least significant byte is followed by the most significant byte before the counter can start operating. In the last example, the least significant byte and most significant byte for counter 1 are separated by the least significant byte for counter 0.

All counters decrement only; if you load a counter with a value of zero, the actual count will equal the maximum value (10,000 in BCD; 65,536 in binary).

The values in the counters may be read. The address lines determine which counter is read. Normally you would read the contents of the counter with normal read operations. However, you may wish to read the value in the counter while it is operating. This presents problems because you may not get a valid result while the counter is continuing to decrement. The timer offers a method of latching the output of the counter so that it is stable while it is being read. The counter continues to decrement during the read operations but the data remains as it was when it was latched.

The timer can read or write values to the counters through IN (read) and OUT (write) programming instructions. Table 14-14 describes each read and write operation that may be performed on the timer.

Table 14-14. Timer Read and Write Operations

PORT	OPERATION	DESCRIPTION
040H	OUT	Write (load) counter 0.
040H	IN	Read the contents of counter 0.
041H	OUT	Write (load) counter 1.
041H	IN	Read the contents of counter 1.
042H	OUT	Write (load) counter 2.
042H	IN	Read the contents of counter 2.
043H	OUT	If either bit 4 or bit 5 are set (1), write a control word to the specified counter's control register. Refer to Table 14-13 for a description of the contents of the control word. If bits 4 and 5 are both clear (0), latch the specified counter's value for reading. NOTE: The specified counter is identified in bit 6 and bit 7 of the control word.
043H	IN	Place the data buffer in the high-impedance state; do not perform any operation.

Pinout

Table 14-15 describes the pin functions of the programmable interrupt controller IC. Refer to Figure 14-11 for an illustration of the pinout.

Table 14-15. Programmable Interval Timer Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1 - 8	D7 - D0	Data bit 0 (pin 8) - data bit 7 (pin 1). These pins are three-state bidirectional data lines that carry 8-bit data to and from the timer's counters. The system lines are AD7 - AD0.

Table 14-15 (Continued). Programmable Interval Timer Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
9	CLK0	Clock 0. This line is TCLK from the gate array. It controls the timing for all three counters in the IC. This counter is dedicated to producing the signal that initiates INT 08H.
10	OUT0	Output 0. This line feeds IRQ0 of the interrupt controller to initiate INT 08H.
11	GATE0	Gate 0. This line is tied high so counter 0 always runs.
12	GND	Ground.
13	OUT1	Output 1. This pin feeds TMR1, which feeds the gate array with various timing signals, depending upon the required operation.
14	GATE1	Gate 1. This line is tied high so counter 1 always runs.
15	CLK1	Clock 1. This line is TCLK from the gate array. It controls the timing for all three counters in the IC.
16	GATE2	Gate 2. This pin is fed by TMR2GATE from the gate array. It allows user control over the programming f the third counter.
17	OUT2	Output 2. This pin feeds TMR2, which feeds the gate array with the output of the third timer. The gate array can route this signal to the speaker to produce sound.
18	CLK2	Clock 2. This line is TCLK from the gate array. It controls the timing for all three counters in the IC.
19 - 20	A0 - A1	Address line 0 - address line 1. These two lines are used to select the port address of the timer. Refer to Table 14-14 for the ports used by read and write operations. $\overline{CS}$ is used in conjunction with these lines.
21	$\overline{CS}$	Chip select. This active low input line, in conjunction with A0 and A1, determine the port address and actively select the timer for read and write operations. The system line is CS54.

Table 14-15 (Continued). Programmable Interval Timer Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
22	$\overline{RD}$	Read. This active low input line will cause the timer to perform a read operation. Refer to Table 14-14 for port addressing.
23	$\overline{WR}$	Write. This active low input line will cause the time to perform a write operation. Refer to Table 14-14 for port addressing.
24	Vcc	+5 VDC power supply.

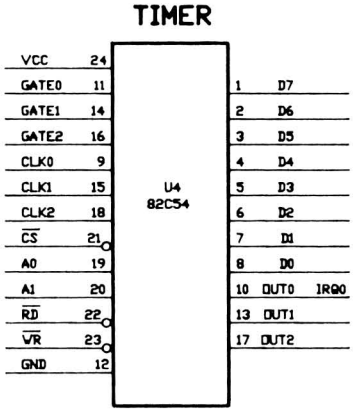


Figure 14-11. 82C54 Programmable Interval Timer Pinout

The Gate Array

Most of the logic in the computer is contained in three devices: the gate array, the decoder, and the monitor ROM. The decoder is described in the next section and the monitor ROM is described in Chapters 4 and 6.

The gate array is a custom device that replaces many of the buffers, latches, and other devices normally required to support the CPU. It generates a number of signals, including many of those associated with the control bus. DMA control and clock circuits are now handled in this device.

As a signal generator, the gate array provides a 4.77 or 8 MHz, (depending on the configuration switch setting) 33% duty cycle clock signal to the CPU. It also provides synchronization for the READY signal and provides the RESET signal at powerup. Additional timing signals (TCLK and CLK) are sent to the timer (previously described in this chapter) and to the control bus to provide synchronized operation with other devices.

The gate array decodes the S0, S1, and S2 status signals from the processor and 8087 to provide latch and enable signals to the data and address buffers and latches. Read and write signals are generated for memory and peripheral control.

Except for the DMA functions, there are no special programming functions for the gate array since it replaces components that do not have registers.

DMA Function

DMA is used to speed the transfer of data from one memory location, device, or peripheral to another memory location, device, or peripheral. The primary advantage to DMA is in moving blocks of data. The DMA controller has hardware instructions that operate much faster than software-based CPU instructions. This computer uses a modified 8237 design in the gate array as the DMA controller. Refer to Figure 14-12 for a block diagram of the DMA design.

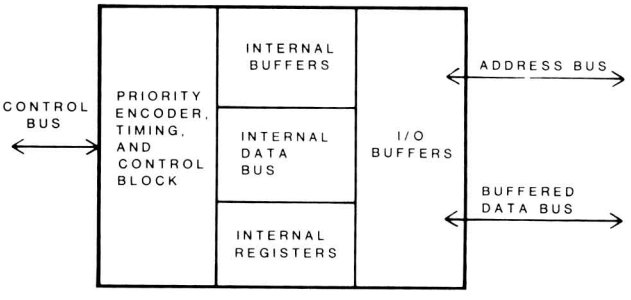


Figure 14-12. DMA Controller Block Diagram

DMA controllers use three modes: single byte transfers, defined block transfers, and gated block transfers. The second mode is controlled with start and end addresses defined by CPU instructions. The third mode is controlled by an external signal, which is not used in this computer.

The controller circuitry operates in one of two types of cycles: the idle cycle and the active cycle. The controller is in an idle cycle when the CPU has control of the system bus. During this time, the CPU can program the controller or read the status registers. The active cycle occurs when the controller has control of the system bus. Some of the input lines become output lines so the controller can send address and control signals to the system bus.

When a transfer of data is desired, the CPU tells the controller where to get the data and where to put it in memory. The controller then handles the data transfer. It generates up to 16 address lines through address outputs A0-A7. The high-order address is first placed on the address bus and latched by a high on the LD1 line. The controller only latches the high-order address when it changes, speeding the process. The address latches and buffers are enabled by a high on the DMA line. The gate array generates all of the memory and input/output read and write signals (MEMR, MEMW, IOR, and IOW) from the S0, S1, and S2 lines through an inverter.

When the DMA is to control the bus, a hold request signal looks for the CPU idle cycle, as defined by the S0, S1, and S2 lines. When the idle cycle is recognized, the controller locks the CPU in a hold state until the transfer is completed.

A peripheral can request DMA service from the controller on any of the DRQ lines; only DRQ2 is used in this computer. The controller responds to the input by requesting a CPU hold, and then sending a DMA acknowledge output to the peripheral when the CPU enters a hold state.

The 8237A design is not fully implemented inside the gate array. For example, the gate array design contains three, instead of four, channels. In addition, the gate array design does not include rotating priority logic or a read buffer. You will also note differences between the pinout of an 8237 versus the pinout of the gate array. This is because the gate array handles some of the support logic for the 8237 and does not need to bring these signals to the outside world. Also, the DMA controller's operation in the computer is specialized and does not need to support the generalized application of the stand-alone chip.

For complete information on programming and using the DMA controller, refer to the data sheet for the Intel 8237A integrated circuit, keeping in mind the differences between the actual stand-alone controller and the implementation used in the gate array.

Pinout

The gate array package is a sixty-eight pin device. The pinout is illustrated in Figure 14-13 and described in Table 14-16.

Table 14-16. Gate Array Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	GND	Ground.
2	TMR2	Timer 2. This input is the output of counter 2 in the programmable interval timer. It is used for generating sound from the speaker in the computer when enabled by the gate array.
3	TCLK	Timer clock. This signal is the clock input to the programmable interval timer. It is considerably slower than the clock signal that runs the processors and must not exceed 2 MHz.
4	TMR2GATE	Timer 2 gate. This signal is the counter 2 gate input to the programmable interval timer.
5	READY	Ready. The gate array develops this signal to let the rest of the system know that the addressed device, either memory or an input/output port, can complete the requested data transfer. This signal is normally used with low-speed devices that cannot complete a data transfer in one machine cycle and is used for wait-state control. Internally, the DMA processor uses this signal as an input to extend memory read and write pulses to accommodate slow memories or input/output peripheral devices.
6	NMI	Nonmaskable interrupt. A low-to-high transition on this line is used to indicate to the CPU that a nonmaskable interrupt request is being made. A number of signals that check the status of various parts of the system can generate this signal including the keyboard, the input/output channels, and memory.
7	INT87	Interrupt 8087. This input from the INT pin on the 8087 is combined with other interrupt requests in the gate array to produce IRQ1, which is sent to the programmable interrupt controller.
8 - 10	$\overline{S0}$, $\overline{S1}$, $\overline{S2}$	Status line 0 - status line 2. These three input lines are decoded by the gate array to provide the sys-

Table 14-16 (Continued). Gate Array Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
		tem with the type of bus cycle being performed by the CPU, when it has control of the bus. These signals are also used by the DMA circuits in the gate array to detect a hold acknowledge signal from the CPU.
11	DT/$\overline{R}$	Data transmit/receive. This signal, developed by the gate array, directs the throughput of the buffer gate array.
12	A7	Address line 7. This signal, bit 7 of the address bus, provides input to the gate array for synchronization and creation of selected signals.
13	A5	Address line 5. This signal, bit 5 of the address bus, provides input to the gate array for synchronization and creation of selected signals.
14 - 17	A3 - A0	Address line 0 (pin 17) - address line 3 (pin 14). These four least significant address lines are bidirectional three-state signals. They provide input to the gate array for synchronization and creation of selected signals. The DMA controller circuits use these lines as inputs to address the control register in the controller to be loaded or read. In the DMA active cycle, they are outputs and provide the lower 4 bits of the output address.
18 - 20	BD0 - BD2	These three-state output lines carry data bit 0 - data bit 2 to the rest of the system. Data bits 3 - 7 are on pins 60 and 56 - 59.
21	DRQ2	Data request 2. This line which comes from the disk controller circuitry, feeds the DMA circuits of the gate array and sets up the system for a user- or disk-controlled DMA transfer.
22	DRQ3	Data request 3. Tied to ground — not used.

Table 14-16 (Continued). Gate Array Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
23	I/O CH RDY	Input/output channel ready. This pin is fed by WAIT, generated by the 6355 MEMRDY signal (pin 60). This input signal is used by the gate array to determine when to address and read video memory.
24	DRQ1	Data request 1. Tied to ground — not used.
25	$\overline{\text{I/O CH CHK}}$	Input/output channel check. Tied high — not used.
26 - 28	LD0 - LD2	Data latch 0 - data latch 2. These output lines control the loading of the three data-to-address latches between the buffered data bus and the address bus.
29	DMA	Direct memory access. This output line is used by the DMA controller to strobe the address that has been loaded into the latches onto the address bus.
30	GA INTA	Interrupt acknowledge. This output pin's signal is derived from the S0, S1, and S2 signals that are generated by the CPU. This signal feeds the interrupt acknowledge line to the interrupt controller and is used by the CPU to acknowledge the receipt of an interrupt request from the controller.
31	IOW	Input/output write. This is a bidirectional line. In the DMA idle cycle, this is an internal output control line that is asserted by CPU signals when data is to be written to the DMA controller registers. During the DMA active cycle, this is an output line to the system bus that is asserted by the DMA controller to write the contents of the system data bus to the selected input/output port. Otherwise, this is an output signal, produced by the gate array from the S0, S1, and S2 signals that goes active during any write port address cycle. During a DMA process, both the IOW and the MEMR will be active at the same time.

Table 14-16 (Continued). Gate Array Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
32	$\overline{\text{RESET}}$	Reset. This signal is produced by the gate array when the power is first applied to reset the major circuits and ICs of the computer. It is derived from the RSTI signal feeding pin 45. The signal is used internally to clear the DMA registers.
33	Vcc	+5 VDC power supply.
34	ROW	Row. This line, produced by the gate array, selects the address row for the memory address multiplexers and the decoder.
35	GND	Ground.
36	BALE	Buffered address latch enable. Not connected — not used.
37	CLKG	Clock. This is the system CPU clock (4.77 or 8 MHz).
38	$\overline{\text{T/C}}$	Terminal count. This active low signal indicates terminal count has been reached by the DMA circuits.
39	MEMR	Memory read. This output signal, produced by the gate array from the S0, S1, and S2 signals, goes active during any read memory cycle. In the idle DMA cycle, this line is asserted along with IOW, to transfer data from the addressed memory location to the selected input/output port.
40	MEMW	Memory write. This output signal, produced by the gate array from the S0, S1, and S2 signals, goes active during any write memory cycle. In the DMA idle cycle, this line is controlled by CPU signals. In the active DMA cycle, this line is asserted along with IOR to transfer data from the selected input/output port to the addressed memory location.
41	IOR	Input/output read. This is a bidirectional line. In the DMA idle cycle, this is an internal input control line generated by the S0, S1, and S2 signals from the CPU and is asserted when the CPU reads data from a DMA controller register. During the DMA active cycle, this is an output line to the system bus that is asserted to read data from the selected input/output port. Otherwise, this is an output signal, produced by the gate array from the S0, S1, and S2 signals, which goes active during any read port address cycle. During a DMA process, both the IOR and the MEMW will be active at the same time.
42	MEM	Memory. This output feeds the decoder for memory operations.
43	KBDD	Keyboard data. This input is the data signal from the keyboard decoder.
44	KBDC	Keyboard data clock. This input is the data clock from the keyboard decoder.
45	RSTI	Initiate reset. This input from the R-C network triggers the RESET signal (refer to pin 32).
46	O5OSC	5 MHz oscillator output.
47	JMPRN	Read jumpers.
48	I8OSC	24 MHz oscillator input.
49	I5OSC	14.318 MHz oscillator input.
50	Vcc	+5 VDC power supply.
51	O8OSC	8 MHz oscillator output. Not used in this computer.
52	GND	Ground.
53	8M/$\overline{\text{5M}}$	8/5 MHz. This input determines the operating speed of the gate array.
54	$\overline{\text{PB6}}$	This active low line disables the keyboard clock input.
55	$\overline{\text{SPKR}}$	This output feeds the speaker's circuits to produce sound for the system.
56 - 59	BD4 - BD7	These three-state output lines carry data bit 4 - data bit 7 to the rest of the system. Data bits 0 - 3 are on pins 18 - 20 and 60.
60	BD3	This three-state output line carries data bit 3 to the rest of the system. Data bits 0 - 2 and 4 - 7 are on pins 18 - 20 and 56 - 59.
61	CAS	Column address strobe. This output is used by the system for memory refresh of the dynamic RAM ICs.

Table 14-16 (Continued). Gate Array Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
62	PARITY	Parity. Tied high — not used.
63	$\overline{\text{CPUGA}}$	CPU gate array. This input is used to enable he input/output port addresses of the gate array.
64	AEN1	Address enable. This output disables the CPU address drives during DMA operation.
65	DEN	Data enable. This signal is used by the decoder for timing with the rest of the system.
66	IRQ1	Interrupt request 1. This output signal is generated by the gate array and sent to the interrupt controller. It pulses 18.2159 times a second to trigger INT 08H (refer to Chapter 8).
67	Vcc	+5 VDC power supply.
68	TMR1	Timer 1. This input from the programmable interval timer triggers the IRQ1 output 18.2159 times a second.

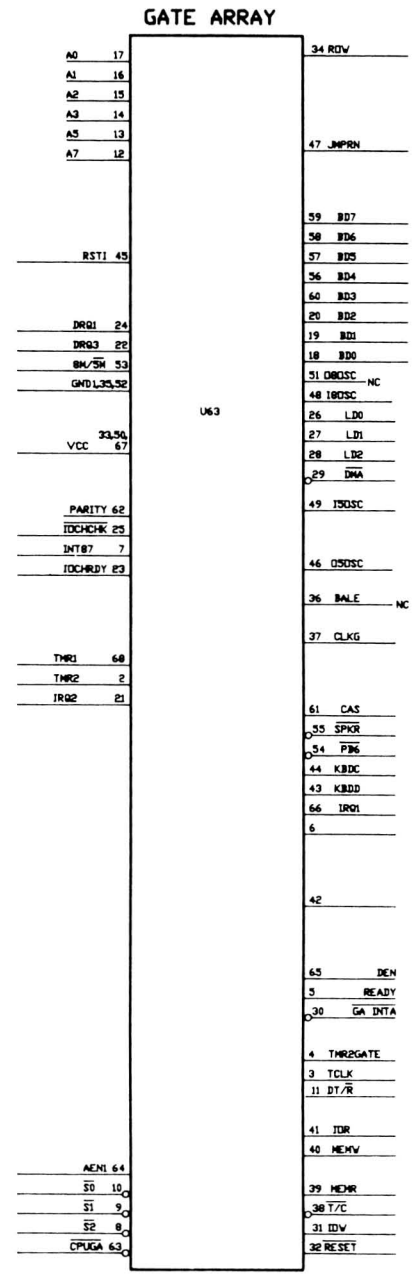


Figure 14-13. Gate Array Pinout

The Decoder

Most of the logic in the computer is contained in three devices: the gate array, the monitor ROM, and the decoder. The gate array is described in the previous section and the monitor ROM is described in Chapters 4 and 6.

The decoder is a custom device that replaces many discrete components normally required to support the CPU. It generates most of the chip select signals including those required for access to video memory. There are no programmable functions on this chip.

The decoder's pinout is illustrated in Figure 14-14 and described in Table 14-17.

Table 14-17. Decoder Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1 - 2	JMP6 - JMP7	These signals are used to select the power-up character font used by the system. They are permanently pulled high.
3 - 7	A19 - A15	Address line 19 - address line 15 (reverse order). These lines, along with address lines 3 - 9, are used for addressing the input/output ports and selection of the various CS and RAS lines generated by this decoder.
8 - 14	A9 - A3	Address line 9 - address line 3 (reverse order). These lines, along with address lines 15 - 19, are used for addressing the input/output ports and selection of the various CS and RAS lines generated by this decoder.
15	$\overline{\text{DACK0}}$	DMA acknowledge. This input line informs the decoder that a DMA transfer is taking place.
16	MEM	Memory. This input from the gate array goes active during memory input/output.
17	ROW	Row. This input from the gate array selects the address row for the decoder and is used for timing of the RAS signals.
18	$\overline{\text{IOW}}$	Input/output write. This active low input from the gate array is used to indicate a port address write operation.
19	$\overline{\text{S2F}}$	This input is generated from the S2 signal from the CPU and is used for timing CS and memory operations.
20	JMP	This input can be tied high, low, or float, depending upon the state of JP6 and JP7. Used to select LPT1 or LPT2 for the programmable peripheral interface. Tied low to always select LPT1.
21	GND	Ground.
22	DEN	Data enable. This input from the gate array is used to time the data input/output with the rest of the system.
23	$\overline{\text{CPUGA}}$	CPU gate array. Chip select signal for input/output addresses of the gate array.
24	$\overline{\text{GCPU}}$	Gate CPU. Chip select signal for the bidirectional address/data buffers. Gates data to and from the processor's address/data bus.
25	$\overline{\text{CSLCDC}}$	LCD controller chip select. Chip select signal for the video controller.
26	$\overline{\text{MODEM}}$	Modem. Chip select signal for the internal modem.
27	$\overline{\text{CSCOM}}$	Communications chip select. Chip select signal for the 8570.
28	$\overline{\text{CS765}}$	Floppy disk controller chip select. Chip select signal for the floppy disk controller.
29	$\overline{\text{PPI}}$	Programmable peripheral interface. Chip select signal for the programmable peripheral interface IC.
30	$\overline{\text{LED}}$	LED. Selects LCD or CRT operation.
31	$\overline{\text{TMRCS}}$	Timer chip select. Chip select signal for the 82C54 timer.
32	$\overline{\text{INTRCS}}$	Interrupt chip select. Chip select signal for the 82C59 interrupt controller.
33	$\overline{\text{ROM}}$	ROM. Chip select signal for the Monitor program ROM.
34	$\overline{\text{XRAM}}$	Scratchpad memory. Chip select signal for scratchpad memory.
35	$\overline{\text{VRAM}}$	Video memory. Chip select signal for video memory.

Table 14-17 (Continued). Decoder Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
36	$\overline{MA8}$	Memory address line 8. This line is used to address 256 kilobit devices.
37	$\overline{RAS3}$	Row address strobe 3. This line is used for refresh timing of the second memory expansion board.
38	$\overline{RAS2}$	Row address strobe 2. This line is used for refresh timing of the first memory expansion board.
39	$\overline{RAS1}$	Row address strobe 1. This line is used for refresh timing of bank 1 of memory.
40	$\overline{RAS0}$	Row address strobe 0. This line is used for refresh timing of bank 0 of memory.
41	$\overline{GM}$	Gate memory. Chip select signal for the bidirectional data buffer.
42	Vcc	+5 VDC power supply.

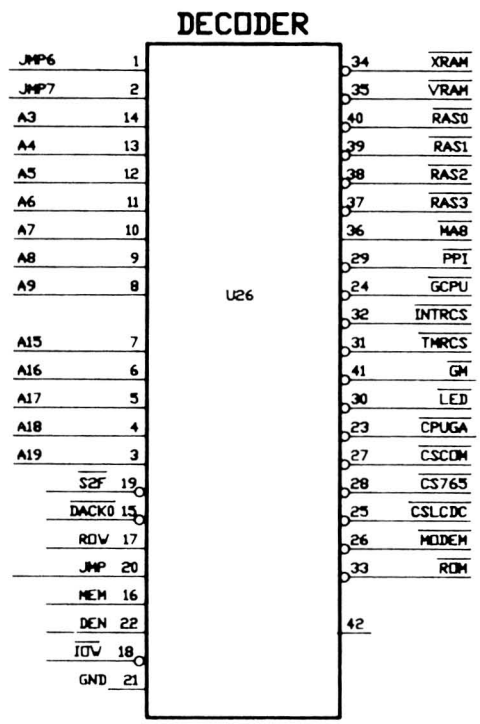


Figure 14-14. Decoder Pinout

System Bus Control Gate Array

The System Bus Controller provides in one package the necessary interface circuitry to replace numerous buffers and latches required in the computer. It provides functions such as data bus buffering, address latching, DMA addressing, and address decoding. This greatly reduces the number of discrete circuits required in the computer and reduces the overall system power requirements.

Memory Circuits

This section of the chapter describes how the memory circuits of this computer operate. A simplified block diagram is illustrated in Figure 14-15.

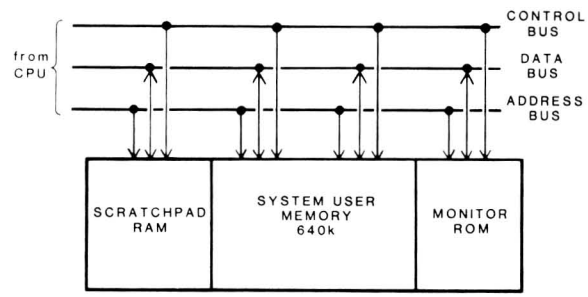


Figure 14-15. System Memory Block Diagram

The memory of this computer is organized into three blocks: Monitor program ROM, scratchpad RAM, and user memory. The basic system memory map is shown in Figure 14-16.

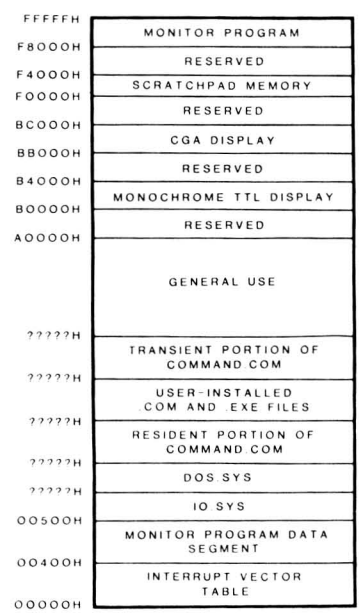


Figure 14-16. System Memory Map

System Memory

Monitor program ROM — The Monitor program ROM is a read-only memory device that contains the Monitor program, described in Chapter 7 of this manual. It contains the initial self-tests, user-executed tests, video commands, disk boot routines, a machine language debugger, and the initialization routines to set up the BIOS.

Scratchpad RAM — The scratchpad memory is used by the Monitor program to temporarily store intermediate information used by the CPU and BIOS. It is located from F0000H to F3FFFH in upper memory (above the video memory).

User memory — User memory is divided into four parts: two banks of 64K and two banks of 256K. The 64K banks are supported by two 64 kilobit x 4 devices each. The 256K banks consist of eight 256 kilobit x 1 devices each. This provides a total installed user memory capacity of 640K.

Operation

A memory cycle can be one of three types: memory read, memory write, or refresh. Any one type can be requested by more than one device. Contention and timing, along with row and column address selection, is handled by the address logic.

Refresh for a particular address is automatic whenever that location is read. During refresh cycles, all rows in all banks are read simultaneously through the RAS and CAS signals, automatically refreshing the memory. By multiplexing the address lines, only eight (for 64 kilobit devices) or nine (for 256 kilobit devices) address lines are required to read all of memory.

When a memory location is to be addressed, lines A0 - A7 are multiplexed onto the MA0 - MA7 lines followed by lines A8 - A15. The sixteen lines provide addressing for 64 kilobit devices. To address 256 kilobit devices, two additional lines are required. This is handled by the decoder, which supplies MA8. Row and column address strobes select the bank of memory to be addressed. RAS0 activates the first 64K bank of memory; RAS1 activates the second 64K bank, RAS2 activates the first optional 256K bank, and RAS3 activates the second optional 256K bank. CAS0 is assigned to the two 64K banks, CAS1 is assigned to the first 256K bank, and CAS2 is assigned to the second 256K bank.

The memory banks first latch the low-order group of bits, then the high-order group, and then combine them to form a 16- or 18-bit address word. The following briefly describes the operation of the row and column address strobe signals.

First, the row address strobe signal allows the memory ICs to latch the eight least-significant address bits from the memory address bus. A few hundred nanoseconds later, the column address strobe signal latches the eight most-significant bits from the memory address bus. For 256 kilobit devices, one additional address bit is latched during the row address strobe and column address strobe time frames allowing a full 16- or 18-bit address word to be developed.

Buffered Data

User memory makes use of a three-state, 8-bit bidirectional data buffer to interface the system address/data bus with the buffered data bus. When a read cycle is in progress, the buffer reads the data from the buffered data bus that obtained the data from the selected memory location and places it on the system address/data bus. During the write cycle, the buffer takes the data from the address/data bus and places it on the buffered data bus to be written to the selected memory location. If neither a read nor a write operation is requested. During memory refresh cycles, the buffer is inactive (data does not flow in either direction). DMA data transfers are handled by the gate array and a different set of data buffers. Refer to the discussion of DMA operation earlier in this chapter.

Optional EMS Memory

Up to 1 megabyte of EMS (Expanded Memory Specification) RAM can be added to this computer. This memory is useful only to programs that support it. The program must instruct the hardware to implement the specific bank switching. The Lotus/Intel/Microsoft Expanded Memory Specification (LIM EMS) attempts to standardize the bank switching scheme. The LIM EMS contains numerous examples that an experienced assembly language programmer can interpret. In this way, different programs can be written to make use of expanded memory. The specification may be obtained free of charge from Intel Corporation by calling 1-800-538-3373.

EMS memory is page-mapped. This means that a small area of memory acts as a window through

which a larger area of memory can be accessed. This window is 64K wide and is composed of four 16K blocks of memory, referred to as page frames. As the CPU addresses the memory range of one of the page frames, the EMS memory page pointer remains set to that frame. If the page frame changes or the page is turned on or off, an 8-bit value is written to an I/O port to select the next correct frame. The value placed in the port moves a specified 16K block of expanded memory page to the page frame.

Data is transferred between the CPU and EMS memory in the same way that date is interfaced to system memory. The same data transceivers and control signals are active.

Memory refresh cycles also occur for expanded memory in the same way as for system memory. Memory refresh is handled during the appropriate processor cycle and addressing is manipulated so the refresh cycle includes expanded memory.

The Keyboard

The keyboard in this computer is strobe-scanned and controlled by a keyboard processor that cannot be programmed. Refer to Figure 14-17 for the block diagram.

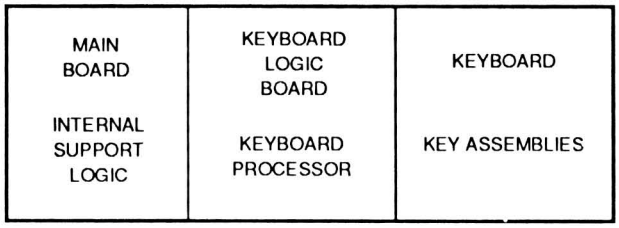


Figure 14-17. Keyboard Block Diagram

The keyboard circuits are on two boards in this computer: the main board, which contains the gate array, initial support logic, and the keyboard processor; and the keyboard itself, which contains the keyswitch matrix.

Gate array support — The gate array supports the keyboard through the internal programmable peripheral interface. The interface sets up and controls three lines, keyboard clock, data, and reset lines, connected to the keyboard processor.

Keyboard processor — The keyboard processor is a dedicated microprocessor that contains 4K of preprogrammed ROM and 128 bytes of RAM. The processor strobe-scans the keyboard and is programmed to handle N-key rollover.

Keyswitch matrix — The 78 keys of the keyboard are configured in a matrix consisting of 10 scan lines and 8 read lines. Each keyswitch is placed in the matrix with a diode to prevent false scan codes from being generated when more than one key is pressed.

The keyboard is not programmable, except through the operating system or software. The ROM in the processor always returns the key scan codes reported in Chapter 8. Before issuing the keyboard interrupt, the up or down code is deposited in port A of the programmable peripheral interface, which is addressable at port 060H. The keyboard interrupt (refer to Chapter 8) takes the up/down code and converts it to the keyboard scan codes seen when you run the keyboard test of the built-in test program (refer to Chapter 19).

Because the interrupts can be programmed to produce any desired code or action from any key, the results produced by the keyboard for a particular applications program is independent of the computer.

Chapter 15
Video Circuits

This chapter describes the video circuits, including the V6355 video controller. The following information provides you with the basis for programming the very complex video section of the computer, although specific assembly and machine language instructions are not provided.

A simplified block diagram of the video circuits is illustrated in Figure 15-1.

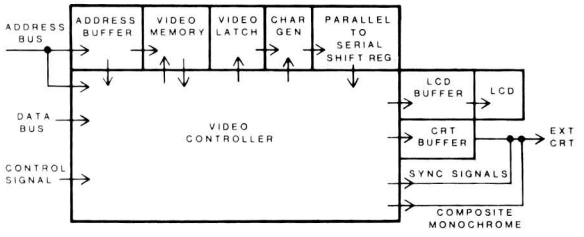


Figure 15-1. Video Block Diagram

The video circuit is fed by the address bus, the data bus, and a number of control signals generated by the processor and support circuits. The address bus selects locations in video memory, while the controller supports both static and dynamic memory devices. No external refresh circuits need to be provided.

Address Buffer — A pair of three-state buffers interface the computer's address bus with the video memory address bus. This allows the video memory to be addressed by either the CPU or the video controller for reading from or writing to specific video memory locations.

Video Controller — The V6355 controller is a 100-pin device, described later, that provides the proper address, data, and drive signals for displaying a video signal on either an LCD device or a CRT device. It retains the programmability and flexibility of the 6845 CRTC, the standard device used in PC-compatible color graphics (CGA) CRT applications, and adds additional unique features required for LCD applications.

Video Memory — While the V6355 can support either static or dynamic memory devices, this computer uses static RAM. The video memory is 16K in two 16 kilobit x 4 devices. The base memory address is at B8000H, standard for CGA video memory.

Video Latch — The video latch holds video memory long enough for the character generator to address a particular character pattern.

Character Generator — The character generator is a programmed ROM that produces the character patterns that are sent in parallel form to the parallel-to-serial shift register for conversion to a serial dot pattern.

Parallel-to-Serial Shift Register — This device takes the parallel data produced by the character generator and shifts it out as a clocked serial data stream to the V6355 controller.

LCD Buffer — The LCD buffer is handled by 12 individual driver/buffers in two devices. The second device contains the four driver/buffers for the CRT.

LCD — The LCD is organized as a pair of 640 x 100 matrices that are driven by 16 segment and two common drivers. The two common drivers share the responsibility for driving all 200 rows, while the 16 segment drivers handle 80 columns of 100 rows each.

CRT Buffer — The CRT buffer is handled by four individual driver/buffers in a device that also supports four driver/buffers for the LCD.

External CRT — The external CRT that is supported by this computer should have the capabilities of accepting RGB and I (intensity) signals and a horizontal resolution of 640 bits. Vertical resolution should be at least 200 lines.

The block diagram also illustrates the development of the composite monochrome signal for driving monochrome CRT monitors. This signal comes directly from the V6355 controller and is driven by a small-signal transistor circuit.

The V6355 Video Controller

The V6355 is a liquid crystal display and cathode-ray tube controller that provides the signals necessary to produce video on the built-in LCD, and composite monochrome and RGB/intensity video signals for use with external raster-scan CRT video monitors.

New and expanded registers in the V6355 provide a sprite, expanded color capabilities, and LCD/CRT options that are not available in CGA designs. Full CGA and 6845 compatibility is maintained by the V6355 and support circuits. All video software that runs on CGA systems will run without need for modification on this computer. However, software that is written to take advantage of the V6355's capabilities will not necessarily perform as expected on CGA systems.

Refer to Figure 15-2 for a simplified block diagram of the video controller. The control section handles decoding of the control signals and communication between the internal registers and the system data bus. The CGA/CRT-LCD support circuits process the video memory data with information from the internal registers to produce the CRT and LCD video signals. The sync generator uses register information to produce the separate sync signals and combine them for the composite monochrome signal.

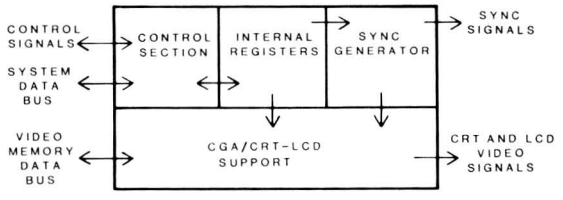


Figure 15-2. V6355 Controller Block Diagram

Controller Input/Output Port Addresses

The port addresses used by the V6355 video controller in this computer are identified in Table 15-1.

Table 15-1. Video Input/Output Port Addresses

ADDRESS	DESCRIPTION
3D0H	6845 pointer address port
3D1H	6845 data port
3D2H	6845 pointer address port
3D3H	6845 data port

Table 15-1 (Continued). Video Input/Output Port Addresses

ADDRESS	DESCRIPTION
3D4H	6845 pointer address port
3D5H	6845 data port
3D6H	6845 pointer address port
3D7H	6845 data port
3D8H	Video mode select port
3D9H	Color select port
3DAH	Status port
3DBH	Light pen clear port (not used)
3DCH	Light pen preset port (not used)
3DDH	Register bank pointer address port
3DEH	Register bank data port
3DFH	Page register port (not used)

The 6845 ports are duplicated through "don t care" bits. Ports 3D0H, 3D2H, 3D4H, and 3D6H are all the same 6845 pointer address port. Likewise, 3D1H, 3D3H, 3D5H, and 3D7H are all the same 6845 data port. This arrangement is identical to the 6845 CRT controller. The functions of ports 3D8H and 3DAH are expanded, and ports 3DDH, 3DEH, and 3DFH are new to the operation of the V6355 over the 6845.

6845 Pointer Address Port

This write-only, 5-bit register serves as a pointer to the 18 6845 data registers. To address one of the 6845 data registers, you must first write the register number (00H - 11H) into this register by executing an OUT instruction to one of the four 6845 pointer address ports (3D0H, 3D2H, 3D4H, or 3D6H). Then either read from (by executing an IN instruction) or write to (by executing an OUT instruction) to one of the four 6845 data ports (3D1H, 3D3H, 3D5H, or 3D7H).

6845 Data Port

The data port serves as the read/write port to the 6845 registers of the V6355. These registers are described in Table 15-2. The initialization value for each register is described in Chapter 12.

Table 15-2. 6845 Data Registers

REGISTER	NUMBER	DESCRIPTION
R0	00H	Horizontal total register.
R1	01H	Horizontal display register.

Table 15-2 (Continued). 6845 Data Registers

REGISTER	NUMBER	DESCRIPTION
R2	02H	Horizontal sync position register.
R3	03H	Sync width register.
R4	04H	Vertical total register.
R5	05H	Vertical total adjust register.
R6	06H	Vertical display register.
R7	07H	Vertical sync position register.
R8	08H	Interlace/skew register.
R9	09H	Maximum scan line register.
R10	0AH	Cursor start line register.
R11	0BH	Cursor end line register.
R12	0CH	High-order byte start address register.
R13	0DH	Low-order byte start address register.
R14	0EH	High-order byte cursor address register.
R15	0FH	Low-order byte cursor address register.
R16	10H	High-order byte light-pen register or mouse X counter register (neither is used in this computer).
R17	11H	Low-order byte light-pen register or mouse Y counter register (neither is used in this computer).

Horizontal total register (R0) — This 8-bit, write-only register is used to set the time between the starting points of character rows. This time, defined in character clock periods, includes all characters in a row, the delay before the horizontal sync pulse, the width of the sync pulse, and the retrace time. Since the first character is 0 (zero), this value will be the number of character clock signals desired minus 1. A value of 112 would represent 113 character positions.

Horizontal displayed register (R1) — This 8-bit, write-only register is used to select the number of characters in each row that will be displayed on the screen. In normal operation, the register will contain a lower value than the horizontal total register (R0).

Horizontal sync position register (R2) — This 8-bit, write-only register is used to determine the position of the sync pulse in each scan line. Any value greater than the horizontal displayed register (R1) and less than the horizontal total register (R0) can be used. When the value of R2 is increased, the display will shift to the left; decreasing this value shifts the display to the right.

Sync width register (R3) — This 8-bit, write-only register is used to set the width of the horizontal and vertical sync pulses. The width of the horizontal sync is controlled by the lower four bits of the pulse width register and the width of the vertical sync is controlled by the upper four bits of the pulse width register. Increasing the value lengthens the width.

NOTE: The sum of the values in register R1 and R3 must be equal to or less than the value in register R0.

Vertical total register (R4) — This 7-bit, write-only register is used to determine the refresh rate of the video monitor and vertical retrace timing. Its value is expressed in character rows; each is normally eight scan lines. The values for the various video modes are provided in Table 12-5 in Chapter 12.

Vertical total adjust register (R5) — This 5-bit, write-only register is used to lengthen the vertical total time to account for the scan lines remaining when the value for register R4 is determined. This value is also expressed in scan lines and is provided in Table 12-5 in Chapter 12.

Vertical displayed register (R6) — This 7-bit, write-only register is used to determine the number of character rows that will be displayed. This value must be equal to or less than the vertical total register (R4).

Vertical position register (R7) — This 7-bit, write-only register is used to control the position of the vertical sync. The value is expressed in character rows and must be equal to or less than the value in the vertical total register (R4). A lower value will move the display down; a higher value will move the display up.

Interlace/skew register (R8) — This 8-bit, write-only register is used to determine interlace operation (on or off) and the skew (offset or delay) of the characters and cursor. CGA operation lacks sufficient memory to operate the computer in an interlaced mode, and none of the PC-compatible modes use an interlaced mode. Therefore, always operate this video in a non-interlaced mode. Table 15-3 defines the functions of bits 0 and 1; Table 15-4 defines the

functions of bits 4 and 5; and Table 15-5 defines the functions of bits 6 and 7; bits 2 and 3 are not used.

Table 15-3. 6845 Register R8, Bits 0 and 1

BIT 0	BIT 1	FUNCTION
0	0	Normal non-interlaced operation
0	1	Normal non-interlaced operation
1	0	Interlaced sync mode operation
1	1	Interlaced sync and video mode operation

Table 15-4. 6845 Register R8, Bits 4 and 5

BIT 4	BIT 5	FUNCTION
0	0	Normal display; no character skew.
0	1	Skewed display; characters are offset one character position.
1	0	Skewed display; characters are offset two character positions.
1	1	Illegal; do not use.

Table 15-5. 6845 Register R8, Bits 6 and 7

BIT 6	BIT 7	FUNCTION
0	0	Normal display; no cursor skew.
0	1	Skewed cursor; the cursor is offset one character position.
1	0	Skewed cursor; the cursor is offset two character positions.
1	1	Illegal; do not use.

Maximum scan line address register (R9) — This 5-bit, write-only register is used to control the number of scan lines in each character row. The first scan line is 0 so this value will be 1 less than the number of scan lines. Normally, eight scan lines are used for a character row; therefore, the value will be 7.

Cursor start register (R10) — This 7-bit, write-only register is used to define the first scan line of the cursor within the character field and whether the cursor is on or off.

The lower five bits define the first scan line. The scan lines are numbered from the top, starting with zero, to the bottom, ending with seven in the modes supported by this computer. Normally, the value in this register is 6.

Bit 5 must be clear (0) for the cursor to be on (blinking). Bit 6 may be set (1) or clear (0). The 6845 defines bits 5 and 6 to control the blinking (steady, 2 Hz, or 4 Hz). Apparently the V6355 does not support this function.

Cursor end register (R11) — This 5-bit, write-only register is used to define the last scan line of the cursor within the character field. Normally, this value is 7.

Start address registers (R12 and R13) — These two read/write registers contain the first displayable address in video memory and are used for scrolling the display. Register R12 contains the 6-bit, high-order (most significant byte) address, and register R13 contains the 8-bit, low-order (least significant byte) address.

Cursor registers (R14 and R15) — These two read/write registers contain the address of the cursor in video memory. This allows the original cursor location to be retained even though the hardware may scroll or page the display. Register R14 contains the 6-bit, high-order (most significant byte) address and register R15 contains the 8-bit, low-order (least significant byte) address.

Light pen or mouse registers (R16 and R17) — These two read-only registers contain the video address captured from a light pen or the X and Y coordinates of a mouse. The mouse coordinates are relative to the last time the mouse coordinates were cleared. Neither of these applications are supported in this computer.

Video Mode Select Port

The video mode select port is actually a single 8-bit, read/write register. Its port address is 3D8H. Even though full CGA compatibility is maintained in this register, it has been expanded over the register defined in CGA designs and offers more flexibility. The bits of this register are somewhat interactive and are defined in Table 15-6 and Table 15-7. For more information, read "Modes of Operation," later in this chapter.

Table 15-6. Video Mode Select Port Register

BIT	DESCRIPTION
0	Alphanumeric width: clear (0) = 40 x 25, set (1) = 80 x 25. Bit 1 must be clear (0). Bit 5 determines color/blink mode.
1	Alphanumeric/graphics mode: clear (0) = alphanumeric, set (1) = graphics. The width of the alphanumeric mode is determined by bit 0. The resolution of the graphics mode is determined by bits 4 and 6.
2	Color/monochrome composite mode: clear (0) = color, set (1) = monochrome composite.
3	Video enable/disable: clear (0) = disable video, set (1) = enable video. Video should be disabled (turned off) during mode changes to prevent an unprofessional display that results from synchronization changes that occur as modes are changed.
4	640 x 200 resolution color or monochrome/320 x 200 resolution. Refer to Table 15-7.
5	Alphanumeric color/blink mode: clear (0) = 16-color background enabled, set (1) = blinking attribute enabled.
6	Color/Monochrome or 160 x 200/640 x 200 resolution. Refer to Table 15-7. This bit can be locked by software to prevent accidental modification by CGA software.
7	Active/standby: clear (0) = active, set (1) = standby (no output). This bit can be locked by software to prevent accidental modification by CGA software.

Table 15-7. Video Mode Select Resolution

BIT 4	BIT 6	DESCRIPTION
0	0	320 x 200 resolution
0	1	160 x 200 resolution
1	0	640 x 200 monochrome only resolution
1	1	640 x 200 16-color only resolution

Color Select Port

The color select port is a 6-bit, write-only register that establishes certain colors used by the various modes. Its port address is 3D9H. Because the colors selected can be modified by the V6355, the colors are represented by R, G, B, and I in Table 15-8, which describes the function of each bit of this register. Table 15-9 defines the two color palettes that can be used by the 320 x 200 graphics mode.

Table 15-8. Color Select Port Register

BIT	DESCRIPTION
0	B color, default is blue. Affects border and background color in 320 x 200 graphics mode, foreground color in 640 x 200 graphics mode, and border color in text modes. Bits 0, 1, 2, and 3 are combined to produce one of 16 programmable colors.
1	G color, default is green. Affects border and background color in 320 x 200 graphics mode, foreground color in 640 x 200 graphics mode, and border color in text modes. Bits 0, 1, 2, and 3 are combined to produce one of 16 programmable colors.
2	R color, default is red. Affects border and background color in 320 x 200 graphics mode, foreground color in 640 x 200 graphics mode, and border color in text modes. Bits 0, 1, 2, and 3 are combined to produce one of 16 programmable colors.
3	I signal, intensifies the color levels produced by the R, G, and B colors. Affects border and background colors in 320 x 200 graphics mode, foreground color in 640 x 200 graphics mode, and border color in text modes. Bits 0, 1, 2, and 3 are combined to produce one of 16 programmable colors.
4	When set (1), intensifies the foreground colors in the 320 x 200 graphics mode. Does not affect the text or 640 x 200 graphics modes.
5	320 x 200 graphics mode color palette selection: set (1) = palette 1, clear (0) = palette 2. Refer to Table 15-9 for palette definition. Does not affect the other modes.

Table 15-9. Palette Definitions

COLOR	PALETTE 1	PALETTE 2
0	See note 1	See note 1
1	G + B (cyan)	G (green)
2	R + B (magenta)	R (red)
3	R + G + B (white)	R + G (brown)

NOTES: 1. If color 0 is called for, the color defined by bits 0 - 3 of the color select port register will be the color used.
2. The colors shown in parenthesis are the default colors produced by the V6355 for CGA operation. The 16 color palette registers can be modified to produce other colors, which will modify the colors in these palettes.

Status Port

The status port is actually a 5-bit, read-only register that reports the status of various features of the controller. Its address is at 3DAH. Table 15-10 describes what each bit means.

Table 15-10. Status Port Register Report

BIT	DESCRIPTION
0	Active display: set (1) = active display, clear (0) = vertical retrace (sync) period.
1	Not used or connected in this computer. In other hardware applications this bit can report that the light pen switch has been activated or that mouse switch 1 is closed.
2	Not used or connected in this computer. In other hardware applications, this bit can report the current status the light pen switch or that mouse switch 0 is closed.
3	Set (1) during the active vertical retrace period. Clear (0) during all other times. If the monochrome video mode is selected, which is not implemented in this computer, this bit will reflect the video dot stream, the same as bit 4.
4	This bit follows the video dot stream for the signal indicated by the condition of bit 0 and bit 1 of port 3DDH, the register bank pointer address port. Refer to Table 15-11 for the signal selected.

Table 15-11. Signal Selected by the Register Bank Pointer Address Port

BIT 0	BIT 1	SIGNAL
0	0	B color
1	0	G color
0	1	R color
1	1	I signal

NOTE: The status port register bit 4 is designed to test the video mode select port, color select port, and all write-only registers.

Light Pen Ports

The two light pen ports, addressed at 3DBH and 3DCH, are not implemented in this computer. The ports are actually write-only registers that are used to verify that the light pen trigger flip-flop and status read functions are operating correctly. Since light-pens are not supported by this computer, there is no light pen flip-flop and hence, these ports are meaningless.

Register Bank Pointer Address Port

The expanded register bank allows many new features to PC-compatible video systems to be implemented without sacrificing the features and capabilities of the 6845. Like the 6845 registers, the registers in the expanded register bank use one port as a register pointer and a second port as a data input/output port. The pointer port address is 3DDH.

Register Bank Data Port

The register bank data port serves as the read/write port to the V6355 expanded register bank. These registers are described in Table 15-12.

Table 15-12. Register Bank Expanded Registers

REGISTER	ADDRESS	DESCRIPTION
R0 - R63	00H - 3FH	Sprite pattern registers
R64 - R65	40H - 41H	Color palette 0 registers
R66 - R67	42H - 43H	Color palette 1 registers
R68 - R69	44H - 45H	Color palette 2 registers
R70 - R71	46H - 47H	Color palette 3 registers
R72 - R73	48H - 49H	Color palette 4 registers
R74 - R75	4AH - 4BH	Color palette 5 registers
R76 - R77	4CH - 4DH	Color palette 6 registers
R78 - R79	4EH - 4FH	Color palette 7 registers
R80 - R81	50H - 51H	Color palette 8 registers
R82 - R83	52H - 53H	Color palette 9 registers
R84 - R85	54H - 55H	Color palette 10 registers
R86 - R87	56H - 57H	Color palette 11 registers
R88 - R89	58H - 59H	Color palette 12 registers
R90 - R91	5AH - 5BH	Color palette 13 registers
R92 - R93	5CH - 5DH	Color palette 14 registers
R94 - R95	5EH - 5FH	Color palette 15 registers
R96 - R97	60H - 61H	Sprite horizontal location registers
R98 - R99	62H - 63H	Sprite vertical location registers
R100	64H	Test/sprite control register
R101	65H	Monitor control register
R102	66H	MDA/LCD control register
R103	67H	Configuration mode register
R104	68H	Sprite color selection register
R105	69H	Control data register

Sprite pattern registers — The video controller has the capability to define and display a sprite, which is formed from two patterns and the screen. One of two patterns is ANDed with the screen and the other pattern is XORed with the results of the ANDing process to produce the actual sprite. The pattern registers are organized as two sets of 16 x 16-dot matrices. The dot pattern organization is shown in Table 15-13. The location is the lower right-hand coordinate of the 16 x 16 matrix. Location, control, and color information is stored in registers R96 - R99, R100, and R104, respectively. The most common use for a sprite is a mouse or graphics tablet pointer. Pointers created by mouse or graphics tablet drivers are created by software and do not use this controller's hardware capabilities. Since the sprite capability is unique to this controller and not common to PC-compatible systems, it is unlikely that the sprite features will be implemented, except in a software application that has a driver specifically designed for this computer.

Table 15-13. Sprite Pattern Register Organization

AND PATTERN		XOR PATTERN	
D7 D0	D7 D0	D7 D0	D7 D0
R00	R01	R32	R33
R02	R03	R34	R35
R04	R05	R36	R37
R06	R07	R38	R39
R08	R09	R40	R41
R10	R11	R42	R43
R12	R13	R44	R45
R14	R15	R46	R47
R16	R17	R48	R49
R18	R19	R50	R51
R20	R21	R52	R53
R22	R23	R54	R55
R24	R25	R56	R57
R26	R27	R58	R59
R28	R29	R60	R61
R30	R31	R62	R63

Colors used in the sprite are determined by colors established by the sprite color selection register (R104).

Color palette registers — Each of the sixteen color registers are created from two 8-bit registers and use nine bits to express one of 512 possible colors. The registers are preset to display the standard CGA colors. The purpose of each bit is described in Table 15-14.

Table 15-14. Color Palette Registers

REGISTER	BIT	DESCRIPTION
0	0	Red bit 0
0	1	Red bit 1
0	2	Red bit 2
0	3 - 7	Not used
1	0	Blue bit 0
1	1	Blue bit 1
1	2	Blue bit 2
1	3	Not used
1	4	Green bit 0
1	5	Green bit 1
1	6	Green bit 2
1	7	Not used

Table 15-15 describes the relationship between the normal R, G, B, and I signals and the signals produced by the color palette registers. The colors described in the table are the default CGA colors.

Table 15-15. Default CGA Colors

R	G	B	I	R0	R1	R2	G0	G1	G2	B0	B1	B2	COLOR
0	0	0	0	0	0	0	0	0	0	0	0	0	Black
0	0	1	0	0	0	0	0	0	0	0	0	1	Blue
0	1	0	0	0	0	0	0	0	1	0	0	0	Green
0	1	1	0	0	0	0	0	0	1	0	0	1	Cyan
1	0	0	0	0	0	1	0	0	0	0	0	0	Red
1	0	1	0	0	0	1	0	0	0	0	0	1	Magenta
1	1	0	0	0	0	1	0	0	1	0	0	0	Brown
1	1	1	0	0	0	1	0	0	1	0	0	1	White
0	0	0	1	0	0	0	0	0	0	0	0	0	Gray
0	0	1	1	0	0	0	0	0	0	0	0	1	Light blue
0	1	0	1	0	0	0	0	0	1	0	0	0	Light green
0	1	1	1	0	0	0	0	0	1	0	0	1	Light cyan
1	0	0	1	0	0	1	0	0	0	0	0	0	Light red
1	0	1	1	0	0	1	0	0	0	0	0	1	Light magenta
1	1	0	1	0	0	1	0	0	1	0	0	1	Yellow
1	1	1	1	0	0	1	0	0	1	0	0	1	Intense white

The mode and pixel information stored in video memory determines the color to display. The three bits for each color are used to determine the equivalent register. In the register, the three bits for each color determines the brightness level of the color displayed. By combining the levels of the three colors and the intensity bit, it is possible to select one of 512 different colors for each of the 16 color palettes, thus altering the standard CGA colors. However, programs that alter and make use of the 16 color palettes to produce different colors than the standard CGA colors will not necessarily work as expected on computer video systems that do not use the V6355.

Sprite location registers — The sprite's display location is determined by four registers, two each for the horizontal and the vertical coordinate on the screen. The location is pixel-oriented and offset 16 units up and 16 units to the left. Placing a value of 0 in all four registers establishes the location at 0,0 on the screen. Since the sprite is offset upward 16 units and to the left 16 units, it will not display. To display the full sprite pattern in the upper-left corner, you would have to establish 16,16 as the location on a 320 x 200 screen.

The horizontal location is established by a 9-bit value in R96 and R97 (60H and 61H). The least significant byte in in R97. R96 contains the ninth bit of the coordinate. The vertical location is established by an 8-bit value in R99 (63H). Register R98 is not used.

Test/sprite control register — This 8-bit register provides some control over the sprite and the screen. It also provides a test mode that is not implemented in this computer. Table 15-16 describes the function of each bit in the register.

Table 15-16. Test/Sprite Control Register

BIT	DESCRIPTION
0	Sprite blink enable. When set (1), this bit enables the blink feature of the sprite. The rate is approximately 2 Hz.
1	Sprite 1 enable. When set (1), this bit enables the AND sprite pattern.
2	Sprite 2 enable. When set (1), this bit enables the XOR sprite pattern.
3 - 5	Screen offset 0 - 2. These three bits allow software-controlled smooth scrolling by offsetting the screen upward. The value in these three bits determines the number of scan lines the screen is shifted. The screen offset up is determined by the value in these three bits.
6 - 7	Test 0 - 1. Not used, leave clear (0).

Monitor control register — This 8-bit register defines the video output of the controller. This register is preset to 01H by the RESET line. Each bit is described in Table 15-17.

Table 15-17. Monitor Control Register

BIT	DESCRIPTION
0 - 1	Vertical scan line number. The values in these two bits establish the number of vertical scan lines to be displayed: 0 = 192, 1 = 200, 2 = 204, and 3 = 64. Since this is a PC-compatible display, always use 200 scan lines.
2	Horizontal pixel number. When set (1), the controller will display 512 (single-dot pattern) or 256 (double-dot pattern) pixels. When clear (0), the controller will display 640 (single-dot pattern) or 320 (double-dot pattern) pixels. PC-compatible displays use the 640/320 pixel setting.
3	PAL/SECAM - NTSC. If set (1), this bit selects PAL/SECAM video parameters for color composite signals. Clear (0) select NTSC video parameters for color composite signals. The color composite signals are not used in this computer; leave this bit clear (0).
4	Monitor type. If set (1), this bit selects a TTL-compatible monochrome output (not used in this computer). If clear, this bit selects CGA output.
5	LCD/CRT select. If set (1), the output goes to the LCD. If clear (0), the output goes to the RGB connector.
6	Static/dynamic RAM. If set (1), the video memory is static RAM. If clear (0), the video memory is dynamic. This computer uses dynamic video memory.
7	Mouse/light pen. Neither is used in this computer even though the V6355 can directly support a mouse or light pen as a pointing device.

MDA/LCD control register — This 8-bit register defines the monochrome display (TTL-compatible) and LCD parameters. Since the MDA output is not functional in this computer, those parameters are not needed. Since the values placed in this register configure the V6355 for use with specific hardware devices, do not modify any of the values in this register. Table 15-18 describes each bit in the register.

Table 15-18. MDA/LCD Control Register

BIT	DESCRIPTION
0 - 1	Raster adjust 0 - 1. These two bits adjust the vertical display position for the upper half of the LCD.

Table 15-18 (Continued). MDA/LCD Control Register

BIT	DESCRIPTION
2 - 3	LCD driver type. These two bits select the LCD driver type: type 1 is a dual one-bit serial output, type 2 requires a dual four-bit parallel output, and type 3 requires a dual four-bit intensity output.
4 - 5	LCD driver shift clock. These two bits establish the shift clock frequency for the LCD. Type 2 requires a different frequency than type 0 and type 1. Refer to Table 15-19 for the relationship between type 2, these two bits, and the selected shift clock frequency.
6	MDA gray scale modification. When set (1), this bit disables the 16-level gray scale for TTL-compatible monitors. When clear (0), this bit enables the 16-level gray scale display. Not used in this computer.
7	MDA output. When set (1), this bit enables TTL-compatible output; when clear (0), CGA output is enabled. Leave clear, the MDA feature is not enabled in this computer.

Table 15-19. LCD Driver Shift Clock

TYPE 2	BIT 5	BIT 4	SHIFT CLOCK FREQUENCY
no	0	0	2.685 MHz
no	0	1	1.34 MHz
no	1	0	5.37 MHz
no	1	1	2.68 MHz
yes	0	0	0.67 MHz
yes	0	1	0.34 MHz
yes	1	0	1.34 MHz
yes	1	1	0.67 MHz

Configuration mode register — This 8-bit register establishes additional parameters for the monitor and video memory. Table 15-20 describes the bits in this register.

Table 15-20. Configuration Mode Register

BIT	DESCRIPTION
0 - 4	The value in these bits is used to generate the enable or weight clock in the LCD mode or to adjust the horizontal display position (between -7 and +8 dots) in the CRT mode.

Table 15-20 (Continued). Configuration Mode Register

BIT	DESCRIPTION
5	When set (1), this bit selects the M signal. Hardware dependent.
6	When set (1), this bit enables the page feature for 64K video memory. Video memory is 16K in this computer, so this bit should always be clear (0). The page feature is not available in this computer.
7	When set (1), this bit enables 16-bit bus operation. This computer's processor uses an 8-bit data but, so this bit should always be clear (0).

Sprite color selection register — This 8-bit register establishes the sprite foreground color. The combination of color and intensity bits establishes the color selected from the color palette registers. Table 15-21 describes the function of the bits in this register.

Table 15-21. Sprite Color Selection Register

BIT	DESCRIPTION
0	When set (1), adds blue to the AND sprite's foreground color.
1	When set (1), adds green to the AND sprite's foreground color.
2	When set (1), adds red to the AND sprite's foreground color.
3	When set (1), adds intensity to the AND sprite's foreground color.
4	When set (1), adds blue to the XOR sprite's foreground color.
5	When set (1), adds green to the XOR sprite's foreground color.
6	When set (1), adds red to the XOR sprite's foreground color.
7	When set (1), adds intensity to the XOR sprite's foreground color.

Control data register — This 8-bit register is used to hold a control word that can be placed on the video data bus in place of normal video data.

Modes Of Operation

The video has two basic modes of operation: alphanumeric and graphics. The V6355 controller supports six modes of operation, which is established by the mode select port register. Table 15-22 describes the modes and their relationship to the values in the mode selection register.

Table 15-22. Mode Selection

MODE REGISTER BIT							
6	5	4	3	2	1	0	MODE
0	B	0	V	M	0	0	40 x 25 alphanumeric
0	B	0	V	M	0	1	80 x 25 alphanumeric
1	X	0	V	M	1	0	160 x 200 graphics
0	X	0	V	M	1	0	320 x 200 graphics
0	X	1	V	X	1	X	640 x 200 monochrome only
1	X	1	V	0	1	X	640 x 200 color only

NOTE: Refer to Table 15-23 for a definition of the symbols used in this table.

Table 15-23. Symbol Definitions

SYMBOL	DEFINITION
0	This bit is clear (0).
1	This bit is set (1).
B	When set (1), this bit enables blinking; when clear (0), this bit disables blinking.
V	When set (1), this bit enables video output; when clear (0), this bit disables video output.
M	When set (1), this bit produces monochrome output; when clear (0), this bit produces color output.
X	The value of this bit does not matter; it can be either set (1) or clear (0).

Alphanumeric Mode

In the alphanumeric mode, two display widths are available: 40 x 25 and 80 x 25. The 16K of video memory in this computer provides eight pages of 40 x 25 text or four pages of 80 x 25 text.

The controller uses a two consecutive bytes to define each character position on the screen. The starting address of video memory must be at an even-numbered location.

The first byte defines the displayable character, using an extended ASCII character set. Ninety-six of the 128 ASCII codes are displayable characters. The extended codes display the 256-character PC-compatible character set that are outside the display-

able ASCII range. The entire displayable character set is illustrated in Figure 15-3.

The second byte defines the attributes of the character. Since both monochrome and color display modes are available, the attributes that control each character are different. The color attribute bits are described in Table 15-24 and the monochrome attribute bits are described in Table 15-25.

Table 15-24. Alphanumeric Color Attribute Byte

BIT	DESCRIPTION
0	Blue foreground color bit.
1	Green foreground color bit.
2	Red foreground color bit.
3	Foreground color intensity bit.
4	Blue background color bit.
5	Green background color bit.
6	Red background color bit.
7	Blink or background color intensity bit. Refer to Table 15-22. When the blink attribute is set (1), this bit selects the characters to be blinked. When the blink attribute is clear (0), this bit is the background color intensity bit. When blink is enabled, this bit, when set (1), will blink the character at about 2 Hz; when clear (0), the character will not be blinked.

Table 15-25. Alphanumeric Monochrome Attribute Byte

	BIT							
	0	1	2	3	4	5	6	7	DESCRIPTION
B	0	0	0	I	1	1	1	Normal video
B	1	1	1	I	0	0	0	Reverse video
B	0	0	0	I	0	0	0	No display (black)
B	1	1	1	I	1	1	1	No display (white)
B	1	0	0	I	1	1	1	Normal underline if enabled

B = If set (1), the character will blink. If clear (0), the character will not blink.

I = If set (1), the character is intensified. If clear (0), the character is not intensified.

The V6355 provides sprite as well as cursor operation. The priority of the characters is sprite, cursor, text, with text being lowest display priority.

Figure 15-3. Displayable Character Set

Graphics Modes

The V6355 supports four graphic modes: 160 x 200, 320 x 200, 640 x 200 monochrome, and 640 x 200 color. The text cursor is not used, but the sprite is available in any of the modes.

160 x 200 Graphics — In this mode, each pixel occupies four dots on a line. Each pixel can display one of sixteen colors out of the palette of 512 that is defined by color palette registers in the V6355. The sprite size is a square, 16 x 16 pixels, as defined by the 320 x 200 resolution mode.

320 x 200 Graphics — In this mode, each pixel occupies two dots on a line. The pixel information is stored in video memory as two 8,000-byte banks. The addressing is illustrated in Figure 15-4. Address range B8000H - B9F3F contains color selection data for the pixels in the even-numbered scan rows (0 - 198). Address range B9FFFH - BBF3FH contains color selection data for the odd-numbered scan rows (1 - 199).

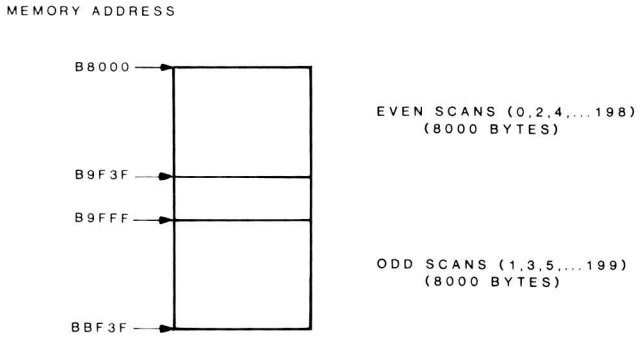


Figure 15-4. Graphics Storage Map

Each byte of memory contains the color information for four consecutive pixels on the scan row. Figure 15-5 illustrates the mapping of this color information.

Each pair can produce four combinations of numbers, causing the computer to display the foreground color according to the logic of Table 15-26. Three of the four bits selected determine which of the sixteen color palette registers is used along with the intensity bit to produce one of 512 possible colors. Bit 5 of the color select port determines which color set used. When bit 5 is clear (0), set 1 is used; when bit 5 is set (1), set 2 is used. The colors in the parenthesis are the default colors set when the system is first turned on.

7 6		5 4		3 2		1 0	
BIT 2	BIT 0	BIT1	BIT 0	BIT 1	BIT 0	BIT 1	BIT 0

Figure 15-5. Medium-Resolution Byte Mapping

640 x 200 Monochrome Graphics — In this mode, the foreground color can assume one of 512 colors, as determined by bits 0 - 3 of the color select port and the color palette registers. Since each pixel is asingle dot that can be either on or off, this results in a monochrome display. The formatting of the data is somewhat different with each bit of video memory representing one pixel on the screen. Figure 15-6 illustrates the byte-bit relationship to the screen pixels.

BIT:	7	6	5	4	3	2	1	0
PIXEL:	1st	2nd	3rd	4th	5th	6th	7th	8th

Figure 15-6. Byte-Bit/Pixel Relationship

640 x 200 Color Graphics — This mode is not implemented, even though the controller will support this mode, because it requires additional video memory and support logic.

Four 16K bit planes in memory correspond to the R, G, B, and I video signals. The configuration of these signals, along with the color palette registers, determine the actual colors used so that sixteen out of 512 possible colors can be used.

Pinout

The V6355 video controller is a surface mounted, 100-pin, large-scale integrated circuit. The pin definitions are described in Table 15-27 and illustrated in Figure 15-7.

Table 15-26. Foreground Color Select Logic

BIT 1	BIT 0	COLOR	SET 1	SET 2	MONOCHROME
0	0	Background	Background	Background	Background
0	1	Color 1	G (green)	G,B (cyan)	G,B
1	0	Color 2	R (red)	R,B (magenta)	R,B
1	1	Color 3	R,G (brown)	R,G,B (white)	R,G,B

Table 15-27. V6355 Video Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
1	—	No connection.
2	DCK	Dot clock. This signal is used by the parallel-to-serial shift register to clock the serial signals to the V6355.
3	CCL	Video memory latch clock. This signal is used to gate the signals from the video memory and latch them for the character generator.
4	$\overline{\text{LOAD}}$	Load. This output signal controls the parallel-to-serial shift register.
5	RA2	Row address 2. This signal is one of three signals used to address the Y matrix of the character generator.
6	RA1	Row address 1. This signal is one of three signals used to address the Y matrix of the character generator.
7	—	No connection.
8	RA0	Row address 0. This signal is one of three signals used to address the Y matrix of the character generator.
9	GND	Ground.
10	—	No connection.
11	—	No connection.
12	GD0	Graphic dot data. This signal is the serial dot data from the parallel to serial converter that converts the character generated signals to a video dot pattern.
13	MY1	This input is pulled low — not used in this computer.
14	MY2	This input is pulled low — not used in this computer.
15	MX1	This input is pulled low — not used in this computer.
16	MX2	This input is pulled low — not used in this computer.
17	VSY/FLM1	Vertical sync/FLM. This signal is the vertical sync signal for CRT operation and an FLM signal, which is not used in this computer, for some types of LCD operation. Both signals are generated by the V6355.
18	VST/FLM0	Vertical retrace/FLM. This signal is the vertical retrace timing signal for CRT operation and the only FLM signal used for the common drivers in the LCD. Both signals are generated by the V6355.

Table 15-27 (Continued). V6355 Video Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
19	HSY/LC	Horizontal sync/latch clock. This signal is the horizontal sync signal for CRT operation and the latch clock for LCD operation. Both signals are generated by the V6355.
20	EWCK	Enable weight clock. This signal enables the LCD drivers and allows them to display a variable intensity signal.
21	SCK	Shift Clock. This signal is used to shift display data in the 80-column LCD segment drivers from 4-bit by 20 shift register into the 80-bit latch.
22	RESET	Reset. This signal clears all internal registers. It is generated by the gate array when the computer is first turned on.
23	$\overline{\text{IOR}}$	Input/output read. This is the active low read signal for reading the contents of the readable registers of the V6355.
24	$\overline{\text{IOW}}$	Input/output write. This is the active low write signal for reading the contents of the writable registers of the V6355.
25	$\overline{\text{IOSEL}}$	Input/output select. This is the active low decode signal for the upper bit of the address signal of the input/output registers of the V6355.
26	—	No connection.
27	—	No connection.
28 - 30	CD0 - CD2	System data bits 0 - 2. These are the system data bus bits 0 through 2.
31	—	No connection.
32 - 36	CD3 - CD7	System data bits 3 - 7. These are the system data bus bits 3 through 7.
37 - 45	RD0 - RD7	Video memory data bits 0 - 7. These are the video memory data bus bits 0 through 7.
46	AVss	Ground for the analog circuitry of the V6355.
47	B/LD0	Blue/LCD data bit 0. This signal is not used for CRT operation in this computer. During LCD operation, it is data bit 0, sent to the LCD drivers.
48	—	No connection.
49	—	No connection.

Table 15-27 (Continued). V6355 Video Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
50	G/LD1	Green/LCD data bit 1. This signal is not used for CRT operation in this computer. During LCD operation, it is data bit 1, sent to the LCD drivers.
51	R/LD2	Red/LCD data bit 2. This signal is not used for CRT operation in this computer. During LCD operation, it is data bit 2 that is sent to the LCD drivers.
52	Y/LD3	Composite monochrome/LCD data bit 3. This signal is used to develop the composite monochrome signal for CRT operation. During LCD operation, it is data bit 3 that is sent to the LCD drivers.
53	—	No connection.
54	CH/M	Chrominance/M. This signal is not used during CRT operation. The M signal is disabled by a jumper in the LCD display section of the computer. It is required by some types of displays and is used for driving the LCD.
55	AV_{DD}	+5 VDC power supply. This supply is fed to the V6355 through a choke.
56	BDIR	No connection.
57	$\overline{\text{MEMSEL}}$	Memory select. This is the active low video memory select signal generated by the decoder (refer to Chapter 15). It is the decode signal for the upper bits of the memory address.
58	$\overline{\text{MEMR}}$	Memory read. This active low signal generated by the gate array is used to read data from video memory.
59	$\overline{\text{MEMW}}$	Memory write. This active low signal generated by the gate array is used to write data to video memory.
60	MEMRDY	Memory ready. This is the video memory ready signal, indicating that video memory can be read from or written to. When low, video memory is busy.
61	$\overline{\text{LDT}}/\overline{\text{MS1}}$	This input is pulled high — not used in this computer.
62	$\overline{\text{LSW}}/\overline{\text{MS0}}$	This input is pulled high — not used in this computer.

Table 15-27 (Continued). V6355 Video Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
63	GND	Ground.
64	X1	Oscillator. This is an input for an external crystal oscillator.
65	X2	Oscillator. This is an input for an external crystal oscillator.
66	—	No connection
67	—	No connection.
68	—	No connection.
69	DI/LD4	Intensity/LCD data bit 4. This signal is the intensity signal for CRT operation. During LCD operation, it is data bit 4, sent to the LCD drivers.
70	DR/LD5	Red/LCD data bit 5. This signal is the red signal for CRT operation. During LCD operation, it is data bit 5, sent to the LCD drivers.
71	DG/LD6	Green/LCD data bit 6. This signal is the green signal for CRT operation and during LCD operation, it is data bit 6 that is sent to the LCD drivers.
72	DB/LD7	Blue/LCD data bit 7. This signal is the blue signal for CRT operation. During LCD operation, it is data bit 7, sent to the LCD drivers.
73	—	No connection.
74	—	No connection.
75 -78	A0 - A3	System address bits 0 - 3. These are the input/output register's address lines 0 through 3.
79	—	No connection.
80	$\overline{\text{XA}}$	Address enable for static RAM.
81	$\overline{\text{XB}}$	Not connected.
82 - 86	AD0 - AD4	Video address bits 0 - 4. These are the video memory address bits 0 through 4. They can be generated either by the V6355 or through the address buffer.
87	—	No connection.
88 - 89	AD5 - AD6	Video address bits 5 - 6. These are the video memory address bits 5 and 6. They can be generated either by the V6355 or through the address buffer.
90	VDD	+5 VDC. +5 volts DC power supply.
91	AD7	Video address bit 7. This is the video memory address bit 7. It can be generated either by the V6355 or through the address buffer.
92	—	No connection.

Table 15-27 (Continued). V6355 Video Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
93 - 97	AD8 - AD12	Video address bits 8 - 12. These are the video memory address bits 8 through 12. They can be generated either by the V6355 or through the address buffer.
98	$\overline{\text{LSEL}}$	Low select. This active low signal enables dynamic video memory to latch the least-significant byte of the 16-bit address word.
99	$\overline{\text{HSEL}}$	High select. This active low signal enables dynamic video memory to latch the most-significant byte of the 16-bit address word.
100	$\overline{\text{WE}}$	Write Enable. This signal, generated by the V6355, is used to enable writing to video memory devices.

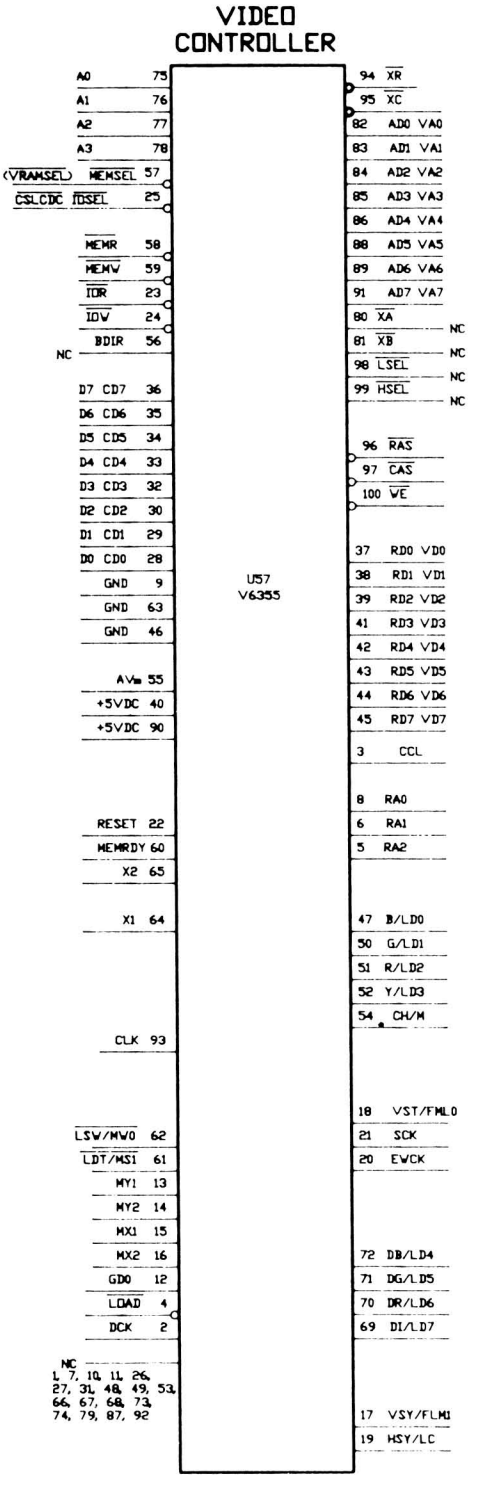


Figure 15-7. V6355 Video Controller Pinout

Chapter 16
Communications

This chapter describes the theory of operation for the circuits that support the serial and parallel connectors and the modem in the Portable computer.

The Programmable Peripheral Interface

The programmable peripheral interface in this computer is integrated into the CPU gate array, which is described in Chapter 14. The interface provides data communications to the keyboard circuits, sends status information from the switch settings to the CPU, and monitors and controls the sound generator timer. It also controls communications through the parallel port.

The interface consists of a control register and 24 programmable input/output lines (PA0 - PA7, PB0 - PB7, and PC0 - PC7), some used internally by the gate array and some used by the buffered data lines BD0 - BD7. A number of control lines, generated by the gate array or controller, allow the buffered data lines to pass data to and from the address bus, the address/data bus, and the data bus. The control register is a write-only register and selects the operating mode of the interface. This operating mode determines whether the lines are inputs or outputs. Although the 8255, the programmable peripheral interface in most PC-compatible computers, has three possible operating modes, only mode 0 is used in this computer.

To program the interface for mode 0, send 099H to the control register at port address 063H. This causes the interface to access the lines as inputs or outputs as described in Table 16-1.

Table 16-1. Interface Status in Mode 0

PORT ADDRESS	LINES	DESCRIPTION
060H	PA0 - PA7	Input
061H	PB0 - PB7	Output
062H	PC0 - PC7	Input

Lines PB0, PB1, PC4, and PC5 control and monitor the activities of the sound generator and are internal to the gate array. PA0 - PA7 and PB7 work with the keyboard processor. The remaining lines establish the system configuration, defined by a set of switches in the original PC-compatible design. In this computer, most of the system configuration is set up by the gate array with only a few features handled by the configuration switch described in Chapter 2. The equipment configuration report is returned by INT 11H and described in Chapter 7.

The Printer Gate Array

The printer gate array is a custom device that replaces many discrete components that would otherwise be required in this computer. It provides the LCD/CRT toggle signal, power control to the LCD backlight, and control signals to enable/disable the internal modem. It also provides the interface to the parallel port.

Parallel Port

Refer to the parallel port block diagram in Figure 16-1. The parallel port is addressed as LPT1. It has eight output data lines and nine handshake lines. Four of the handshake lines can be programmed as either output or input; the remaining five are input only. The data lines are designed with input circuits that are used for loop-back diagnostic programs to determine fault isolation between the parallel port and the attached peripheral. The connector is described in Chapter 12.

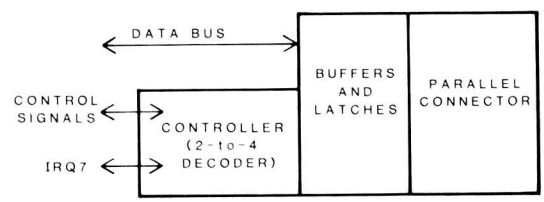


Figure 16-1. Parallel Port Block Diagram

There are no major integrated circuits in the parallel port. The majority of the signals go through a dual 2-to-4 decoder that acts as a controller for the buffers

and latches between the system data bus and the input and output lines of the parallel connector.

IRQ7, the parallel printer interrupt, handles the clocking of the data signals out to the parallel device and operates only during the transmitting of parallel data. Refer to Chapter 6 for more information on this hardware interrupt.

While individual output operations can be handled directly by software, the system interrupts are very efficient and easier to use. Refer to Chapter 9 for more information.

Serial Port

Refer to the serial port block diagram in Figure 16-2. The RS-232C serial port is addressed as COM1. It is designed around the TC8570 Universal Asynchronous Receiver Transmitter. This IC is bidirectional and permits input/output communication with other serial devices. The IC handles both parallel-to-serial and serial-to-parallel conversion and any necessary parity and handshaking processing.

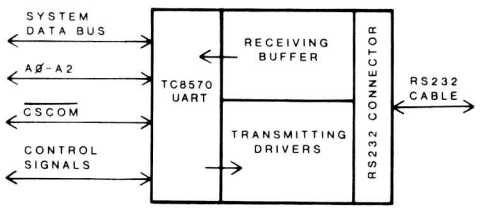


Figure 16-2. Serial Port Block Diagram

The TC8570 Universal Asynchronous Receiver Transmitter

The TC8570 controls the serial communications channel. A block diagram of this device is shown in Figure 16-3.

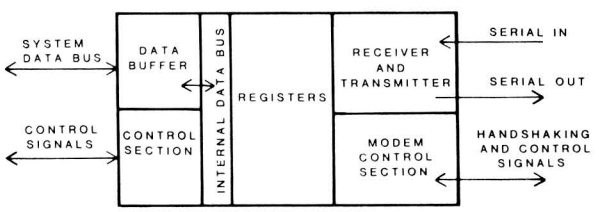


Figure 16-3. TC8570 Block Diagram

The TC8570 interfaces the CPU with the RS 232C serial input/output connector on the back of the computer. The TC8570 receives information through an internal data buffer from the system data bus. The TC8570 can be programmed with this data or transmit it out the port. Addressing of the TC8570 is handled by the chip select line and address lines 0 - 2. Control logic inside the TC8570 determines if a register is being programmed or if there is data ready to transmit or receive.

The receiver obtains serial data from the serial in pin of the connector and converts it into a parallel form before making it available for the system data bus.

The transmitter converts parallel data from the system data bus into a serial form and then sends it to the serial out pin of the connector.

The modem controller provides the control signals used for serial communications.

The interrupt controller signals the processor when the TC8570 requires service to continue transmitting or receiving data. In this computer, this line is used only to request service while receiving data.

The TC8570 provides a number of programmable options: manipulating interrupts, modem controls, word length, parity, number of stop bits, and the baud rate. Most of these functions can be handled at the system level through the interrupts, described in Chapter 9. Register programming information follows the pinout.

Handshaking

Two devices asynchronously communicate through handshaking. Two methods are used: software and/or hardware. This part of the manual is concerned only with hardware handshaking. For information on software handshaking, refer to Chapter 9.

The *four* input handshaking lines at the TC8570 are RLSD, DSR, CTS and RI. The RLSD (receive line signal detect) line is connected to the carrier detect line on the RS 232C connector. It is asserted whenever an externally connected modem detects an incoming carrier. The DSR (data set ready) line is asserted when a modem or other device is ready to establish communications with the computer. The CTS (clear to send) line is asserted when the external device is ready to accept data. The RI (ring indicator) line is asserted when an attached modem

detects a ring signal on the telephone line. All signals are inverted between the connector and the TC8570 by an inverter/receiver.

The two output handshake lines are DTR and RTS. The DTR (data terminal ready) line is asserted when the computer is ready to communicate with an external device. The RTS (request to send) line is asserted when the computer is ready to transmit data. These two lines are buffered by drivers between the TC8570 and the connectors.

TC8570 Programming

There are ten registers in the TC8570: the receiver buffer register, the transmitter holding register, the interrupt enable register, the interrupt identification register, the line control register, the modem control register, the line status register, the modem status register, the scratchpad register, and the two divisor latches. Refer to Table 16-2 for the port addresses of these registers.

Table 16-2. TC8570 Register Ports

ADDRESS	REGISTER
3F8H	Receiver buffer register
3F8H	Transmitter holding register
3F8H	Divisor latch (least significant byte)
3F9H	Divisor latch (most significant byte)
3F9H	Interrupt enable register
3FAH	Interrupt identification register
3FBH	Line control register
3FCH	Modem control register
3FDH	Line status register
3FEH	Modem status register
3FFH	Scratchpad register

The receiver buffer register and the transmitter holding register share the same port address. The receiver buffer register is a read-only register and the transmitter holding register is a write-only register. Both registers contain the data received or to be transmitted. The first data bit to be transmitted or received is the least-significant bit.

The divisor latches establish the baud rate based on the input frequency of the oscillator clock to the TC8570. This computer uses an oscillator with a frequency of 1.8432 MHz. You can program the latches with any number to produce a value that is sixteen times the actual baud rate. The values that produce standard baud rates, along with the error percentage, are provided in Table 16-3.

NOTE: To write to the divisor latches, you must first set bit 7 in the line control register. Otherwise, the receiver buffer or transmitter holding register, or the interrupt enable register will be written to.

Table 16-3. Baud Rate and Divisor Latch Values

DESIRED BAUD RATE	DIVISOR VALUE	ERROR PERCENTAGE
50	2304	0
75	1536	0
110	1047	0.026
134.5	857	0.058
150	768	0
300	384	0
600	192	0
1200	96	0
1800	64	0
2000	58	0.69
2400	48	0
3600	32	0
4800	24	0
7200	16	0
9600	12	0
19200	6	0
38400	3	0
56000	2	2.86

NOTE: Although these and any other baud rate can be programmed into the TC8570, PC-compatible hardware and software supports the following baud rates: 110, 150, 300, 600, 1200, 2400, 4800, 9600, and 19,200.

The two interrupt registers work together, but have different port addresses. The interrupt enable register uses four bits to enable the interrupt capabilities of the TC8570. Four interrupts are recognized: received data available interrupt, transmitter holding register empty interrupt, receiver line status interrupt, and modem status interrupt. These interrupts are identified by the first three bits of the interrupter identification register. Their priority level, source, and reset control are described in Table 16-4.

Table 16-4. TC8570 Interrupts

INTERRUPT IDENTIFICATION REGISTER VALUE	PRIORITY LEVEL	SOURCE	RESET CONTROL
0	4	Clear to send, data set ready, ring indicator, or received line signal detect.	Read modem status register.
1	—	None.	None.
2	3	Empty interrupt identification register or transmitter holding register.	Read the interrupt identification register or the transmitter holding register, whichever caused the interrupt.
3	—	None.	None.
4	2	Received character available in the receiver buffer register.	Read the receiver buffer register.
5	—	None.	None.
6	1	Overrun, parity, or framing error, or break interrupt.	Read the line status register.
7	—	None.	None.

The line control register establishes many of the parameters for serial communications. Each bit is described in Table 16-5.

Table 16-5. Line Control Register

BIT	DESCRIPTION
0 - 1	Word length: 0 = 5 bits, 1 = 6 bits, 2 = 7 bits, and 3 = 8 bits. Bit 0 is the least significant bit.
2	Number of stop bits dependent upon the word length. Word length = 6, 7, or 8: clear (0) = 1 stop bit, set (1) = 2 stop bits. Word length = 5: clear (0) = 1 stop bit, set (1) = 1.5 stop bits.
3	Parity enabled or not: clear (0) = parity disabled, set (1) = parity enabled.
4	Even or odd parity: clear (0) = odd parity, set (1) = even parity.
5	Stick parity. When parity (bit 3) and stick parity are both enabled, parity will be opposite that indicated by bit 4 (odd/even parity). Clear (0) = stick parity disabled, set (1) = stick parity enabled.
6	Set break control. When this bit is set (1), the serial output goes low (space) and remains there until this bit is cleared (0). This feature allows the computer to alert a remote station, or vice versa, in a data communications network.
7	Allows divisor latches to be accessed: clear (0) = latch access disabled, set (1) = the latches can be read from or written to.

The modem control register handles several parameters and some of the handshaking protocol for the TC8570. Each bit is described in Table 16-6.

Table 16-6. Modem Control Register

BIT	DESCRIPTION
0	This bit controls the data terminal ready output: clear (0) = DTR output is high, set (1) = DTR output is low. See the note at the bottom of this table.
1	This bit controls the request to send output: clear (0) = RTS output is high, set (1) = RTS output is low. See the note at the bottom of this table.
2	This bit controls the output 1 signal: clear (0) = OUT1 output is high, set (1) = OUT1 output is low.
3	This bit controls the output 2 signal in same manner that the output 1 signal is controlled. However, OUT2 is pulled high in this computer and not used.
4	This bit provides a local loop back condition to test the TC8570. It is used by the disk-based diagnostics to check the data transmit and receive circuits of the TC8570. For complete information on the use of this bit, refer to the Toshiba data sheet for the TC8570.
5 - 7	Not used, permanently clear (0).

NOTE: The outputs described in this table are at the chip and do not necessarily reflect output signals at the connector.

The line status register provides a status report concerning data transfer. Each bit is described in Table 16-7.

Table 16-7. Line Status Register Report

BIT	DESCRIPTION
0	Indicates that a complete character has been received and transferred into the receiver buffer register: set (1) = character received and ready; clear (0) = receiver buffer register is empty (either no character has been received or the receiver buffer register has been read.
1	Indicates that the receiver buffer register was not read before the next character was received: set (1) = the next character has been received before the receiver buffer register was read; clear (0) = no overrun has taken place.
2	Indicates that a parity error has been detected: set (1) = parity error detected; clear (0) = no parity error.
3	Indicates that a framing error has taken place, no valid stop bit was received: set (1) = framing error detected; clear (0) = no framing error.
4	Indicates that a break interrupt has been detected. A break interrupt occurs when the received data input is held at a logical 0 state longer than a full word time (start plus data plus parity plus stop bits).
5	Indicates that the transmitter holding register is empty: set (1) = transmitter holding register empty; clear (0) = transmitter holding register receiving or has a character. This bit will produce an interrupt to the CPU if the transmit holding register empty interrupt is enabled.
6	Indicates whenever the transmitter shift register is idle: set (1) = shift register is idle; clear (0) = shift register has received a character or is active (shifting data out).
7	Not used, permanently clear (0).
NOTE:	Bits 1 - 4 produce a receiver line status interrupt when one of the conditions is detected.

The modem status register provides a status report concerning the handshaking and control lines. Each bit is described in Table 16-8.

Table 16-8. Modem Status Register Report

BIT	DESCRIPTION
0	Indicates that the $\overline{CTS}$ signal has changed state since the last time this register was read: set (1) = changed state, clear (0) = no change.
1	Indicates that the $\overline{DSR}$ signal has changed state since the last time this register was read: set (1) = changed state, clear (0) = no change.
2	Indicates that the $\overline{RI}$ signal has changed from a mark (set or logical 1) condition to a space (clear or logical 0) condition: set (1) = changed state, clear (0) = no change.
3	Indicates that the $\overline{DCD}$ signal has changed state since the last time this register was read: set (1) = changed state, clear (0) = no change.
4	Reflects the compliment of the $\overline{CTS}$ input: set (1) = $\overline{CTS}$ = clear (0), clear (0) = $\overline{CTS}$ set (1).
5	Reflects the compliment of the $\overline{DSR}$ input: set (1) = $\overline{DSR}$ = clear (0), clear (0) = $\overline{DSR}$ set (1).
6	Reflects the compliment of the $\overline{RI}$ input: set (1) = $\overline{RI}$ = clear (0), clear (0) = $\overline{RI}$ = set (1).
7	Reflects the compliment of the $\overline{DCD}$ input: set (1) = $\overline{DCD}$ = clear (0), clear (0) = $\overline{DCD}$ set (1).
NOTE:	Bits 0 - 3 produce a modem status interrupt when one of the conditions is detected and the corresponding bit is set.

Pinout

The pin functions of the TC8570 are described in Table 16-9 and the pinout is illustrated in Figure 16-4.

Table 16-9. TC8570 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1 - 3	DB5 - DB7	Data line 5 - data line 7. These three-state, bidirectional lines are connected to the computer's data bus.

Table 16-9 (Continued). TC8570 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
4	RCLK	Receive clock. This input is tied to pin 11, the baud out signal. It provides the timing for the baud rate of the receiving circuits inside the TC8570. By tying this line to the baud out signal, the transmitting and receiving baud rates remain the same.
5	—	No connection.
6	SIN	Serial input. This input receives the input signal from the connector. It is buffered by an external inverter/buffer.
7	SOUT	Serial output. This output line transmits the serial data to the connector through an external buffer/driver.
8 - 9	CS0 - CS1	Chip select lines 0 - 1. These two lines are tied high. Selection of the serial port is handled by pin 10.
10	$\overline{CS2}$	Chip select. This input signal comes from the decoder, described in Chapter 14. It is used to select the serial port.
11	$\overline{BAUDOUT}$	Baud out. This signal is a clock signal that runs sixteen times the transmitted baud rate. This output is tied to pin 4, the receive clock, to make the transmit and receive baud rates identical.
12	XTAL 1	External clock in. This is one side of the 1.8432 MHz crystal oscillator. It is used as the frequency base for the various baud rates that can be programmed for the TC8570.
13	XTAL 2	External clock out. This is the other side of the 1.8432 MHz crystal oscillator. The output is used to stabilize the oscillator.
14	$\overline{DOSTR}$	Data out strobe. This line is connected to the system IOW line, which is used to write data to the selected TC8570 register.
15	DOSTR	Data out strobe. Tied low and not used.
16	GND	Ground.
17	Vcc	+5 VDC power supply.
18	$\overline{DISTR}$	Data in strobe. This line is connected to the system IOR line, which is used to read the contents of the selected TC8570 register.

Table 16-9 (Continued). TC8570 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
19	DISTR	Data in strobe. Tied low and not used.
20	DDIS	Driver disable. Not connected.
21	CSOUT	Chip select out. Not connected.
22	$\overline{ADS}$	Address strobe. This input provides latching for the address and chip select lines. Since these inputs are stable in this computer, this line is tied low.
23 - 25	A2 - A0	Address line 0 - address line 2. These lines provide register addressing information for the TC8570.
26	—	Not used.
27	INTRPT	Interrupt. This output is ANDed with the IOR line to produce IRQ4 for the system. OUT2 activates the interrupt.
28	—	Not used.
29	$\overline{OUT2}$	Output 2. This output is programmed by the system to control IRQ4 selection.
30	$\overline{RTS}$	Request to send. This output indicates that the TC8570 is ready to send information. It is a programmed handshaking signal.
31	$\overline{DTR}$	Data terminal ready. This output indicates that the TC8570 is read to communicate. It is a programmed handshaking signal.
32	OUT1	Output 1. This output is pulled high and not used in this computer.
33	MR	Master reset. This input line is tied to RESET and clears and resets the TC8570's registers.
34	$\overline{CTS}$	Clear to send. This input indicates that the remote device is ready to transmit. It is a programmed handshaking signal.
35	$\overline{DSR}$	Data set ready. This input indicates the status of the data set. It is a programmed handshaking signal.
36	$\overline{DCD}$	Data carrier detect. This input is tied to the connector's carrier detect input pin and indicates that an acceptable-quality signal carrier has been detected. It is a programmed handshaking signal.

Table 16-9 (Continued). TC8570 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
37	RI	Ring indicator. This input is tied to the serial connector's ring detect pin and indicates that a ringing signal has been detected. It is a programmed handshaking signal.
38 - 39	Vcc	+5 VDC power supply.
40 - 44	DB0 - DB4	Data line 0 - data line 4. These three-state bidirectional lines are connected to the computer data bus.

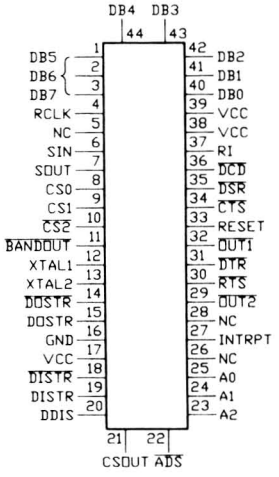


Figure 16-4. TC8570 Pinout

The Modem

The modem is similar to the serial port. In fact, it is the second COM port. MS-DOS identifies this port as COM2. The following material discusses the modem's serial device. The modem programming interprets the register and device input and output lines.

There are nine registers in the modem's serial device: the receiver buffer register, the transmitter holding register, the interrupt enable register, the interrupt identification register, the line control register, the modem control register, the line status register, the modem status register, and the two divisor latches. Refer to Table 16-10 for the port addresses of these registers.

Table 16-10. Serial Device Register Ports

ADDRESS	REGISTER
2F8H	Receiver buffer register
2F8H	Transmitter holding register
2F8H	Divisor latch (least significant byte)
2F9H	Divisor latch (most significant byte)
2F9H	Interrupt enable register
2FAH	Interrupt identification register
2FBH	Line control register
2FCH	Modem control register
2FDH	Line status register
2FEH	Modem status register
2FFH	Scratchpad register

The receiver buffer register and the transmitter holding register share the same port address. The receiver buffer register is a read-only register and the transmitter holding register is a write-only register. Both registers contain the data received or to be transmitted. The first data bit to be transmitted or received is the least-significant bit.

The divisor latches establish the baud rate based on the input frequency of the oscillator clock to the serial device. This computer uses an oscillator with a frequency of 1.8432 MHz. You can program the latches with any number to produce a value that is sixteen times the actual baud rate. The values that produce standard baud rates, along with the error percentage, are provided in Table 16-11.

NOTE: To write to the divisor latches, you must first set bit 7 in the line control register. Otherwise, the receiver buffer or transmitter holding register, or the interrupt enable register will be written to.

Table 16-11. Baud Rate and Divisor Latch Values

DESIRED BAUD RATE	DIVISOR VALUE	ERROR PERCENTAGE
50	2304	0
75	1536	0
110	1047	0.026
134.5	857	0.058
150	768	0
300	384	0
600	192	0
1200	96	0

Table 16-11 (Continued). Baud Rate and Divisor Latch Values

DESIRED BAUD RATE	DIVISOR VALUE	ERROR PERCENTAGE
1800	64	0
2000	58	0.69
2400	48	0
3600	32	0
4800	24	0
7200	16	0
9600	12	0
19200	6	0
38400	3	0
56000	2	2.86

NOTE: Although these and any other baud rate can be programmed into the serial device, PC-compatible hardware and software supports the following baud rates: 110, 150, 300, 600, 1200, 2400, 4800, 9600, and 19,200.

The two interrupt registers work together, but have different port addresses. The interrupt enable register uses four bits to enable the interrupt capabilities of the serial device. Four interrupts are recognized: received data available interrupt, transmitter holding register empty interrupt, receiver line status interrupt, and modem status interrupt. These interrupts are identified by the first three bits of the interrupter identification register. Their priority level, source, and reset control are described in Table 16-12.

Table 16-12. Serial Device Interrupts

INTERRUPT IDENTIFICATION REGISTER VALUE	PRIORITY LEVEL	SOURCE	RESET CONTROL
0	4	Clear to send, data set ready, ring indicator, or received line signal detect.	Read modem status register.
1	—	None.	None.
2	3	Empty interrupt identification register or transmitter holding register.	Read the interrupt identification register or the transmitter holding register, whichever caused the interrupt.
3	—	None.	None.
4	2	Received character available in the receiver buffer register.	Read the receiver buffer register.
5	—	None.	None.
6	1	Overrun, parity, or framing error, or break interrupt.	Read the line status register.
7	—	None.	None.

The line control register establishes many of the parameters for serial communications. Each bit is described in Table 16-13.

Table 16-13. Line Control Register

BIT	DESCRIPTION
0 - 1	Word length: 0 = 5 bits, 1 = 6 bits, 2 = 7 bits, and 3 = 8 bits. Bit 0 is the least significant bit.
2	Number of stop bits dependent upon the word length. Word length = 6, 7, or 8: clear (0) = 1 stop bit, set (1) = 2 stop bits. Word length = 5: clear (0) = 1 stop bit, set (1) = 1.5 stop bits.
3	Parity enabled or not: clear (0) = parity disabled, set (1) = parity enabled.
4	Even or odd parity: clear (0) = odd parity, set (1) = even parity.
5	Stick parity. When parity (bit 3) and stick parity are both enabled, parity will be opposite that indicated by bit 4 (odd/even parity). Clear (0) = stick parity disabled, set (1) = stick parity enabled.
6	Set break control. When this bit is set (1), the serial output goes low (space) and remains there until this bit is cleared (0). This feature allows the computer to alert a remote station, or vice versa, in a data communications network.
7	Allows divisor latches to be accessed: clear (0) = latch access disabled, set (1) = the latches can be read from or written to.

The modem control register handles several parameters and some of the handshaking protocol for the serial device. Each bit is described in Table 16-14.

Table 16-14. Modem Control Register

BIT	DESCRIPTION
0	This bit controls the data terminal ready output: clear (0) = $\overline{\text{DTR}}$ output is high, set (1) = $\overline{\text{DTR}}$ output is low. See the note at the bottom of this table.
1	This bit controls the request to send output: clear (0) = $\overline{\text{RTS}}$ output is high, set (1) = $\overline{\text{RTS}}$ Routput is low. See the note at the bottom of this table.
2	This bit controls the output 1 signal: clear (0) = $\overline{\text{OUT1}}$ output is high, set (1) = $\overline{\text{OUT1}}$ output is low.

Table 16-14 (Continued). Modem Control Register

BIT	DESCRIPTION
3	This bit controls the output 2 signal in same manner that the output 1 signal is controlled. However, $\overline{\text{OUT2}}$ is pulled high in this computer and not used.
4	This bit provides a local loop back condition to test the serial device. It is used by the disk-based diagnostics to check the data transmit and receive circuits of the serial device. For complete information on the use of this bit, refer to the Toshiba data sheet for the TC8570.
5 - 7	Not used, permanently clear (0).

NOTE: The outputs described in this table are at the chip level and do not reflect output signals.

The line status register provides a status report concerning data transfer. Each bit is described in Table 16-15.

Table 16-15. Line Status Register Report

BIT	DESCRIPTION
0	Indicates that a complete character has been received and transferred into the receiver buffer register: set (1) = character received and ready; clear (0) = receiver buffer register is empty (either no character has been received or the receiver buffer register has been read.
1	Indicates that the receiver buffer register was not read before the next character was received: set (1) = the next character has been received before the receiver buffer register was read; clear (0) = no overrun has taken place.
2	Indicates that a parity error has been detected: set (1) = parity error detected; clear (0) = no parity error.
3	Indicates that a framing error has taken place, no valid stop bit was received: set (1) = framing error detected; clear (0) = no framing error.
4	Indicates that a break interrupt has been detected. A break interrupt occurs when the received data input is held at a logical 0 state longer than a full word time (start plus data plus parity plus stop bits).

Table 16-15 (Continued). Line Status Register Report

BIT	DESCRIPTION
5	Indicates that the transmitter holding register is empty: set (1) = transmitter holding register empty; clear (0) = transmitter holding register receiving or has a character. This bit will produce an interrupt to the CPU if the transmit holding register empty interrupt is enabled.
6	Indicates whenever the transmitter shift register is idle: set (1) = shift register is idle; clear (0) = shift register has received a character or is active (shifting data out).
7	Not used, permanently clear (0).

NOTE: Bits 1 - 4 produce a receiver line status interrupt when one of the conditions is detected.

The modem status register provides a status report concerning the handshaking and control lines. Each bit is described in Table 16-16.

Table 16-16. Modem Status Register Report

BIT	DESCRIPTION
0	Indicates that the $\overline{CTS}$ signal has changed state since the last time this register was read: set (1) = changed state, clear (0) = no change.
1	Indicates that the $\overline{DSR}$ signal has changed state since the last time this register was read: set (1) = changed state, clear (0) = no change.
2	Indicates that the $\overline{RI}$ signal has changed from a mark (set or logical 1) condition to a space (clear or logical 0) condition: set (1) = changed state, clear (0) = no change.
3	Indicates that the $\overline{DCD}$ signal has changed state since the last time this register was read: set (1) = changed state, clear (0) = no change.
4	Reflects the compliment of the $\overline{CTS}$ input: set (1) = $\overline{CTS}$ = clear (0), clear (0) = $\overline{CTS}$ set (1).
5	Reflects the compliment of the $\overline{DSR}$ input: set (1) = $\overline{DSR}$ = clear (0), clear (0) = $\overline{DSR}$ set (1).
6	Reflects the compliment of the $\overline{RI}$ input: set (1) = $\overline{RI}$ = clear (0), clear (0) = $\overline{RI}$ = set (1).
7	Reflects the compliment of the $\overline{DCD}$ input: set (1) = $\overline{DCD}$ = clear (0), clear (0) = $\overline{DCD}$ set (1).

NOTE: Bits 0 - 3 produce a modem status interrupt when one of the conditions is detected and the corresponding bit is set.

Chapter 17
Mass Storage

The term mass storage applies to all forms of off-line storage including magnetic disk and tape, and punched tape and cards. Modern microcomputers use magnetic disks of varying sizes almost exclusively. Many hard disk systems offer magnetic tape backup systems and occasionally you will find punched or marked card readers attached to computers. This computer uses two types of disks for mass storage: 3.5-inch disks with a formatted capacity of 730,112 bytes and 5.25-inch disks with a formatted capacity of 362,496 bytes. This chapter discusses the programmable elements of the mass storage system in this computer.

NOTE: Throughout this chapter, "cylinder" and "track" are used interchangeably. A cylinder has one or more tracks that can be accessed by the disk drive heads without moving the heads through a seek or recalibrate operation.

Supported Drives

This computer supports one or two built-in 3.5-inch disk drives, identified by MS-DOS as drive A and drive B. An optional external disk drive system supports one or two standard 5.25-inch disk drives, identified by MS-DOS as drive C and drive D.

The operating system and some of the devices used in this computer will support both dual-speed, high-density 5.25-inch disk drives and 8-inch disk drives. However, the actual circuits used in the computer do not make provision for 8-inch external drives or the circuits required for high-density operation. The circuits feeding the data separator are used exclusively (not available at the external floppy disk connector) to treat the 3.5-inch disks as 8-inch disks for data transfer purposes. For high-density operation, the necessary change disk signal and transfer speed register are not supplied even though you can configure the operating system. Do not attach or program the computer or its operating system for either 8-inch or high-density 5.25-inch operation.

Floppy Disk Control

The advanced technology employed in the design of the Portable Computer allows the combination of almost all floppy disk control circuitry into one device. This device, the floppy disk gate array, contains all circuits necessary for reading and writing data to the drives, data separation, timing, and control. Internally, the gate array is compatible with the 765 Floppy Disk Controller (FDC) used in most PC compatible products. Because of this the programming information described in this chapter is applicable to the gate array used in this computer as well as the more common 765 FDC

NOTE: There is no control mode register to support 5.25-inch, dual-speed, high-density disk drives in this computer. Therefore, do not attach dual-speed, high-density drives or configure the operation system for high-density use.

The Floppy Disk Controller

The floppy disk controller provides an interface between the processor and the disk drives. When reading a disk, the controller converts the serial data from the disk drive to parallel data. When writing to a disk, the controller converts the parallel data to a serial form that can be written to the disk. The controller also adds a number of timing signals before sending the data to the selected drive.

The floppy disk controller in this computer is an extended CMOS version of the 765 disk controller used in most PC-compatible designs. The controller has the added capability to handle the format required to read from and write to 3.5-inch disks.

The floppy disk controller contains control functions to interface the processor and up to four disk drives. It supports FM and MFM disk encoding formats and can be operated in DMA and non-DMA modes.

The internal data buffer transfers data between the internal bus and the system data bus.

The internal bus passes information between the data buffer and the serial controller, where it is written to or read from the disk and the registers, where it is used to program the controller.

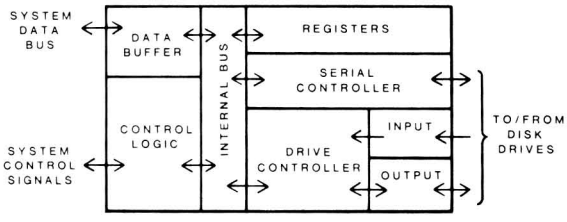


Figure 17-1. 765-Compatible Disk Controller Block Diagram

The serial controller sends data to the disk drives, converting the 8-bit parallel-format data into a serial form. The controller also converts serial-format data from the drive into an 8-bit parallel form. Using timing control provided by the write clock signal, the controller makes sure that the data is correctly timed during all transfer operations. The serial controller also generates the synchronizing signals used by the data separator for timing.

The drive controller monitors the activities of the input and output channels of the floppy disk controller.

The input channel monitors and receives drive ready, write-protect, index, and track 0 signals. These signals go to the drive controller, which responds by sending signals out the output channel or to the control logic section of the floppy disk controller.
The output channel generates seek, head load, head select, direction, and step signals for the drive. It also controls the recording mode (FM or MFM) used by the drive.

The control logic section of the controller is the master control for all operations and uses signals from the system control bus. These control signals handle the data buffer, internal registers, serial controller, and the drive controller through the internal bus. It also handles interrupts, DMA transfers, request and acknowledge signals, and read and write operations.

The registers in the controller are accessed by reading from or writing to two ports: 3F4H for reading the main status register and 3F5H for reading from or writing to the data register stack. The status register contains the main status information of the floppy disk controller and may be read at any time. The data register port actually provides sequential access to a stack of registers used to program the functions of the controller, temporarily hold the data being read from or written to the disk, and provide additional status reports. The sequence and data used with the data register stack is dependent upon the programmed instruction.

The Main Status Register

The main status register is an 8-bit read-only register located at port address 3F4H. The information defined by each bit in the register is described in Table 17-1.

Table 17-1. Main Status Register Report

BIT	DESCRIPTION
0	Set (1) when disk drive A is busy (in the seek mode). The controller will not accept read or write commands for any disk drive.
1	Set (1) when disk drive B is busy (in the seek mode). The controller will not accept read or write commands for any disk drive.
2	Set (1) when disk drive C is busy (in the seek mode). The controller will not accept read or write commands for any disk drive.
3	Set (1) when disk drive D is busy (in the seek mode). The controller will not accept read or write commands for any disk drive.
4	Set (1) when a read or write command is in process. The controller will not accept any other command.
5	Set (1) during the execution phase of a non-DMA operation.
6	Indicates the direction of data flow between the data register stack and the system data bus: clear (0) = transfer from system data bus to data register stack, set (1) = transfer from data register stack to system data bus.
7	Used for handshaking between the controller and the processor: clear (0) = the data register is not ready for transfer, set (1) = the data register is ready for a transfer operation.

Programming Disk Operations

Programming the controller is not a simple task. In addition to the controller, the DMA processor, a part of the gate array, also must be programmed and synchronized to move data to and from the disk. The BIOS of the computer contains all the necessary commands for accessing the floppy disk system. While you can gain a great deal of control over reading and writing information on the disk by writing your own programs, you greatly increase the chances of accidental damage to the data on your disks. What would appear to be normal read and write operations can quickly destroy data on a disk if not done correctly.

The main status register, at port 3F4H, reports the status of the controller and can be used for determining when to perform read and write operations to data register stack at port address 3F5H for programming and obtaining information.

There are 15 commands that the controller can process, which are explained later. In executing these commands, the controller has three basic phases of operation: command, execution, and result. In the command phase, the controller is sent as many as nine command bytes containing the instructions to be performed. These instructions are stored in the data register stack. During the execution phase, the controller executes the commands. In the result phase, the controller returns as many as seven bytes that report the results of the command performed. These bytes are read from the data register stack.

Before it can carry out a disk operation, the control register must first be given the proper information. For example, a data byte consisting of the bit pattern 00011100 (4CH) selects drive A, enables the DMA and interrupt circuits, and turns on disk drive A's motor.

NOTE: Bit 2, the reset bit, was not set. This bit should only be used if a serious error occurs. When the controller is reset, the disk drives must be recalibrated, which requires that each drive's head is moved to track 0.

The following sequence represents a typical series of steps that could be used in programming a disk operation.

1. Select a drive and turn on the motor.
2. Wait for the motor to come to speed.
3. Send the command instruction set to the controller.
4. Wait for the interrupt that indicates the completion of the operation.
5. Initialize the DMA to move data to or from memory.
6. Send the proper instruction to the controller to read or write data.
7. Wait for the interrupt that indicates the completion of the operation.
8. Read the status report from the controller and analyze it to make sure the operation was performed correctly.
9. Turn off the motor.

These steps represent a simplified version of a basic disk access operation, such as reading or writing a sector. Due to the structure of the disk, reading or writing a sector is not quite as simple. In order to read or write a specific sector, first read the disk directory to locate the file. Next, use the file allocation tables to locate the desired sector on the disk. Next, move the heads to the track and located the sector. Then perform the read or write operation. The software interrupts contain all the necessary routines to perform these complex operations. Refer to Chapter 11 for complete programming information on the disk drive interrupts. In addition, refer to the Programmer's Utility Package for a description of the organization of the MS-DOS disk.

The Floppy Disk Controller Commands

The controller commands consist of reading and writing one or more sectors, reading a track, locating the position of the head on a track, formatting a track, scanning and comparing written data against memory, seeking to a track, calibrating (synchronizing) head-to-track information, checking interrupt and drive status, and seeking to a specified track. The 15 commands are described in Table 17-2.

Table 17-2. Floppy Disk Controller Commands

COMMAND	DESCRIPTION
Read data	Reads data from one or more sectors from a two-track cylinder.
Read deleted data	Allows a sector marked with a deleted data mark to be read.
Write data	Writes data to one or more sectors of a one- or two-track cylinder.
Write deleted data	Same as a write data command except that deleted data marks are written instead of normal data marks.
Read a track	Reads the contents of an entire track.
Read sector ID	Reads the values of the first ID field it is able to read.
Format a track	Allows an entire track to be formatted.
Scan equal	Reads the data on the disk and compares it against data supplied by the processor. If the two sets of data are equal, the operation terminates.
Scan low or equal	Reads the data on the disk and compares it against data supplied by the processor. If the two sets of data are equal or of lesser value, the operation terminates.
Scan high or equal	Reads the data on the disk and compares against data supplied by the processor. If the two sets of data are equal or greater value, the operation terminates.
Recalibrate	Causes the head of the selected drive to be moved to track 0. The internal track counter is cleared for this drive.
Interrupt Status	Allows the user to sense interrupts that during seek and recalibrate commands, which have no result phase.
Specify	Sets the initial values of each of three internal timers. These are the head unload time, the step rate time, and the head load time.
Drive Status	Reports back the status of the disk drives.
Seek	Causes the disk drive to seek to the specified cylinder. The specified drives track counter is incremented or decremented according to the number of step pulses sent to the drive.

Read Data — This command uses nine command bytes and returns seven status bytes. It reads data from one or more sectors from a two-track cylinder. Each of the command and status bytes are described in Table 17-3.

Table 17-3. Read Data Command

BYTE	DESCRIPTION
Write 1	Write the first set of command codes. Bits 0 - 4 = 06H; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 17-4.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Write 3	Write the cylinder (track) number.
Write 4	Write the head address. This value is equal to the value of the head number (0 or 1) as defined in Table 17-4.
Write 5	Write the sector number to be read.
Write 6	Write the number of bytes per sector.
Write 7	Write the end of track, which is the final sector number to be read on a cylinder.
Write 8	Write the gap length, which is used for synchronization in read operations.
Write 9	Write the data length. If the bytes per sector value (write 6) is 0, this value is the data length to be read from or written to each sector.
Read 1	Read status register 0. Refer to Table 17-5.
Read 2	Read status register 1. Refer to Table 17-6.
Read 3	Read status register 2. Refer to Table 17-7.
Read 4	Read cylinder (track) number.
Read 5	Read head number.
Read 6	Read sector number.
Read 7	Read bytes per sector.

Table 17-4. Controller Command Bits

BIT	DESCRIPTION
Skip bit	If set (1), this bit causes the operation to skip the deleted data address mark.
FM/MFM bit	If clear (0), the controller is placed in the FM read/write mode. If set (1), the controller is placed in the MFM read/write mode. For a discussion of these modes, refer to Chapter 3.
Multitrack bit	If set (1), the multitrack mode is in effect. At the end of side 0, side 1 will be used automatically.

Table 17-4 (Continued). Controller Command Bits

BIT	DESCRIPTION
Head number bit	This is the same as the head address and will be set (1) for side 1, or clear (0) for side 0.
Drive unit number bits	These two bits specify the drive unit number: unit 0 = drive A, unit 1 = drive B, unit 2 = drive C, and unit 3 = drive D.

Table 17-5. Controller Status Register 0 Report

BIT	DESCRIPTION
0 - 1	Disk drive unit number: 0 = drive A, 1 = drive B, 2 = drive C, and 3 = drive D.
2	Head address: 0 = side 0 and 1 = side 1.
3	Not ready. If the controller is not ready during a read or write operation or if a read or write operation is attempted to the second side of the single-sided disk, this bit will be set (1).
4	Equipment check. This bit is set (1) if a fault signal has been received or track 0 was not detected before or after 77 step pulses occurred during a recalibrate operation.
5	Seek end. This bit is set (1) upon successful completion of a seek operation.
6 - 7	These two bit supply the interrupt code: 0 = normal completion of the command, no problems were encountered; 1 = command failed, problems were encountered; 2 = invalid command; 3 = command failed because the ready signal changed state.

Table 17-6. Controller Status Register 1 Report

BIT	DESCRIPTION
0	Missing address mark. No ID address mark was detected during two complete disk revolutions, or neither a data address mark nor a deleted address mark were detected. The latter fault also sets bit 0 of status register 2.

Table 17-6 (Continued). Controller Status Register 1 Report

BIT	DESCRIPTION
1	Write-protected. During a write or format command, a write-protect signal is detected from the selected drive.
2	Not found. The specified sector, ID field, or starting sector cannot be found during various read or write operations.
3	Not used, always clear (0).
4	Overrun. The computer failed to read data within a specified time limit and more data is coming from the disk.
5	Data error. A CRC error has been detected in the ID or data field.
6	Not used, always clear (0).
7	End of cylinder. This bit is set if the controller tries to access a sector beyond the last sector on the cylinder.

Table 17-7. Controller Status Register 2 Report

BIT	DESCRIPTION
0	Missing address mark. When data is read, the data address mark or deleted data address mark cannot be found.
1	Bad cylinder. This bit will be set if the track counter and the number recorded on the track do not agree and the cylinder number is equal to FFH.
2	Scan no good. This bit is set if the scan operation failed to find a matching record.
3	Scan equal. This bit is set if the scan operation found a perfectly matching record (the record and data were equal).
4	Wrong cylinder. This bit will be set if the track counter and the number recorded on the track do not agree.
5	Data error in data field. This bit is set if a CRC error is detected in the data field.
6	Control mark. This bit is set if deleted data address mark is detected during a read data or scan command.
7	Not used, always clear (0).

Table 17-8. Controller Status Register 3 Report

BIT	DESCRIPTION
0	Unit select 0. This bit reflects the signal state of pin 29, the unit select 0 output, which is not connected in this computer.
1	Unit select 1. This bit reflects the signal state of pin 28, the unit select 0 output, which is not connected in this computer.
2	Head address. This bit reflects the signal state of pin 27, the side select output signal.
3	Two side. This bit reflects the signal state of pin 34, the WPRT/2SID input signal. This is the same line that is checked by bit 6. In this case, the 2 SIDE signal is being checked, which is not used in this computer.
4	Track 0. This bit reflects the signal state of pin 33, the FLT/TRK0 signal. This is the same line that is checked by bit 7. In this case, the track 0 line from the disk drives is being checked, which is used in this computer.
5	Ready. This bit reflects the signal state of pin 35, the READY signal line from the disk drives.
6	Write protected. This bit reflects the signal state of pin 34, the WPRT/2SID input signal. This is the same line that is checked by bit 3. In this case, the write protect signal is being checked, which is used in this computer.
7	Fault. This bit reflects the signal state of pin 33, the FLT/TRK0 signal. This is the same line that is checked by bit 4. In this case, the fault line is being checked, which is not used in this computer.

Read deleted data — This command uses nine command bytes and returns seven status bytes. It reads data from a sector that has been marked with a deleted data mark as described in Table 17-9. With the exception of the first set of command codes, this command is identical to the read data command.

Table 17-9. Read Deleted Data Command

BYTE	DESCRIPTION
Write 1	Write the first set of command codes. Bits 0 - 4 = 0CH; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 17-4.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Write 3	Write the cylinder (track) number.
Write 4	Write the head address. This value is equal to the value of the head number (0 or 1) as defined in Table 17-4.
Write 5	Write the sector number to be read.
Write 6	Write the number of bytes per sector.
Write 7	Write the end of track, the final sector number to be read on a cylinder.
Write 8	Write the gap length, which is used for synchronization in read operations.
Write 9	Write the data length. If the bytes per sector value (write 6) is 0, this value is the data length to be read from or written to each sector.
Read 1	Read status register 0. Refer to Table 17-5.
Read 2	Read status register 1. Refer to Table 17-6.
Read 3	Read status register 2. Refer to Table 17-7.
Read 4	Read cylinder (track) number.
Read 5	Read head number.
Read 6	Read sector number.
Read 7	Read bytes per sector.

Write data — This command uses nine command bytes and returns seven status bytes. It writes data to one or more sectors to a two-track cylinder as described in Table 17-10.

Table 17-10. Write Data Command

BYTE	DESCRIPTION
Write 1	Write the first set of command codes. Bits 0 - 4 = 05H; bit 5 = 0; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 17-4.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Write 3	Write the cylinder (track) number.
Write 4	Write the head address. This value is equal to the value of the head number (0 or 1) as defined in Table 17-4.
Write 5	Write the sector number to be read.
Write 6	Write the number of bytes per sector.
Write 7	Write the end of track, the final sector number to be read on a cylinder.
Write 8	Write the gap length, which is used for synchronization in read operations.
Write 9	Write the data length. If the bytes per sector value (write 6) is 0, this value is the data length to be read from or written to each sector.
Read 1	Read status register 0. Refer to Table 17-5.
Read 2	Read status register 1. Refer to Table 17-6.
Read 3	Read status register 2. Refer to Table 17-7.
Read 4	Read cylinder (track) number.
Read 5	Read head number.
Read 6	Read sector number.
Read 7	Read bytes per sector.

Write deleted data — This command uses nine command bytes and returns seven status bytes. It writes data with a deleted data mark as described in Table 17-11. With the exception of the first set of command codes, this command is identical to the write data command.

Table 17-11. Write Deleted Data Command

BYTE	DESCRIPTION
Write 1	Write the first set of command codes. Bits 0 - 4 = 09H; bit 5 = 0; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 17-4.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Write 3	Write the cylinder (track) number.
Write 4	Write the head address. This value is equal to the value of the head number (0 or 1) as defined in Table 17-4.
Write 5	Write the sector number to be read.
Write 6	Write the number of bytes per sector.
Write 7	Write the end of track, the final sector number to be read on a cylinder.
Write 8	Write the gap length, which is used for synchronization in read operations.
Write 9	Write the data length. If the bytes per sector value (write 6) is 0, this value is the data length to be read from or written to each sector.
Read 1	Read status register 0. Refer to Table 17-5.
Read 2	Read status register 1. Refer to Table 17-6.
Read 3	Read status register 2. Refer to Table 17-7.
Read 4	Read cylinder (track) number.
Read 5	Read head number.
Read 6	Read sector number.
Read 7	Read bytes per sector.

Read a track — This command uses nine command bytes and returns seven status bytes. It reads the contents of an entire track from the index (timing) hole to the end of track, as described in Table 17-12.

Table 17-12. Read a Track Command

BYTE	DESCRIPTION
Write 1	Write the first set of command codes. Bits 0 - 4 = 02H; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = 0. Refer to Table 17-4.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.

Table 17-12 (Continued). Read a Track Command

BYTE	DESCRIPTION
Write 3	Write the cylinder (track) number.
Write 4	Write the head address. This value is equal to the value of the head number (0 or 1) as defined in Table 17-4.
Write 5	Write the sector number to be read.
Write 6	Write the number of bytes per sector.
Write 7	Write the end of track, the final sector number to be read on a cylinder.
Write 8	Write the gap length, which is used for synchronization in read operations.
Write 9	Write the data length. If the bytes per sector value (write 6) is 0, this value is the data length to be read from or written to each sector.
Read 1	Read status register 0. Refer to Table 17-5.
Read 2	Read status register 1. Refer to Table 17-6.
Read 3	Read status register 2. Refer to Table 17-7.
Read 4	Read cylinder (track) number.
Read 5	Read head number.
Read 6	Read sector number.
Read 7	Read bytes per sector.

Read sector ID — This command uses two command bytes and returns seven status bytes. It reads the values of the first ID field it is able to read as described in Table 17-13.

Table 17-13. Read Sector ID Command

BYTE	DESCRIPTION
Write 1	Write the first set of command codes. Bits 0 - 4 = 0AH; bit 5 = 0; bit 6 = FM/MFM bit; bit 7 = 0. Refer to Table 17-4.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Read 1	Read status register 0. Refer to Table 17-5.

Table 17-13 (Continued). Read Sector ID Command

BYTE	DESCRIPTION
Read 2	Read status register 1. Rever to Table 17-6.
Read 3	Read status register 2. Refer to Table 17-7.
Read 4	Read cylinder (track) number.
Read 5	Read head number.
Read 6	Read sector number.
Read 7	Read bytes per sector.

Format a track — This command uses six command bytes and returns seven status bytes. It formats an entire track as described in Table 17-14.

Table 17-14. Format a Track Command

BYTE	DESCRIPTION
Write 1	Write the first set of command codes. Bits 0 - 4 = 0DH; bit 5 = 0; bit 6 = FM/MFM bit; bit 7 = 0. Refer to Table 17-4.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Write 3	Write the number of bytes per sector.
Write 4	Write the number of sectors per track.
Write 5	Write the number of bytes in gap 3. During read/write commands, this values determines the time that the sync signal will be active.
Write 6	Write the pattern to be used in the data area of each sector.
Read 1	Read status register 0. Refer to Table 17-5.
Read 2	Read status register 1. Refer to Table 17-6.
Read 3	Read status register 2. Refer to Table 17-7.
Read 4	Read cylinder (track) number.
Read 5	Read head number.
Read 6	Read sector number.
Read 7	Read bytes per sector.

Scan equal — This command uses nine command bytes and returns seven status bytes. It reads the data on the disk and compares it against data supplied by the processor as described in Table 17-15. If the two sets of data are equal, the operation terminates.

Table 17-15. Scan Equal Command

BYTE	DESCRIPTION
Write 1	Write the first set of command codes. Bits 0 - 4 = 11H; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 17-4.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Write 3	Write the cylinder (track) number.
Write 4	Write the head address. This value is equal to the value of the head number (0 or 1) as defined in Table 17-4.
Write 5	Write the sector number to be read.
Write 6	Write the number of bytes per sector.
Write 7	Write the end of track, the final sector number to be read on a cylinder.
Write 8	Write the gap length, which is used for synchronization in read operations.
Write 9	Write the sector timing pattern. If this byte = 1, the sectors are contiguous. If this byte = 2, the sectors alternate.
Read 1	Read status register 0. Refer to Table 17-5.
Read 2	Read status register 1. Refer to Table 17-6.
Read 3	Read status register 2. Refer to Table 17-7.
Read 4	Read cylinder (track) number.
Read 5	Read head number.
Read 6	Read sector number.
Read 7	Read bytes per sector.

Scan low or equal — This command uses nine command bytes and returns seven status bytes. It reads the data on the disk and compares it against data supplied by the processor as described in Table 17-16. If the two sets of data are equal or of lesser value, the operation terminates.

Table 17-16. Scan Low or Equal Command

BYTE	DESCRIPTION
Write 1	Write the first set of command codes. Bits 0 - 4 = 19H; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 17-4.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Write 3	Write the cylinder (track) number.
Write 4	Write the head address. This value is equal to the value of the head number (0 or 1) as defined in Table 17-4.
Write 5	Write the sector number to be read.
Write 6	Write the number of bytes per sector.
Write 7	Write the end of track, the final sector number to be read on a cylinder.
Write 8	Write the gap length, which is used for synchronization in read operations.
Write 9	Write the sector timing pattern. If this byte = 1, the sectors are contiguous. If this byte = 2, the sectors alternate.
Read 1	Read status register 0. Refer to Table 17-5.
Read 2	Read status register 1. Refer to Table 17-6.
Read 3	Read status register 2. Refer to Table 17-7.
Read 4	Read cylinder (track) number.
Read 5	Read head number.
Read 6	Read sector number.
Read 7	Read bytes per sector.

Scan high or equal — This command uses nine command bytes and returns seven status bytes. It reads the data on the disk and compares it against data supplied by the processor as described in Table 17-17. If the two sets of data are equal or greater value, the operation terminates.

Table 17-17. Scan High or Equal Command

BYTE	DESCRIPTION
Write 1	Write the first set of command codes. Bits 0 - 4 = 1DH; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 17-4.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Write 3	Write the cylinder (track) number.
Write 4	Write the head address. This value is equal to the value of the head number (0 or 1) as defined in Table 17-4.
Write 5	Write the sector number to be read.
Write 6	Write the number of bytes per sector.
Write 7	Write the end of track, the final sector number to be read on a cylinder.
Write 8	Write the gap length, which is used for synchronization in read operations.
Write 9	Write the sector timing pattern. If this byte = 1, the sectors are contiguous. If this byte = 2, the sectors alternate.
Read 1	Read status register 0. Refer to Table 17-5.
Read 2	Read status register 1. Refer to Table 17-6.
Read 3	Read status register 2. Refer to Table 17-7.
Read 4	Read cylinder (track) number.
Read 5	Read head number.
Read 6	Read sector number.
Read 7	Read bytes per sector.

Recalibrate — This command uses two command bytes. It causes the head of the selected drive to be moved to track 0. The internal track counter for this drive is cleared. Refer to Table 17-18 for the description of the two command bytes.

Table 17-18. Recalibrate Command

BYTE	DESCRIPTION
Write 1	Write the first command code. Bits 0 - 7 = 07H.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bits 2 - 7 = not used. Refer to Table 17-4.

Interrupt Status — This command uses one command byte and returns two status bytes. It allows the user to sense interrupts that result during seek and recalibrate commands, as described in Table 17-19. The seek and recalibrate commands have no result phase to return a status report.

Table 17-19. Interrupt Status Command

BYTE	DESCRIPTION
Write 1	Write the command code. Bits 0 - 7 = 08H.
Read 1	Read status register 0. Refer to Table 17-5.
Read 2	Read cylinder (track) number.

Specify — This command uses three command bytes. It sets the initial values of each of three internal timers as described in Table 17-20. These are the head unload time, the step rate time, and the head load time.

Table 17-20. Specify Command

BYTE	DESCRIPTION
Write 1	Write the command code. Bits 0 - 7 = 03H.
Write 2	Write the head unload time and step rate time. Bits 0 - 3 = head unload time in 16 ms increments. Bits 4 - 7 = step rate time in 1 ms increment.
Write 3	Write the DMA mode flag and the head load time. Bit 0 = DMA mode flag: clear (0) = DMA mode, set (1) = non-DMA mode. Bits 1 - 7 = head load time in 2 ms increments.

Drive Status — This command uses two command bytes and returns one status byte. It reports back the status of the specified disk drive as described in Table 17-21.

Table 17-21. Drive Status Command

BYTE	DESCRIPTION
Write 1	Write the first command code. Bits 0 - 7 = 04H.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Read 1	Read status register 3. Refer to Table 17-8.

Seek — This command uses three command bytes. It causes the specified disk drive to seek to the specified cylinder as described in Table 17-22. The specified drive's track counter is incremented or decremented according to the number of step pulses sent to the drive.

Table 17-22. Seek Command

BYTE	DESCRIPTION
Write 1	Write the first command code. Bits 0 - 7 = 0FH.
Write 2	Write the second set of command codes. Bits 0 - 1 = drive unit number (0 - 3); bit 2 = head number; bits 3 - 7 = not used. Refer to Table 17-4.
Write 3	Write the new cylinder number to which the heads are to be stepped.

NOTE: Any other command codes that are sent to the data register will result in a "NOP" operation and one status byte with a value of 80H will be returned.

For more information on programming this controller, refer to the NEC's µPD765A/7265 data sheet.

Hard Disk System

The Portable Computer contains all necessary circuitry to support the internal hard disk drive. The hard disk controller contains firmware that responds to control signals provided by the BIOS firmware in the computer. The control functions are discussed in other portions of this manual. Refer to the discussion on interrupts in Chapter 10.

The hard disk drives used in the Portable Computer series are 3.5 inch size format drives. The formatted capacity of the drive is in excess of 20 MB of data. The drive uses very little power and is internally protected by the computer system in two specific ways.

The heads used on a hard disk drive are very sensitive. To protect the heads, drive activity is monitored. If access to the drive is not requested by the system within 5 seconds, the heads will be retracted to the park zone. This decreases the chance of accidental damage to the heads or the disk surface. If the system detects a low power condition, access to the hard disk will not be permitted. This reduces the chances of lost data during write operations.

NOTE: The MS-DOS version supplied with this computer provides a **Mode Motor** command for powering down the hard disk drive after a period of non-use. This conserves battery power. Refer to the MS-DOS manual for further information.

Part IV

General

Chapter 18

Specifications

CPU

Processor	80C88 CMOS 16-bit processor.
Type	16-bit internal.
Clock speed	Switch selected: 4.77 MHz (IBM standard) and 8 MHz.
Coprocessor	Option: 8087-2 numeric data coprocessor.
Memory	640K total: 128K dynamic RAM (64-kilobit × 4 devices) and 512K dynamic RAM (256-kilobit × 1 devices).
Display	Electroluminescent backlit Supertwist Liquid crystal (270° twist).
Capacity	80 × 25 characters in text mode. 640 × 200 pixels in graphics mode. PC-compatible in normal text and graphics modes.
Sound	Miniature transducer.

Input/output

Serial port	Asynchronous serial RS-232C port (DB-9 connector). One start bit; 7- or 8-bit word length; one or two stop bits; selectable baud rates of 110, 150, 300, 600, 1200, 2400, 4800, 9600, or 19,200 baud; RD, CTS, DSR, CD signals recognized; TD, RTS, DTR control signals generated; half- or full-duplex operation.
Parallel port	Centronics-type parallel output port (DB-25 connector).
Video	Composite (monochrome) video and RGB (color) video with intensity signals from a single 9-pin connector.
Modem (optional)	RJ-11 modular line and telephone connectors. 1200 or 2400 baud, Hayes AT command set compatible.

Disk drives

Floppy disks	3.5-inch double-sided, double-density floppy disk drives, 720K formatted capacity each drive, 135 tpi, nine sectors per track. Write-protection recognized.

	5.25-inch (optional, external) double-sided, double-density floppy disk drives, 360K formatted capacity each drive, 48 tpi, nine sectors per track. Write-protection recognized.
Hard drive	3.5-inch 20M hard disk drive, depending upon computer model.
Keyboard	78 keys: 60-key alphanumeric typewriter arrangement, four multifunction keys, four cursor control keys, and ten function keys. Full PC keyboard compatibility maintained by using mode switching and multiple keys to duplicate keypad and special function key operation.
Power requirements	
AC Powers	Either 115V ± 17% (50/60 Hz) or 220V ± 15% (50/60 Hz) (autosensing adapter/charger unit included). 8 hour recharge.
Battery Powers	12.5 volt (2.5 AHr) NiCad battery pack (removable) standard - FDD systems. Thermocontrolled charging with overcharge and short circuit protection Low battery indicator. Battery life 3.1 hours (heavy user duty cycle). Computer is operational while internal battery is recharging. Battery life - minimum 200 charge/discharge cycles. Shelf life up to 5 years. Battery pack weight - 1.0 lbs. 12.5 volt (4.0 AHr) NiCad battery pack (removable) standard - FDD systems w/ STN-LCD and HDD systems. Thermo-controlled charging with overcharge and short circuit protection Low battery indicator. Battery life 3.1 hours (heavy user duty cycle). Computer is operational while internal battery is recharging. Battery life - minimum 200 charge/discharge cycles. Shelf life up to 5 years. Battery pack weight - 2.2 lbs.
Environment	
Operating	40° - 104° F (10° - 40° C) at 20% - 80% relative humidity (non-condensing).
Storage	–13° - +132° F (–25° - +56° C) at 20% - 80% relative humidity (non-condensing).

Dimensions	Computer: 12.2" wide x 12.2" deep (front to back) x 3.1" tall at back (31.28 cm x 31.28 cm x 7.95 cm). Computer with 2.5 AHr battery enclosure: 12.2" wide x 15.2" deep x 3.1" tall (31.28 cm x 3641 cm x 7.95 cm). Computer with 4.0 AHr battery enclosure: 12.2" wide x 14.3" deep x 3.1" tall (31.28 cm x 39.23 cm x 7.95 cm).

Chapter 19
In Case of Difficulty

This chapter will help you identify and solve common problems you can experience with your computer. This chapter also describes power-up and user-executed tests and error messages.

NOTE: This chapter does not cover the repair of problems requiring qualified service. For repair of the hardware, have the computer serviced by a qualified service center.

Solving Computer-Related Problems

This computer is a very complex device. Anytime you face a complex problem, you can solve that problem by breaking it into smaller, manageable parts. The following procedure will help you break down a problem you might have with the computer into a manageable task. This procedure takes the form of questions and answers. Your responses will lead you through the computer until you locate the problem and resolve it.

Problem-Solving Procedure

Before you start, disconnect all peripherals. Refer to Chapter 2 and configure the computer for operation from the LCD (internal display). Make sure the drive count equals the number of floppy disk drives in your computer and drive A location is internal.

1. Turn on the computer. Did it turn on? If your answer is no, go to step 2. If your answer is yes, go to step 7.

2. Turn off the computer and plug in the external adapter. Wait a few moments. Turn on the computer. Did it turn on? If your answer is no, go to step 3. If your answer is yes, go to step 6.

3. Turn off the computer and refer to Chapter 2. Check switch section 6. Is it set for the internal (LCD) display? If your answer is no, set switch section 6 for the internal display and return to step 1. If your answer is yes, go to step 4.

4. Remove the battery. Plug in the adapter and turn on the computer. Did it turn on? If your answer is no, go to step 5.

   If your answer is yes, the battery could be bad. Refer to Chapter 3 and review the material on the battery. Replace the battery with a charged battery and return to step 1.

5. Check the adapter's power source. You can do this by plugging an appliance or a lamp into the same outlet and trying it. If the other device works, you have a problem with either the adapter or the computer. Proceed to "Seeking Service" later in this chapter. If the other device does not work, replace the battery, use a good power outlet, and return to step 2.

6. Since the adapter works, you could have a discharged or faulty battery. Read the material in Chapter 3 about the battery. If you have a faulty battery, replace it.

7. Does the computer make any strange noise, feel hot, or smell like it is burning? If your answer is yes, turn the computer off. Proceed to "Seeking Service" later in this chapter. If your answer is no, go to step 8.

8. Does the backlight come on when you turn on the computer? If your answer is yes, go to step 10. If your answer is no, go to step 9.

9. Turn the computer off and refer to Chapter 2 and check configuration switch section 6. Make sure it selects the internal (LCD) display. Is the switch set for the internal display? If your answer is no, set the switch for internal (LCD) display and return to step 8. If your answer is yes, turn off the computer, unplug it, and proceed to "Seeking Service" later in this chapter.

10. Do you see any error message on the screen? If your answer is yes, go to "Error Messages" in this chapter. If your answer is no, go to step 11.

11. Does the computer attempt to boot the disk in drive A within a minute of when you turn on the

computer? If the answer is no, go to step 12. If your answer is yes, go to step 16.

12. Turn off the computer. Refer to Chapter 2 and check switch section 2 for drive A location. Is drive A location set for internal? If your answer is no, change the switch to select drive A location internal, turn on the computer, and return to step 10. If your answer is yes, go to step 13.

13. Press CTRL-ALT-INS to reset the computer to the Monitor program. Wait a minute or until a message appears on the screen. Refer to Chapter 4 for information on the Monitor program. Does the Monitor program prompt appear on the screen? If your answer is yes, go to "User-Executed Tests" in this chapter. If your answer is no, go to step 14.

14. Does an error message appear on the screen? If your answer is yes, go to "Error Messages" in this chapter. If your answer is no, go to step 15.

15. Turn off the computer. Refer to Chapter 2 and check configuration switch section 6 for the internal (LCD) display. Is switch section 6 set for the internal (LCD) display? If your answer is yes, turn off the computer and proceed to "Seeking Service" later in this chapter. If your answer is no, change the switch to select the internal (LCD) display, turn on the computer, and return to step 13.

16. Place a bootable disk in drive A and allow the computer to boot it. After the disk drive runs for a minute, do you see anything on the screen? If your answer is no, go to step 17. If your answer is yes, go to step 18.

17. Is the disk drive still running? If your answer is yes, remove the disk and try the backup disk if you have one. If you do not, repeat step 16 with a different bootable disk. If your answer is no, turn off the computer and proceed to "Seeking Service" later in this chapter.

18. Do you see an error message? If your answer is yes, go to "Error Messages" later in this chapter. If your answer is no, go to step 19.

19. Refer to your operating system or application program documentation. Did the disk boot normally? If your answer is yes, go to step 20. Otherwise, go to step 21.

20. Your problem could be the configuration switch settings, a peripheral attached to the computer, or a circuit not tested in this procedure. Review the installation and configuration section of Chapter 2. If you still experience problems, proceed to "Seeking Service" in this chapter.

21. You probably have a software- or operator-oriented problem. Proceed to "Error Messages" later in this chapter.

Seeking Service

Before you contact a qualified service center or your dealer, look over the following questions. Give your responses to these questions to your dealer or the service center. Be as specific as possible.

NOTE: It is possible that, as you go through these questions, you may discover a correctable problem. If you do, correct the problem using the problem-solving procedure at the beginning of this chapter.

1. Did the computer ever work properly? By determining whether the computer once worked correctly, you can determine if some change has taken place that could affect the operation. Such changes can involve modifications to the hardware or software, or perhaps a new operator. Sometimes you can forget a commonly-used procedure over a weekend or during a vacation. If you suspect it is an operator-oriented problem, review the documentation for the application programs that are not running correctly.

2. Can you turn the computer on? If so, how long does it run before it fails? By knowing how long you can use the computer, and whether the computer has ever functioned correctly, you can determine whether the problem is operator-, software-, or hardware-oriented. If you are not sure, proceed to "Error Messages" later in this chapter.

3. Was the computer or any of its peripherals dropped or damaged in any way? Describe what happened when you have the computer or peripheral serviced.

4. Were any connecting cables, including power cords, damaged, run over, crushed, crimped, sharply bent, or abused in any other way? Check all cables. They can cause problems if damaged.

5. Has anyone exposed the computer to liquids of any kind, including an extremely damp environment? Rain, snow, or extremely damp environments can affect normal operation. Make sure the computer is dry before you use the problem-solving procedure.

6. Can you detect a burned smell from any part of the computer or its peripherals? If you can, describe the odor when you have the computer or peripheral serviced.

7. Does the computer make a loud noise when you turn it on? If it does, describe the noise when you contact the service center.

8. Check the following. If you find any damage, correct the problem or have the computer serviced.

   - Computer installation. Review the installation and configuration information in Chapter 2 of this manual.

   - Cables. Make sure all cable connections are secure.

   - Disks. Replace any physically damaged disks.

   - External power supply. Make sure the supply is working by disconnecting the battery and using only external power. Wiggle the external supply's cables. Is operation interrupted? If so, you have a faulty cable. Replace faulty cables or have the external power supply serviced.

Common Problems

NOTE: You cannot use the computer without an operating system and application software. The only built-in programs are the user-executed tests described later in this chapter.

Other common problems are:

PROBLEM: "I can't use drive A."
CHECK: You may have switch section 2 set for the wrong location. If you want to use the internal 3.5-inch drive as drive A, you must set drive A location for internal. If you want to use the external disk drive as drive A, you must set drive A location for external. Refer to Chapter 2 for more information on this switch.

PROBLEM: "The program runs too fast."
CHECK: You may have switch section 3 set for the 8 MHz clock speed. Most software will run at the faster clock speed and so you usually set this switch for 8 MHz. However, some of your programs may require the slower speed. As a result, some displays or messages might not stay on the screen long enough for you to read them. If this happens, set this switch for the slower speed. Refer to Chapter 2 for more information on this switch.

PROBLEM: "The computer won't use one (or more) of the disk drives."
CHECK: Switch sections 7 and 8 select the number of disk drives that you can use with the computer. If you set these switches for too few drives, you will not be able to use all of the disk drives. If you set these switches for too many drives, the computer can generate an error message. Refer to Chapter 2 for more information on this switch.

PROBLEM: "The computer doesn't operate very long on the battery."
CHECK: This usually results from misuse of the battery. Chapter 3 contains a complete discussion of battery care and maintenance. Review this material. If this complaint persists, replace the battery.

PROBLEM: "I can't get the computer to turn on."
CHECK: Make sure the computer has a fully charged battery or plug it into an operating external adapter. If you can, check the output voltage of the battery; it must measure at least +12 VDC. Check the power source of any adapter being used and, finally, check the output of the adapter.

PROBLEM: "The display characters are too wide."
CHECK: If you set switch section 5 for 40 character display width, each line will display only 40 characters. If a line has more than 40 characters, the extra characters appear on the next line. Refer to Chapter 2 for more information on this switch.

NOTE: Some programs use a 40-character line during normal operation. As a result, the program can leave the computer in this condition. Normally, the next program you run will correct the problem. If it does not, you may have a software-oriented problem.

PROBLEM: "I can't see anything on the display."
CHECK: Make sure you turned on the computer. If you set switch section 6 for external operation, you will not see any display on the internal (LCD) display. If you are not going to use an external monitor, make sure the configuration switch selects the internal display. Refer to Chapter 2 for more information on this switch.

Also, move the CONTRAST and BRIGHTNESS controls to see if they affect the display and toggle the video output control (press FN-F10). If the computer still does not display anything, press CTRL-ALT-INS to reset the computer to the Monitor program and adjust the controls. If you still cannot get a display, toggle the video output control and connect an external monitor.

If you have RGB video output but cannot toggle the video output control to produce a display on the LCD, have the computer serviced.

PROBLEM: "The screen is too dark."
CHECK: Adjust the brightness and contrast controls for the best display. If this does not solve the problem, have the computer serviced.

PROBLEM: "The characters on the screen are too light."
CHECK: Adjust the brightness and contrast controls for the best display. If this does not solve the problem, have the computer serviced.

PROBLEM: "I can't see all of the display."
CHECK: Any program that checks for a color controller will adjust for color operation on this computer. In this case, the computer might not display one or more of the colors with enough contrast to the background to be readable. If this is the case, adjust the brightness and contrast controls and press FN-F8 or FN-F9 to toggle the LCD display palette. If the problem persists, check the display with an external monitor.

PROBLEM: "The disk won't work correctly."
CHECK: Five conditions can create disk problems: incompatible operating system or format, unformatted disk, formatted data disk, magnetically altered disk, and physically damaged disk. In all cases, you must either use a different disk or reformat the disk for your application.

Incompatible operating system or format — Most operating systems use different formats on the disk. One operating system cannot read disks created by a different operating system without special software.

Unformatted disk — New disks are not formatted. If you attempt to use an unformatted disk, the drive will not have any recorded references on the disk and the operating system will generate an error message that indicates a bad disk.

Formatted data disk — Disks that are formatted as data disks display No System on the screen and lock up the computer when you boot them. You must use a disk that contains the operating system on it to boot the computer.

Magnetically altered disk — If you handle disks carelessly, you can alter the magnetic patterns on them. When this happens the computer can display one or more disk error messages. Review the disk care information in Chapter 3 and avoid placing disks where sunlight or magnetic fields can affect them.

Physically damaged disk — You can damage disks with fingerprints, dust, dirt, moisture, liquids, and rough handling. Physical damage to a disk that is not readily apparent can cause the computer to display disk error messages. Avoid using damaged disks because they can also damage the disk drives.

PROBLEM: "The keyboard doesn't work correctly."
CHECK: This computer has a unique keyboard that uses several different operating modes to emulate a standard PC-compatible keyboard. You can usually solve this problem by changing keyboard modes. For a complete discussion on the keyboard modes, refer to Chapter 3.

PROBLEM: "My program does not work correctly."
CHECK: Four conditions can cause a particular program not to work correctly on this computer: incorrect installation, invisible colors, incompatible software, and software bugs.

Incorrect installation — Some programs require specific hardware to operate correctly. If the program is copy-protected, use caution in making backups, particularly if you are transferring programs from 5.25-inch disks to the 3.5-inch disks used in this computer. Some copy-protected software will permit a restricted number of copies for backup purposes, while other software requires you use only the distribution copies. Other copy-protected software requires that you place the distribution disk in drive A.

Use the configuration switch to configure the external drive for drive A.

Invisible colors — The video section of this computer tells software that it is a CGA type of display, indicating that it will support a color display. Some software packages use foreground and background colors that do not show up well on the built-in display. If this is the problem, adjust the brightness and contrast controls and toggle the screen display palette with the FN-F8 and FN-F9 keys to obtain the most readable display.

Incompatible software — If the software will not work and has never worked on this computer, try it on another PC-compatible computer. If it works on that computer, you have a compatibility problem with the custom circuits used in this computer. The software may be expecting specific hardware not present in the computer, or the hardware may be operating in a mode not normal for the software. Contact the software's manufacturer for assistance.

Software bug — Software, no matter how extensively tested, often contains bugs or error conditions that cause it to operate incorrectly. If you positively identify the software as the cause of the problem, contact the software's manufacturer for assistance.

Power-Up Self-Tests

During powerup, the Monitor program performs a number of tests to make sure the computer is ready to function. The Monitor program initializes all circuits and synchronizes all disk drives with the electronics. If the program detects a malfunction, it will attempt to send one or more messages to the screen to alert you of the problem. Refer to "Error Messages" later in this chapter.

These power-up self-tests check the Monitor program ROM, memory, processor, interrupt controller, and disk controller. While the tests are not all-inclusive, they do let you know that the computer is ready to run. When finished with the tests, the computer attempts to read the first sector from drive A and execute the code. This autoboot procedure requires a disk in drive A or a bootable partition in the hard disk system. Otherwise, after about 10 seconds in floppy disk systems or 30 seconds in a hard disk system, the computer will display an error message. Refer to "Error Messages" later in this chapter.

User-Executed Tests

Among the Monitor program's commands are user-executed tests. These tests are more comprehensive than the power-up self-tests and can help you quickly evaluate problems in memory, the disk drives, and the keyboard.

The user-executed tests generate the same error messages that the power-up self-tests generate. If these tests generate an error message, refer to "Error Messages" later in this chapter.

Test Menu

The Monitor program's command summary identifies the tests as "Extended diagnostics." To display the menu from which you run the tests, enter TEST at the Monitor program prompt and press the ENTER key. Figure 19-1 illustrates the menu.

```
CHOOSE ONE OF THE FOLLOWING

1.  DISK READ TEST
2.  KEYBOARD TEST
3.  MEMORY TEST
4.  POWER-UP TEST
5.  EXIT
ENTER YOUR CHOICE:
```

Figure 19-1. Test Menu

To execute a test, press the number that corresponds to the desired test. The Monitor program starts executing the selected test immediately.

The Disk Read Test

The disk read test continuously reads the first sector on track 0 of the disk in the default drive. The computer must read this sector successfully to boot the disk. Even though the boot track is being read, the disk does not need an operating system on it for this test. For example, formatted data disks have valid data in the first sector of the disk. This sector contains a short routine that, when booted and executed, displays No system and locks up the computer. This message tells you that you have a formatted disk that does not contain an operating system.

The disk read test is non-destructive. If the disk you are testing has an operating system or data on the first track, the test will not erase or destroy the information. During operation, the test displays the test count, indicating how many times the program read the first track.

By continuously reading the boot sector, you can determine if you have a reliable disk/drive combination. You can attribute errors that occur during this test to the disk, the drive, or both. If the test immediately displays an error message, use a known formatted disk before assuming you have a hardware-oriented error.

You can also use this test to discharge the battery. A fully charged battery will run about three hours using this test before the low power indicator starts flashing. The larger battery will run about five hours.

The Keyboard Test

The keyboard test allows you to check most of the keys on the keyboard and generate the key scan codes. A complete list of all scan codes is in the technical manual for this computer. Some of the keys will cause the computer to exit the test rather than produce a display.

The test displays several items of information. Each time you press a key that produces a printable ASCII character, it fills the display. The program displays the code produced by the key in the upper-right corner of the screen. If the key does not produce a code, the last code displayed remains on the screen.

The Memory Test

The power-up self-tests check only the first and last 64K bank of memory. The user-executed memory test checks all system memory and video memory. While this test is checking video memory, various patterns appear on the screen.

The test produces a clicking sound and displays a number indicating the current memory test count. Each time the test finishes, the program increments the test count. The upper-right corner of the screen displays the current 64K bank of memory being tested in hexadecimal format. If the program detects an error, the test stops and the program displays information about the error. This information includes the address where the program detected the error and the bit number. Report this address when you contact a technician for service.

The Power-Up Test

The power-up test repeatedly runs the power-up self-tests. These tests, described earlier in this chapter, include a check of all major circuits. While the power-up self-tests are not all-inclusive, you can detect some random fault conditions using this test. The disk-based diagnostics, available separately, are more useful since they exhaustively and repeatedly test key elements of the computer. For more information on the disk-based diagnostics, refer to the disk-based diagnostics manual.

Exiting the Test Menu

All user-executed tests display a status screen while running. To exit any test, press the ESC key. Except for the keyboard test, the escape key first halts the test. Pressing it a second time returns the program to the test menu. The ESC key exits directly to the test menu when you press it during the keyboard test.

Once you have exited to the test menu, press the 5 key to return to the Monitor program. The Monitor program prompt reappears on the screen.

NOTE: Do not use the CTRL-ALT-INS or CTRL-ALT-DEL key sequence to exit any of the tests. Exiting directly to the Monitor program can leave the computer in an unknown state and can produce unpredictable results.

Error Messages

You can group computer problems into three categories: operator-, software-, or hardware oriented.

Operator-oriented problems are quite common. They usually result in a condition that you can correct. Pressing the wrong key, using the wrong disk, or using partially-erased disks are examples of operator-oriented problems.

Software-oriented problems can come from one of two sources: locally-produced or modified software or commercially-sold unmodified software.

Hardware-oriented problems are the least common problems you will have. The computer should provide uninterrupted service for many years. If you use undue force, drop the computer, or spill something into the computer, you can damage the computer. If you have damaged equipment or it fails to operate correctly, take it to a qualified service center.

You can group the error messages generated by the computer during the power-up and user-executed tests into the same three categories. Table 19-1 describes each error message and the category into which it falls. When an error message falls into more than one category, the first-mentioned category is the usual source of the problem.

NOTE: If you configure the computer for an external monitor, you will not see any messages on the computer's built-in display. Refer to Chapter 2 for information on the configuration switches.

Operator-Oriented Problems

Most operator-oriented problems fall into three categories: incorrect hardware configuration, incorrect operation, or disk problems.

Incorrect hardware configuration — The operation of this computer is very flexible. As a result, you can set the computer's configuration switch sections for several different options. However, some of these options may not be correct for your situation. Refer to "Common Problems" for more information.

Incorrect operation — The most common operator error is pressing the wrong keys in a program or command. When you use a command or respond to a prompt in a program, make sure you use the correct sequence of operations. If you experience difficulty with an application program, carefully review the operating instructions that accompany that software.

Incorrect operation can cause one of three error messages. The first error message can occur after the computer attempts to autoboot a disk:

MESSAGE:
`+++DISK ERROR: Drive not ready! +++`
EXPLANATION: The computer generates this message when it attempts to boot from a disk drive that does not contain a disk. Make sure you removed the shipping insert and inserted the disk correctly.

Table 19-1. Error Messages

MESSAGE	CATEGORY
`DISK ERROR: Bad disk controller!`	Hardware/Operator
`DISK ERROR: CRC error!`	Operator/Hardware
`DISK ERROR: DMA overrun error!`	Hardware/Operator
`DISK ERROR: Drive not ready!`	Operator/Hardware
`DISK ERROR: Invalid address mark detected!`	Operator/Hardware
`DISK ERROR: Sector not found!`	Operator/Hardware
`DISK ERROR: Seek error!`	Operator/Hardware
`ERROR: Calendar failure!`	Hardware
`ERROR: CPU failure!`	Hardware
`ERROR: Divide by zero!`	Software/Hardware
`ERROR: Invalid command!`	Operator
`ERROR: Non-maskable interrupt received!`	Hardware
`ERROR: Overflow!`	Software/Hardware
`ERROR: RAM failure!`	Hardware
`ERROR: ROM checksum failure!`	Hardware
`ERROR: Timer interrupt failure!`	Hardware
`I/O error`	Operator/Hardware
`Invalid command!`	Operator
`No system`	Operator

A faulty disk can also cause this problem. Therefore, if this message appears when a disk is in the drive, try another disk or try booting from the other disk drive.

The other two error messages can occur after you manually defeat autoboot and use the Monitor program to boot a disk:

MESSAGE: `^ Invalid Command`
EXPLANATION: This message indicates that you pressed the wrong keys in your command line. The caret (^) points to the first syntax error in your command line. This error does not lock up the system; therefore, the command can be reentered or another command entered in its place. Refer to Chapter 4 and Chapter 6 for the syntax for Monitor program commands.

MESSAGE:
`+++DISK ERROR: Seek failure! +++`
EXPLANATION: The Monitor program generates this message approximately five seconds after you attempt to boot from a disk drive not attached to the computer. Because the operation that creates this error establishes an invalid drive as the default disk drive, you must specify a valid drive in the boot syntax after receiving this error message.

Disk problems — Except for those actions that result in physical damage, you can usually fix most operator-oriented disk problems if you keep backup disks of important information and programs. You can attribute the following error messages during autoboot to faulty disks:

MESSAGE:
`+++DISK ERROR: Drive not ready! +++`
EXPLANATION: The computer generates this message when the it attempts to boot from a disk drive that does not contain a disk. However, a faulty disk can also cause this message. Try to boot the disk from another drive or try another disk.

MESSAGE: `No system`
EXPLANATION: Data disks contain a short program that displays this message when you try to boot them. You must use a bootable system disk to boot the computer. Refer to your operating system documentation for more information.

MESSAGE: `I/O error`
EXPLANATION: The computer generates this message when you use a partially erased disk in the boot drive. A powerful magnetic field can partially erase a disk placed near it. You must reformat the disk before you can use it.

MESSAGE:
`+++ DISK ERROR: Invalid address mark detected! +++`
EXPLANATION: The computer generates this message when you use a new, unformatted disk in the boot drive. You must format a new disk before you can use it.

MESSAGE: `+++ DISK ERROR: (any message)`
EXPLANATION: A partially erased disk can generate a number of different errors. The location of the affected tracks determines the message. Sometimes, the disk can complete part of the boot process and cause the computer to lock up without generating any error. To get out of the error condition, you may have to turn the computer off. Try booting the computer from a disk that you know is good.

Replace or reformat partially erased disks and damaged disks. You will lose any work that is on the disk. Therefore, make periodic backups of all disks that you use in the computer.

You can also experience three other operator-oriented disk problems: disk ejection problems, disk insertion problems, or the disk wrong side up.

Disk ejection problems — The disk latch may not release the disk. If the disk will not eject when you press the eject button, contact a qualified service center.

Disk insertion problems — The disk drive disk latch can become latched. This will keep the disk from sliding all the way into the drive. This can happen if you placed the disk in the drive wrong end first. When this happens, you will see the metal disk cover sticking slightly out of the drive. To correct the problem, remove the disk and press the disk eject button. This will release the latch so you can insert the disk fully into the drive.

The latch can also bind for some unknown reason. This is very unusual, but you can cause it by forcibly attempting to insert a disk into the computer incorrectly, or by attempting to push the disk drive into the operating position with a partially inserted disk. If this happens, remove any disk in the drive and press the eject button. Read the instructions on inserting disks in Chapter 1 and attempt to place a disk into the drive correctly. If the latch remains latched and you cannot get it released using this procedure, contact a qualified service center.

CAUTION: Do not forcibly attempt to insert a disk. If you force a disk into the drive, you can damage the disk or disk drive. If you force the disk drive into its operating position, particularly if the disk is not fully in the drive, you

can jam the disk into the computer. A service technician will have to disassemble the computer to repair the disk drive.

Disk wrong side up — If the disk slides all but about a half an inch into the drive and wants to eject, then the disk is upside down. Remove the disk and note the position of the disk hub. It should face down, toward the keyboard.

Software-Oriented Problems

During the power-up sequence, a partially erased system or data disk can cause a software-oriented error. Refer to the error messages described in "Operator-Oriented Problems" earlier in this chapter.

If you think you have a partially-erased disk, try the backup disk. If the backup disk works properly, then reformat the questionable disk before using it again. If the backup does not work, read "Seeking Service" earlier in this chapter.

Most other software-oriented problems occur with commercial or custom software, rather than software you use to boot the computer. Incorrectly operating software can stem from a number of causes, described in "Common Problems" earlier in this chapter.

Hardware-Oriented Problems

The following error messages occur during the power-up sequence and indicate a computer failure. If any of the messages appear, attempt to clear the message by turning the computer off and then on again or, if indicated in the explanation, try a different disk. If you cannot clear any of the following messages, note the message and refer to "Seeking Service" earlier in this chapter.

MESSAGES:
```
+++ DISK ERROR: Bad disk controller! +++
+++ DISK ERROR: DMA overrun error! +++
```
EXPLANATION: These error messages usually indicate a hardware failure, caused by defective circuits in the computer. These messages can also result from an incorrectly formatted disk or a damaged disk. If you suspect a disk problem, try booting from a different disk.

MESSAGES:
```
+++ ERROR: ROM checksum failure! +++
+++ ERROR: CPU failure!+++
+++ ERROR: RAM failure!
address:XXXX:YYYY+++
+++ ERROR: Timer Interrupt failure! +++
+++ ERROR: Non Maskable Interrupt Received! +++
```
EXPLANATION: These error messages indicate a hardware failure. Refer to "Seeking Service" earlier in this chapter.

MESSAGE:
```
+++ ERROR: Calendar Failure! +++
```
EXPLANATION: This error indicates that the real-time clock circuit has failed. This can result from the machine being off for more than three days after you remove the battery. If the real time clock integrated circuit does not power up in a running state, the Monitor program may incorrectly interpret this as a hardware failure. Turn the computer off and then on again. Refer to the operating system documentation and reset the clock using the appropriate DOS command.

MESSAGES:
```
+++ ERROR: Divide By Zero! +++
+++ ERROR: Overflow! +++
```
EXPLANATION: These errors result from an illegal processor state. The divide by zero and overflow errors are normal during mathematic operations in application programs. However, if these errors occur during power-up, they indicate hardware failures.

Disk-Based Diagnostics

The optional disk-based diagnostic package (Model CB-3163-38) is a supplemental testing package available for this computer. These tests contain comprehensive routine to check both the system core circuits and the interfaces required to operate peripheral devices. These tests can be very useful it you should encounter an difficulties with the computer.

The ROM-based tests check all functions of the hardware required to load and run the disk-based diagnostics. For a complete discussion on how to configure and execute disk-based diagnostics, refer to the disk-based diagnostic manual.

Chapter 20

Adding to Your Computer

Up to 1 megabyte of RAM with EMS (Extended Memory Specification) RAM can be added to your computer. However, this memory can only be installed by a qualified service technician. Contact you nearest Zenith Authorized service center for information about installing EMS memory in your computer.

There are only two options that you can add to your computer without voiding your warranty, the 8087 numeric coprocessor and the internal modem. Instructions for installing the internal modem are supplied with modem kits purchased from Zenith Data Systems. The following steps will guide you through the installation of the numeric coprocessor.

Numeric Coprocessor Installation

NOTE: To avoid any possible damage to either the computer or the numeric coprocessor, it is recommended that you have your nearest Zenith authorized service center install the coprocessor for you. They will install it at no charge.

To install a numeric coprocessor in your computer:

1. Turn the computer off. If the external AC power supply is attached to the computer, disconnect it.
2. Close and latch the lid.
3. Place a soft cloth on your work surface and turn the computer upside down on your work surface.
4. Carefully pry up on the configuration switch cover, as shown in Figure 20-1.

CAUTION: ICs are complex electronic devices that can be damaged by static electricity. Once you remove the IC from its protective foam packing, do not lay the IC down or let go of it until it is installed in its socket.

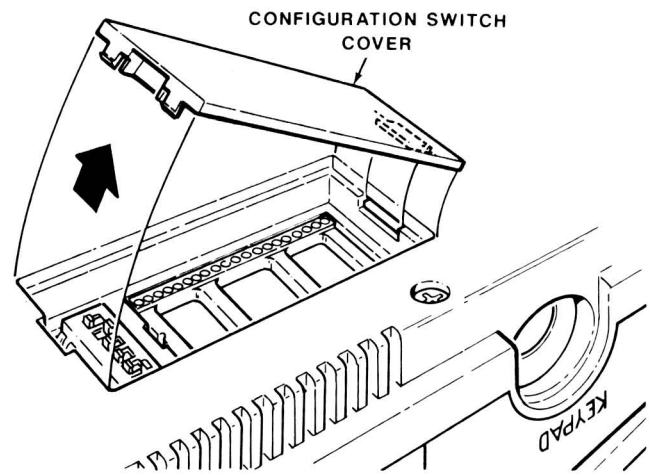


Figure 20-1. Removing the Configuration Switch Cover

5. The pins on the IC may be bent out at an angle so they do not line up with the holes in the IC socket. The pins must be bent into alignment; incorrect installation may damage the IC pins or the socket, causing intermittent and unreliable contact. To bend the pins of an IC, hold the IC in one hand and place you other hand on your work surface before you touch the IC to your work surface. This will equalize the static electricity between the work surface and the IC. Next, lay the IC on its side, as shown in Figure 20-2, and roll it very carefully toward the pins to bend them in line. Then turn the IC over and bend the pins on the other side in the same manner.

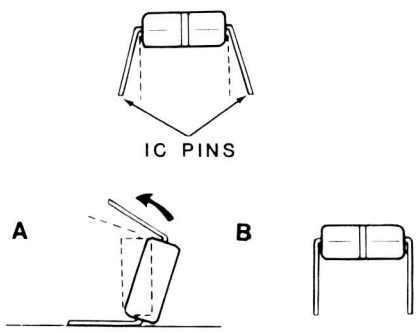


Figure 20-2. Rolling the IC to Bend the Pins

6. Make sure that the pin 1 end of the IC is positioned over the index mark on the circuit board (not the socket) as shown in Figure 20-3.

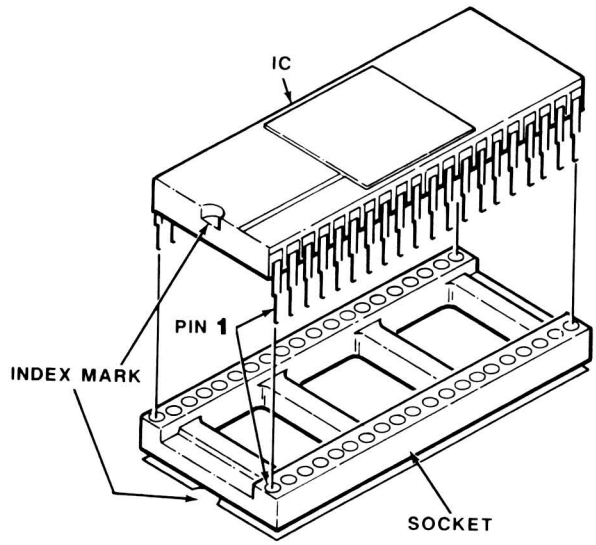


Figure 20-3. Orientation and Alignment of the IC

7. Press the IC firmly into the socket.

NOTE: A pin can become bent under the IC and will appear to be correctly seated in its socket. If a malfunction occurs while testing, examine the IC (and remove it, if necessary) to be certain that all pins are correctly inserted.

8. Replace the configuration switch cover.

Index