Operations/DOS

Reference Guide

Sr. Partner™
Portable Computer

Panasonic

SOFTWARE LICENSE AGREEMENT

THE SOFTWARE PROGRAM PROVIDED WITH THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY BE USED ONLY IN ACCORDANCE WITH THE LICENSE TERMS DESCRIBED BELOW. USE OF DISK OR THE ACCOMPANYING MANUAL SHALL BE DEEMED TO CONSTITUTE YOUR ACCEPTANCE OF THE TERMS OF THIS LICENSE.

Panasonic Industrial Company, Division of Matsushita Electric Corporation of America ("PIC") provides this Program and licenses its use in the United States and Canada under the following terms and conditions:

1. You may use the Program only on the single Panasonic Sr. Partner computer with which the Program was provided;
2. You may copy the Program into any machine readable or printed form for backup or modification purposes in support of your use of the Program on the single Panasonic Sr. Partner Computer;
3. You may transfer the Program and license it to another party if the other party agrees to accept the terms and conditions of this Agreement. At the time of such a transfer you must also transfer all copies, whether in printed or machine readable form, to the same party or destroy any copies not so transferred;
4. You may not remove any copyright, trademark or other notice or product identification from the Program and you must reproduce and include any such notice or product identification on any copy of the Program.

The Program contains unpublished materials, and the existence of any copyright notice shall not mean that publication has occurred or that all or any part of the Program is not secret.

YOU MAY NOT USE, COPY, MODIFY, OR TRANSFER THE PROGRAM, OR ANY COPY OF THE PROGRAM, IN WHOLE OR IN PART, EXCEPT AS EXPRESSLY PROVIDED FOR IN THIS LICENSE.

IF YOU TRANSFER POSSESSION OF ANY COPY, MODIFICATION OR MERGED PORTION OF THE PROGRAM TO ANOTHER PART, YOUR LICENSE IS AUTOMATICALLY TERMINATED.

This license is effective until terminated. You may terminate it at any time by destroying the Program, together with all copies in any form. It will also terminate upon conditions set forth elsewhere in this Agreement or if you fail to comply with any term or condition of this Agreement. Upon such termination you must destory the Program together with all copies, modifications and merged portions in any form.

LIMITED WARRANTY
MEDIUM ON WHICH SOFTWARE PROGRAM FOR SR. PARTNER IS RECORDED

This limited warranty applies only to the medium on which the Software Program is recorded. Except for this limited warranty on the medium, Panasonic Industrial Company makes no warranties, express or implied, with respect to the Software Program, its medium, the user manual or the results, use or performance.

If, as a result of faulty manufacture, a defect occurs in the medium on which the Software Program is recorded, and User returns it postage prepaid to anthorized Panasonic & Service Dealer or Matsushita Engineering & Service Company, Industrial Service Division, One Panasonic Way, Secaucus NJ 07094 within sixty (60) days from licensing by User, accompanied by proof of licensing and an explanation of the suspected defect, Panasonic Industrial Company will, at its option, replace the medium free or charge or return of credit an appropriate portion of the license fee paid by User.

This limited warranty applies only to the initial User and does not apply if the product has been subjected to physical abuse or used in defective or non-compatible equipment.

THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR USE FOR A PARTICULAR PURPOSE, IF ANY, AS TO THE MEDIUM ON WHICH THE SOFTWARE PROGRAM IS RECORDED ARE LIMITED TO SIXTY (60) DAYS FROM THE DATE OF LICENSING BY THE INITIAL USER OF THE PRODUCT AND ARE NOT EXTENDED TO ANY OTHER PARTY. ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR USE FOR A PARTICULAR PURPOSE WITH RESPECT TO THE SOFTWARE PROGRAM AND MANUAL ARE EXPRESSLY DISCLAIMED. User agrees that any liability of Panasonic Industrial Company hereunder, regardless of the form of action, shall not exceed the license fee paid by user to Panasonic Industrial Company.

Panasonic Industrial Company shall not be liable for incidental or consequential damages, such as, but not limited to, loss or injury to business, profits, goodwill, or for exemplary damages, even if Panasonic Industrial Company has been advised of the possibility of such damages.

The remedies stated herein are your sole and exclusive remedies; however, some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitations or exclusions may not apply to you.

To locate an Authorized Servicenter in Your Area within the Continental U.S.A.

DIAL TOLL FREE: 800-447-4700

24 Hours a Day, 7 Days a Week

Requests for assistance in obtaining repairs or technical information...contact any one of the following Panasonic Factory Servicenters

Panasonic Factory Servicenter

Eastern	Midwest	Western	Southern
45 Hartz Way Secaucus, NJ 07094 201-348-7466	425A EAST Algonquin Road Arlington Heights, IL 60005 312-981-4841	6550 Katella Avenue Cypress, CA 90630 714-895-7450	3 Meca Way Norcross, GA 30093 404-925-6855 1825 Walnut Hill Lane Irving, TX 75062 214-256-1387

Correspondence requesting products information should be sent to: Computer Dept., Panasonic Industrial Company, Division of Matsushita Electric Corp of America. 1 Panasonic Way, Secaucus, N.J. 07094

FEDERAL COMMUNICATIONS COMMISSION RADIO FREQUENCY INTERFERENCE STATEMENT

This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC rules. Use only with peripherals certified to comply with Class B limits. Use with non-certified peripherals is likely to cause interference to radio and TV reception.

This equipment generates and uses radio frequency energy. If not installed and used properly, that is, in strict accordance with the manufacturer's instructions, it may cause interference to radio and television. This equipment has been tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J or Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference in a residential installation.

If this equipment does cause interference to radio or television reception, which can be determined by turning the equipment on and off, the user is encouraged to attempt to correct the interference by one or more of the following measures:

Reorient the receiving antenna

Relocate the computer with respect to the receiver

Move the computer away from the receiver

Plug the computer into a different outlet so that the computer and receiver are on different branch circuits

If necessary, the user should consult the dealer or an experienced radio/television technician for additional suggestions.

FEDERAL COMMUNICATIONS COMMISSION RADIO FREQUENCY INTERFERENCE STATEMENT

This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC rules. Use only with peripherals certified to comply with Class B limits. Use with non-certified peripherals is likely to cause interference to radio and TV reception.

This equipment generates and uses radio frequency energy. If not installed and used properly, that is, in strict accordance with the manufacturer's instructions, it may cause interference to radio and television. This equipment has been tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J or Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference in a residential installation.

If this equipment does cause interference to radio or television reception, which can be determined by turning the equipment on and off, the user is encouraged to attempt to correct the interference by one or more of the following measures:

Reorient the receiving antenna

Relocate the computer with respect to the receiver

Move the computer away from the receiver

Plug the computer into a different outlet so that the computer and receiver are on different branch circuits

If necessary, the user should consult the dealer or an experienced radio/television technician for additional suggestions.

The user may find the following booklet, prepared by the FCC, to be helpful:

"How to Identify and Resolve Radio-TV Interference Problems".

This booklet, Stock No. 004-000-00345-4, is available from the US Government Printing Office, Washington, D.C. 20402.

*** NOTE ***

Recommended shielded cables must be used to connect between RS-232C devices parallel printer or monitor in order that FCC Class B emission limits are still being satisfied.

Operations/DOS

Reference Guide

Sr. Partner™
Portable Computer

Panasonic

USA
Panasonic Industrial Company
Division of Matsushita Electric Corporation of America
One Panasonic Way,
Secaucus, New Jersey 07094

Panasonic Hawaii Inc.
91-238 Kauhi St. Ewa Beach
P.O. Box 774
Honolulu, Hawaii 96808-0774

Panasonic Sales Company
Division of Matsushita Electric of Puerto Rico, Inc.
Ave. 65 De Infanteria, KM 9.7
Victoria Industrial Park
Carolina, Puerto Rico 00630

CANADA
Matsushita Electric of Canada Limited
5770 Ambler Drive, Mississauga,
Ontario L4W 2T3

OTHERS
Matsushita Electric Trading Co., Ltd.
32nd floor, World Trade Center Bldg.,
No. 4-1, Hamamatsu-Cho 2-Chome,
Minato-Ku, Tokyo 105, Japan
Tokyo Branch P.O. Box 18 Trade Center

CONTENTS

CHAPTER 3

SETTING UP THE SYSTEM 3-1

SET UP

CHAPTER 4

OPERATING THE SYSTEM 4-1

OPERATION

OPERATION

OPERATION | USING DOS

CHAPTER 5

CHAPTER 6

COMMAND REFERENCE

CHAPTER 7

DOS COMMANDS

BATCH

CHAPTER 8

EDLIN

CHAPTER 9

LINK

LINK

CHAPTER 10

CONTROL

IN/OUT

APP. A APP. B APP. C APP. D APP. E APP. F APP. G APP. H

CHAPTER 1

INTRODUCTION

FEATURES OF THE SR. PARTNER SYSTEM

INTRO

Welcome to the world of portable computing!

Portability means you can take your computer wherever the work needs to be done, at your office, at your home and anywhere along the way. But don't let the size of your new Sr. Partner™ fool you–there's a lot of computing power packed into that convenient, easy to transport package.

Your Sr. Partner comes complete with everything you need to meet your exacting business and personal computer needs:

- Powerful 16-bit CPU
- 256K RAM (expandable up to 512K)
- 83-key, tilt-adjustable keyboard
- 9" high resolution video display unit
- Internal thermal printer
- 5¼ disk drive (double-sided, double-density)
- Runs IBM-compatible software and accepts IBM-compatible hardware
- Bundling popular application software

Best of all, your Sr. Partner is a Panasonic, a name known for quality, dependability, and service.

AVAILABLE OPTIONS FOR THE SR. PARTNER

Your Sr. Partner is a powerful computer system just as you purchase it from the dealer. These options can be added to the Sr. Partner system to make it even more responsive to your individual needs:

OPTION	MODEL NUMBER
Floppy Disk Drive	RL-M700
Slot Extender	RD-9652
Video Adaptor	RD-9651
Thermal Paper	RD-9671

HOW TO USE THIS MANUAL

This OPERATIONS/DOS REFERENCE GUIDE is divided into two sections. The first section, OPERATIONS GUIDE (Chapter 1–Chapter 4), is designed to help you set up your Sr. Partner. It offers instruction on operating your computer and introduces the DOS-Disk Operating System.

The second section of the manual is a DOS REFERENCE GUIDE (Chapter 5–Chapter 12). This guide will be a powerful reference tool once you have become familiar with DOS.

Chapter 1–Introduction. Outlines the features of your system, lists options and explains manual organization.

Chapter 2–The Parts of a Computer System. Details the internal and external components of a computer system.

Chapter 3–Setting Up the System. Describes the unpacking and set up of the Sr. Partner including detailed explanations of the system unit, keyboard, and printer, and installation options.

Chapter 4–Operating the System. Gives operating instructions including use of the keyboard, disk drives and a tutorial introducing DOS.

Chapter 5–Using DOS. Explains keyboard usage under DOS, filenames, specifications and directories.

Chapter 6–Command Reference. Presents a complete explanation of each DOS command, including syntax, purpose, comments and examples.

Chapter 7–Batch Processing. Describes the commands and procedures for creating and using batch files.

Chapter 8–EDLIN. Details the operation and commands of the line editor program.

Chapter 9–LINK. Details the operation and commands of the LINK program.

Chapter 10–DEBUG. Details the operation and commands of the DEBUG program.

Chapter 11–Control of Screen and Keyboard. Provides information on setting up special characters to control the screen and keyboard.

Chapter 12–Input and Output Options. Gives instructions for modifying standard input and output procedures.

Several useful Appendices appear at the back of this manual:

APPENDIX A–User Diagnostics. Helps you troubleshoot your system should problems arise when turning on or operating your Sr. Partner System.

APPENDIX B–Error Messages. Lists the error messages you might encounter during DOS operation.

APPENDIX C–Pin Configurations. Contains the wiring specifications for the serial port, the parallel port and RGB monitor port.

APPENDIX D–DOS Interrupts and Function Calls. Provides technical information for DOS operations.

APPENDIX E–Control Blocks. Details advanced DOS usage.

APPENDIX F–Character Set. Gives the character set for quick reference.

APPENDIX G–Specifications. Gives complete specifications of your Sr. Partner System.

APPENDIX H–Index. Presents a complete index to the Operations /DOS Reference Guide.

CHAPTER 2

THE PARTS OF A COMPUTER SYSTEM

PARTS

PARTS

THE PARTS OF THE COMPUTER YOU CAN SEE

From the outside your Sr. Partner looks rather like a suitcase. But inside that "suitcase" is the equivalent of the "room size" computers of just a few decades ago! Continuing technological breakthroughs have reduced the size and cost of computers, while increasing their computing power. And so today you can carry around a Sr. Partner that is at your beck and call, that can provide years of experience in a few moments, that is an efficient and well-trained secretary, that can help you meet you business and personal computer needs—RIGHT NOW!

When you open that "suitcase", what do you see? Well at first glance it appears you are carrying part of a typewriter and a rather small television.

The "typewriter" is your **keyboard**. This is your means of communication with your Sr. Partner. You use this keyboard to enter information into the computer, both using the letters of the alphabet and the numeric keys (part of the keyboard even resembles the keypad on a calculator). Using this keyboard you instruct your computer on what to do, to which information, and how to present the results.

The "TV" is your **video display terminal**. This is the computers method of communicating with you. All of the information you type in on the keyboard is displayed on this screen, just so you and the computer know that you're both talking about the same thing! The computer also asks for instruction through the display, by awaiting your response to "prompts". Finally, the display shows you the results of certain procedures. These results can be text, or they can be graphics, (sometimes they can even include sound).

Next to the video display terminal (VDT) is one or two **disk drive units**. Think of these drives as vertical phonographs. The records these phonographs "play" are called **disks**. Now actual phonographs use a needle which tracks along the surface of a recorded disk and reproduce the signals laid down on the record. Disk drives have a stationary "head" which decodes information stored in the magnetic coating on the disks. This head does not move along the disk. Instead, it reads data as the disk spins below its surface.

The disk drive head can also act like a tape recorder. That is, it can also record information on the disk. When the head is transferring information already stored on the disk, it is "reading". When it is laying down information on the disk, it is "writing".

Both the VDT and the disk drives are enclosed in the **system unit**. The system unit also contains some connections and controls for controlling parts of your Sr. Partner system as well as an internal **printer**.

But it's what is inside the system unit that makes your computer the powerful tool that it is!

THE PARTS OF THE COMPUTER YOU CAN'T SEE

Much of the inside of the computer is taken up with the electrical and mechanical connections that enable the computer to operate. These include the bulk of the disk drive units, cooling fans and display mechanics.

Also inside the system unit is the **central processing unit** (CPU). This microprocessor is the real "brains" of your computer system. The central processor monitors and controls the electronic passageways that send signals to the appropriate locations within the machine or to the correct "peripherals" (keyboard, display, printer, disk or other external device).

Computers store information in binary format. Binary means two, and in this case it means information is stored in one of two ways "off" and "on". Each piece of binary information is called a "bit". Eight bits make one "byte". In the English language, each byte represents one characters. Thus each character is read by the computer as a series of bits, in either the off or on position. The Character Set in Appendix F gives the binary (hexadecimal) representation of all the characters on your keyboard.

Information is stored in **memory**. Your Sr. Partner has some data permanently recorded inside the machine. This part of memory is called **ROM (Read Only Memory)**. As the name implies this memory cannot be altered in any way. It is wired into the computer. ROM is permanent, it cannot be affected by interactions with the computer.

The other type of memory in your computer is called **RAM (Random Access Memory)**. RAM is volatile memory, it disappears when you turn the computer off. You use RAM while you are working with data. RAM reads stored data (usually from your disks), uses this data to perform procedures and functions, and then writes the data back to the storage device. RAM is large enough to hold DOS (read in from your SYSTEM DISK), application programs (read in from the program disks) and data files which you use to create and store your personal information. Your Sr. Partner comes with 256K bytes of RAM. You can easily add expansion RAM to your system.

INFORMATION FLOW

Imagine now how the computer handles the information it processes. First, it must have electrical power, because information is transferred electrically.

After the computer is turned on, it performs some internal tests which are contained in ROM. These checks make sure the parts of the computer are "up" (on) and in good working order.

Now the computer looks for something to do. The computer receives its instructions from the **operating system**. Your computer uses the disk operating system. Some of DOS is built into the ROM of your machine. Some of DOS is read in from your System Disk. DOS is activated when you "boot" the machine.

Booting is simply bringing up the system. It turns on the DOS in ROM and reads in additional DOS from the disk.

You type in a DOS command and the machine starts performing. It may start up an applications programs or go into BASIC so that you can do some programming. Or you may create, delete, update or move your data files using DOS programs.

All instructions go through the CPU. Get this information from a disk. Perform this operation. Write this result. Run this program. Display these results.

YOU ARE THE KEY

Your Sr. Partner is a very "smart" machine. That is, it has a lot of computing power. But it has one very big limitation.

ALL IT CAN DO IS FOLLOW INSTRUCTIONS!!

The key to successful computing is **you.** The Sr. Partner can make it easy for you to issue the right instructions. Using DOS is quite simple. Everyday more and more applications programs are being introduced, allowing you to perform many complicated procedures. Using BASIC, you can learn to write your own programs very quickly.

So take some time now to acquaint yourself with your new system. Once you become familiar with the exciting capacity and capabilities of your Sr. Partner you will be in control of one of the most useful tools you have ever earned.

CHAPTER 3

SETTING UP THE SYSTEM

SET UP

UNPACKING

Inside the Carton

Inside the carton you will find:

1. the **system unit** including the computer display, disk drive and printer
2. the **keyboard unit**
3. the **AC power cord** (stored in the rear of your system unit)
4. the **Operations/DOS Reference Guide** (including the System Disk in a jacket inside the rear cover) and **BASIC Reference Guide**
5. a **roll of paper** for the internal printer

6 a **paper rod**

7. a **plastic cover**

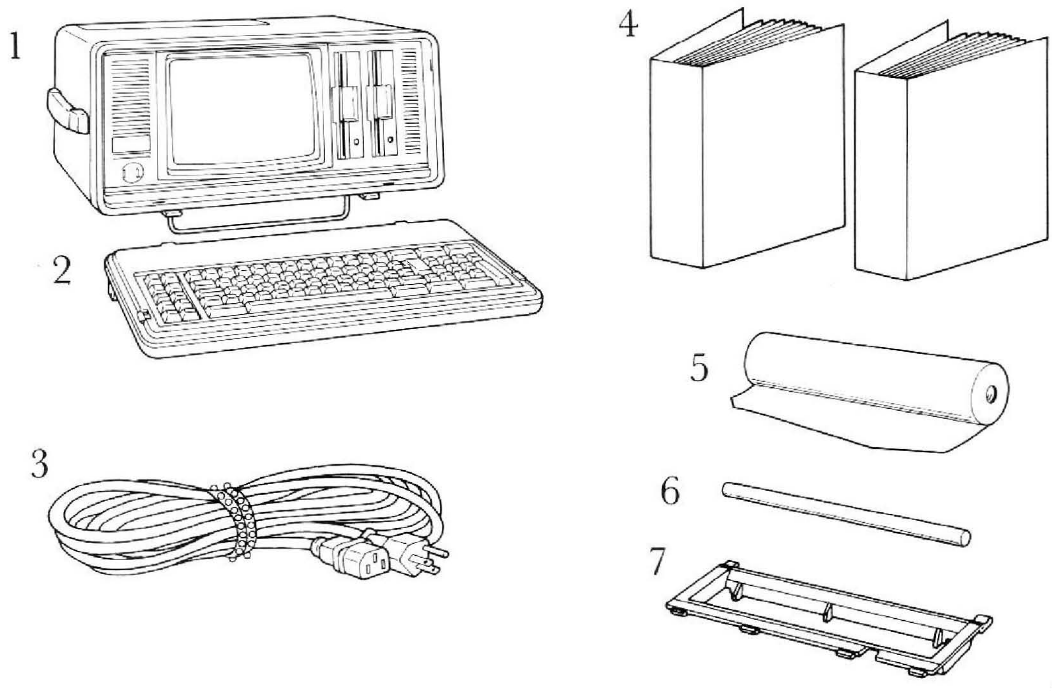


Fig. 3-1. COMPONENTS IN THE CARTON

Setting Up the System Unit

> **NOTE:** Be sure you set up your Sr. Partner within easy reach of an electrical outlet.

Set the machine down on its "feet". The handle is on top of the unit.

SET UP

The metal rod folded over the front of your machine serves to protect the keyboard from being dislodged while your Sr. Partner is in storage or transport.

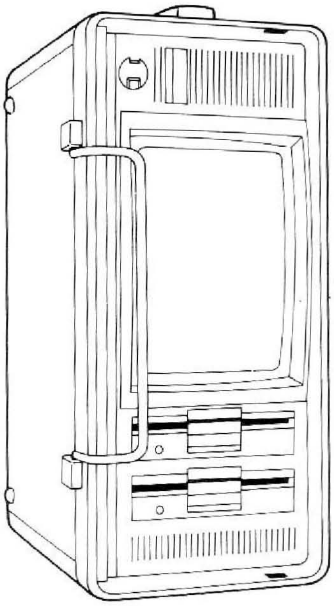


Fig. 3-2. MACHINE BEING SET DOWN IN CORRECT ORIENTATION

This metal rod also serves as a stand for the system unit while you are using your computer. Fold the metal rod down, until the metal rod touches the case. Now place the computer on a firm surface so that it is resting on this stand.

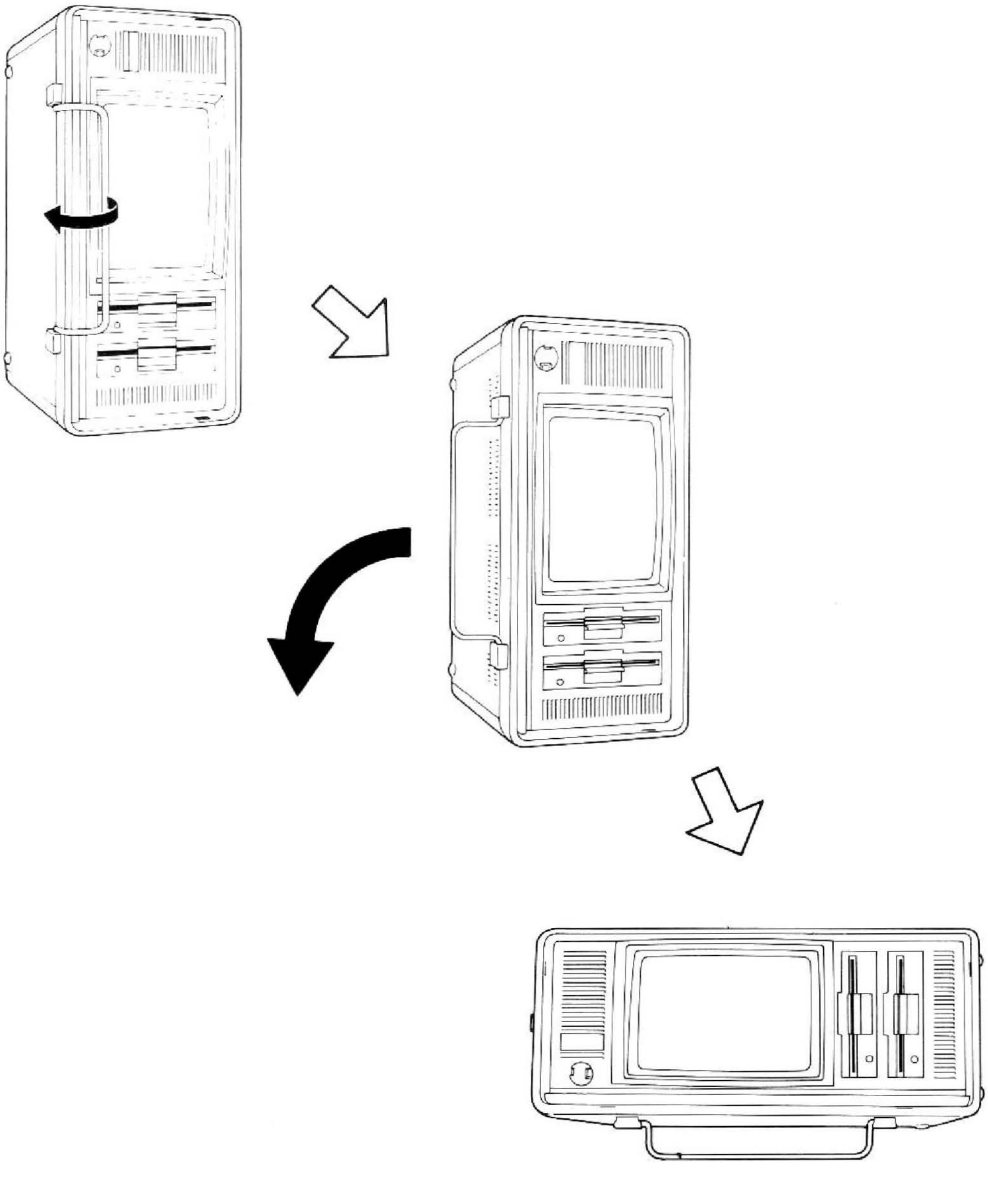


Fig. 3-3. HOW TO REST THE SYSTEM UNIT ON STAND

Setting Up the Keyboard

Turn over your keyboard unit.

At the upper corners of the unit are the legs which allow you to raise the keyboard for easier typing. Pull the two ribbed rectangular sections toward you. This releases the legs. Pull the legs out completely and be sure they are secure.

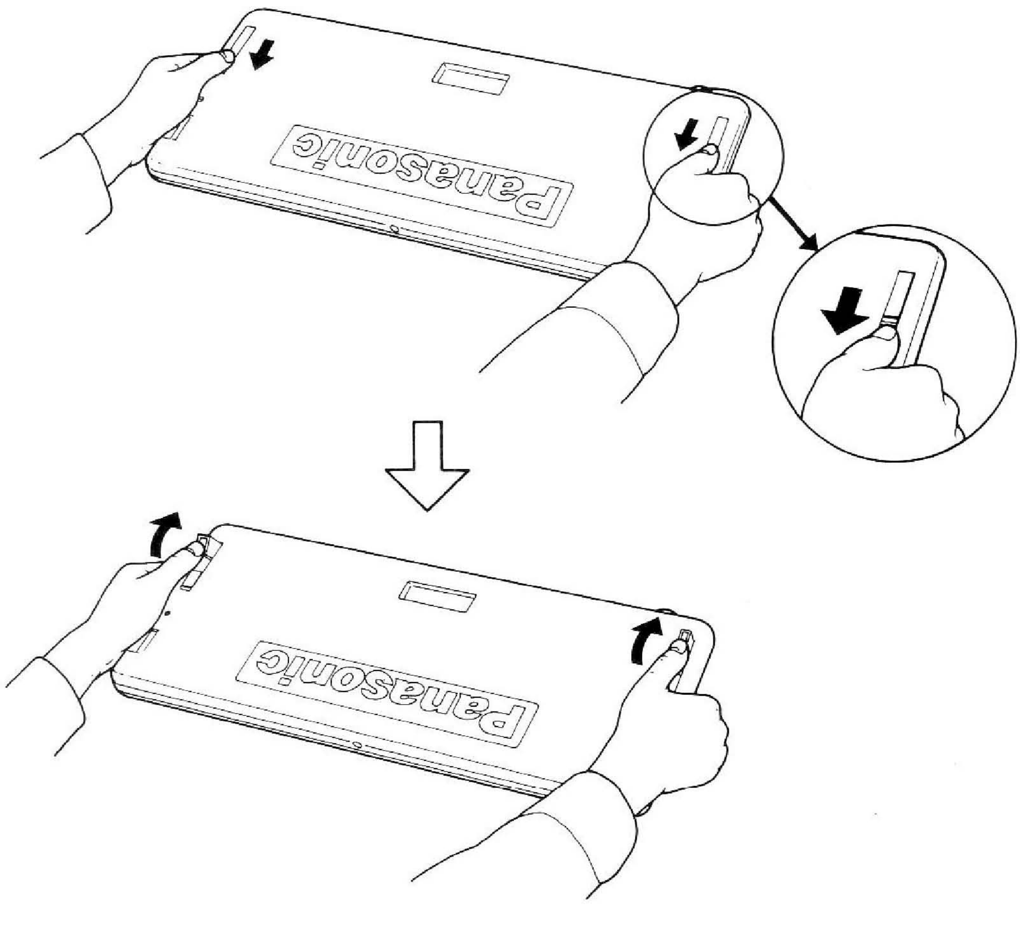


Fig. 3-4. PULLING OUT THE LEGS ON KEYBOARD

Turn over the keyboard and rest it on the extended legs. This slant provides added comfort when you are inputting large amounts of data using the keyboard.

THE SYSTEM UNIT

The system unit is the heart and mind of your computer system. It contains the computers memory, the disk drive units and the mechanical components which make the computer operate.

SET UP

Front Panel

The system unit contains a **video display terminal** (also referred to as a VDT). The VDT displays information that you enter or receive from the computer. The display screen presents up to 25 lines of information, with 80 columns in each line. The display appears as light green characters on a darker green background.

Immediately to the right of the screen is the door of your **disk drive**. Sr. Partner, RL-H7000, comes with one disk drive, as an option you may have another disk drive installed in the system unit. Sr. Partner, RL-H7000W, comes with two disk drives. Notice that the drive nearest to the screen (to the left) is labelled drive A. To the right of drive A is drive B.

In the bottom left hand corner you will see a connection on the end of a short cord. This cable is used to connect your system to the keyboard. This connection is described in the keyboard section.
Surrounding the outside of the front panel are ventilation openings for cooling the computer during operation.

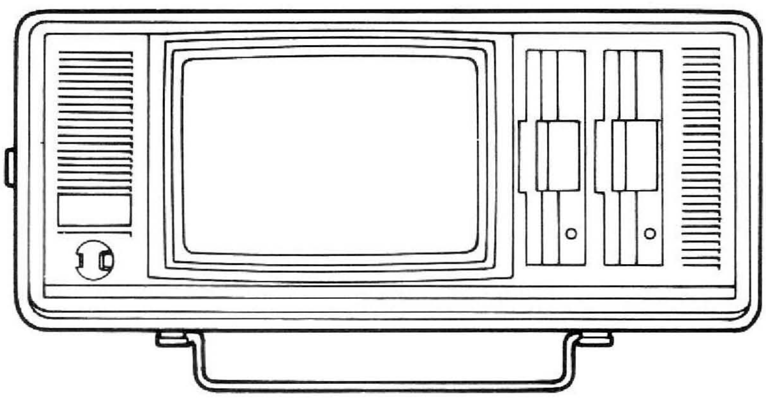


Fig. 3-5. FRONT PANEL (RL-H7000W)

Rear Panel

Turn your Sr. Partner around so that the rear panel is facing you.

There are two ribbed tabs in the middle of the panel. Push down on these tabs and slowly lower the cover protecting the rear panel.

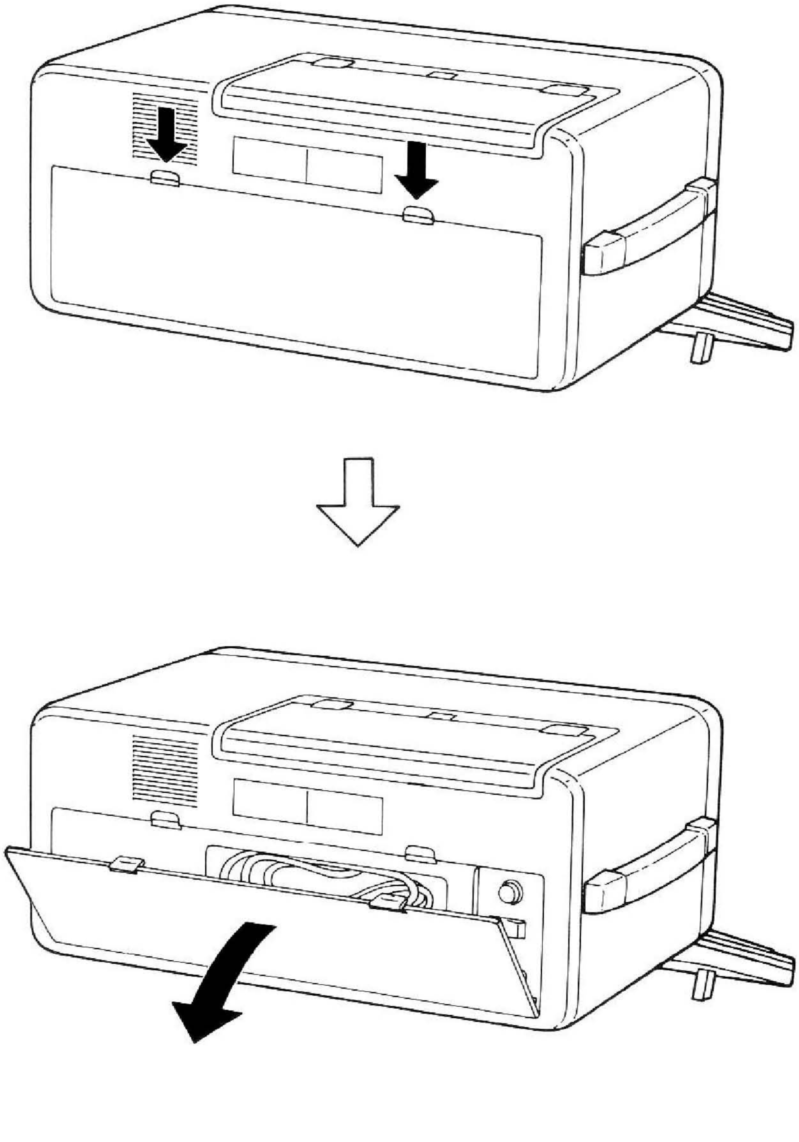


Fig. 3-6. REMOVING COVER FROM REAR PANEL

Your **AC power (connection cord)** is stored in the rear panel. Remove the cord and put it aside while you familiarize yourself with the connections and controls of your Sr. Partner. DO NOT CONNECT THE CORD AT THIS TIME.

In the upper left hand is the **cooling fan.**

Surroundings the power connection cord storage compartment are several controls and connections for powering your Sr. Partner and connecting other peripherals to your Sr. Partner system.

The **fuse** compartment contains one fuse. It is installed at the factory. To change the fuse see the instructions in Appendix A, "User Diagnostics".

Directly below the fuse is the main **power switch**. When the Sr. Partner is connected to a AC power outlet, this rocker switch is the main power control. When the right side of switch is pressed in, the Sr. Partner is turned ON. When the left side is pressed in, the power is OFF.

The **AC power cord connection** is in the lower right hand corner. DO NOT CONNECT THE CORD AT THIS TIME.

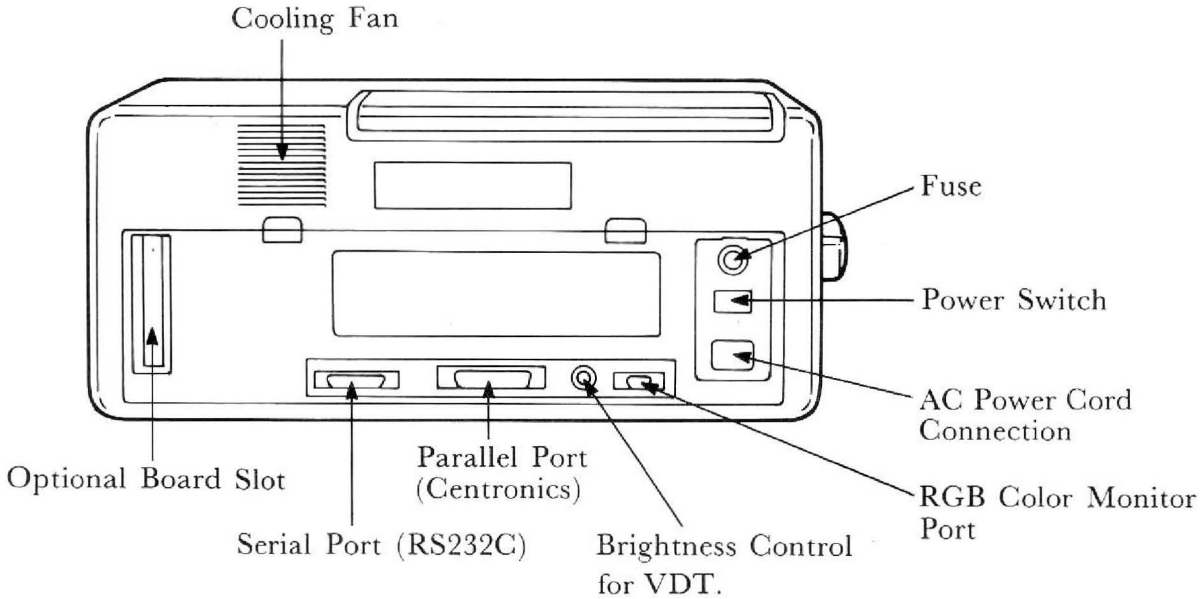


Fig. 3-7. REAR PANEL

Along the bottom edge of the rear panel are several controls and connections. Moving to the left you will see:

RGB Color Monitor Port

Brightness Control

Parallel Port (Centronics)

Serial Port (RS232C)

Optional Board Slot (in which you insert additional boards you may purchase to upgrade your Sr. Partner system).

Connecting the AC Power Cord

*** **WARNING** ***

BE SURE THE MAIN POWER SWITCH IS IN THE OFF POSITION BEFORE ATTACHING THE AC POWER CORD

One end of the AC power cord has 3 holes arranged in a triangle. Hold this end so that the single prong is on the top and the two prongs are on the bottom. Insert this end into the AC power cord connection in the lower right hand corner of the rear panel.

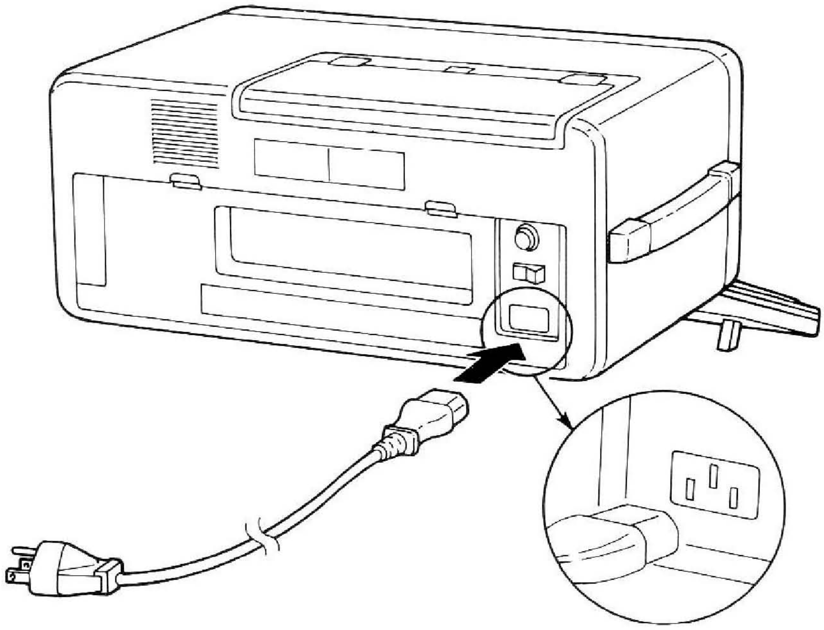


Fig. 3-8. INSERTION OF PLUG INTO THE SYSTEM UNIT

Now plug the other end of the AC power cord into a wall outlet.

NOTE: Make sure the cord reaches the outlet easily. You should not stretch or put pressure on the cord. An inconsistent power supply can damage your machine and cause you to lose valuable data.

SET UP

*** **WARNING** ***

Your Sr. Partner must be plugged into a three pronged outlet. Do not attempt to plug the cord into a two prong extension cord or into an adapter. The third prong provides an essential safety mechanism as it makes sure the machine is properly grounded. Failure to heed this warning may cause malfunctioning of your Sr. Partner.

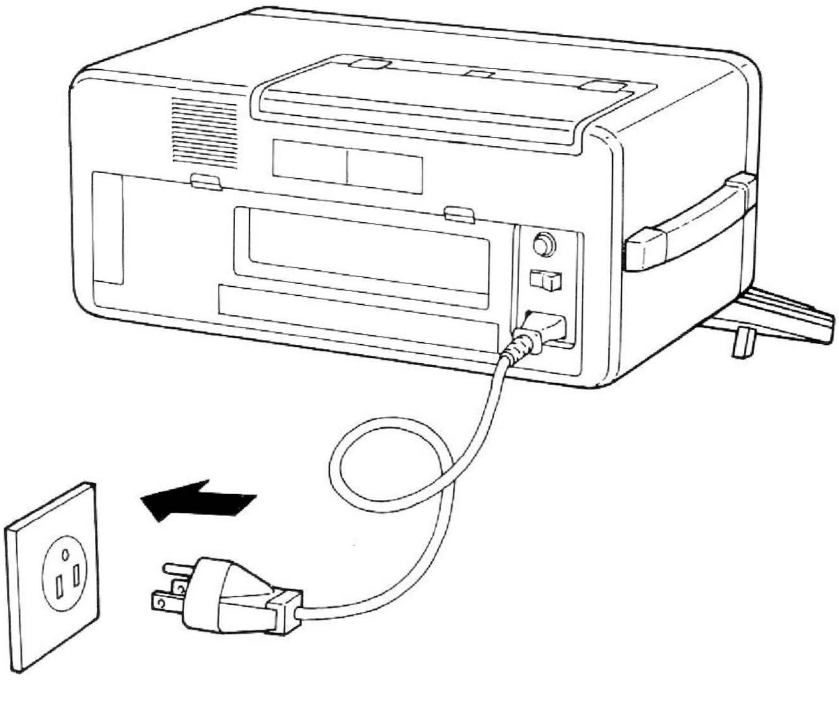


Fig. 3-9. PLUG INSERTED IN WALL OUTLET

Connecting the Keyboard and System Unit

*** **WARNING** ***

BE SURE THE MAIN POWER SWITCH IS IN THE OFF POSITION BEFORE CONNECTING THE KEYBOARD

Turn the machine around so that you are again facing the front panel.

The connector is located on the end of a coiled cable which is located at the left bottom of the front panel on the system unit. Gently pull out this cable.

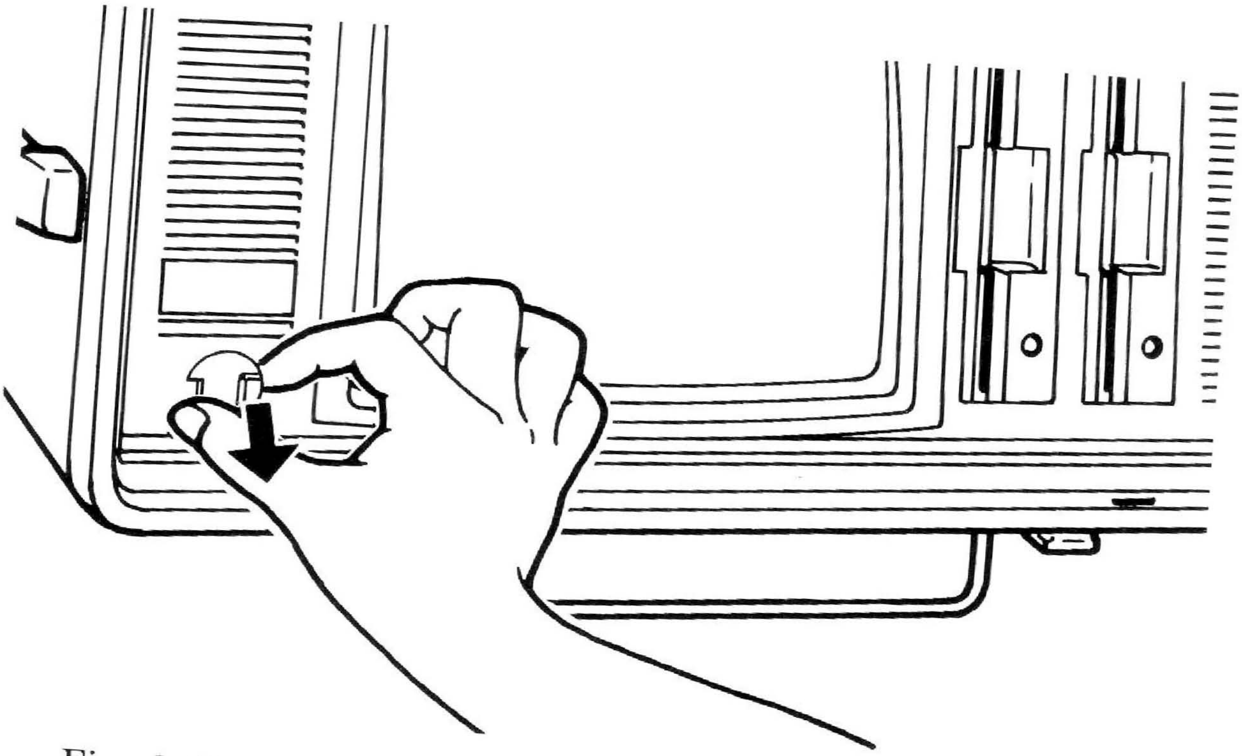


Fig. 3-10. PULLING OUT COILED CABLE FOR KEYBOARD

SET UP

The connector is protected by a plastic cover. Not only does this assure that the connector will not be bent or damaged, it also prevents the connector from slipping inside the front panel when it is disconnected from the keyboard.

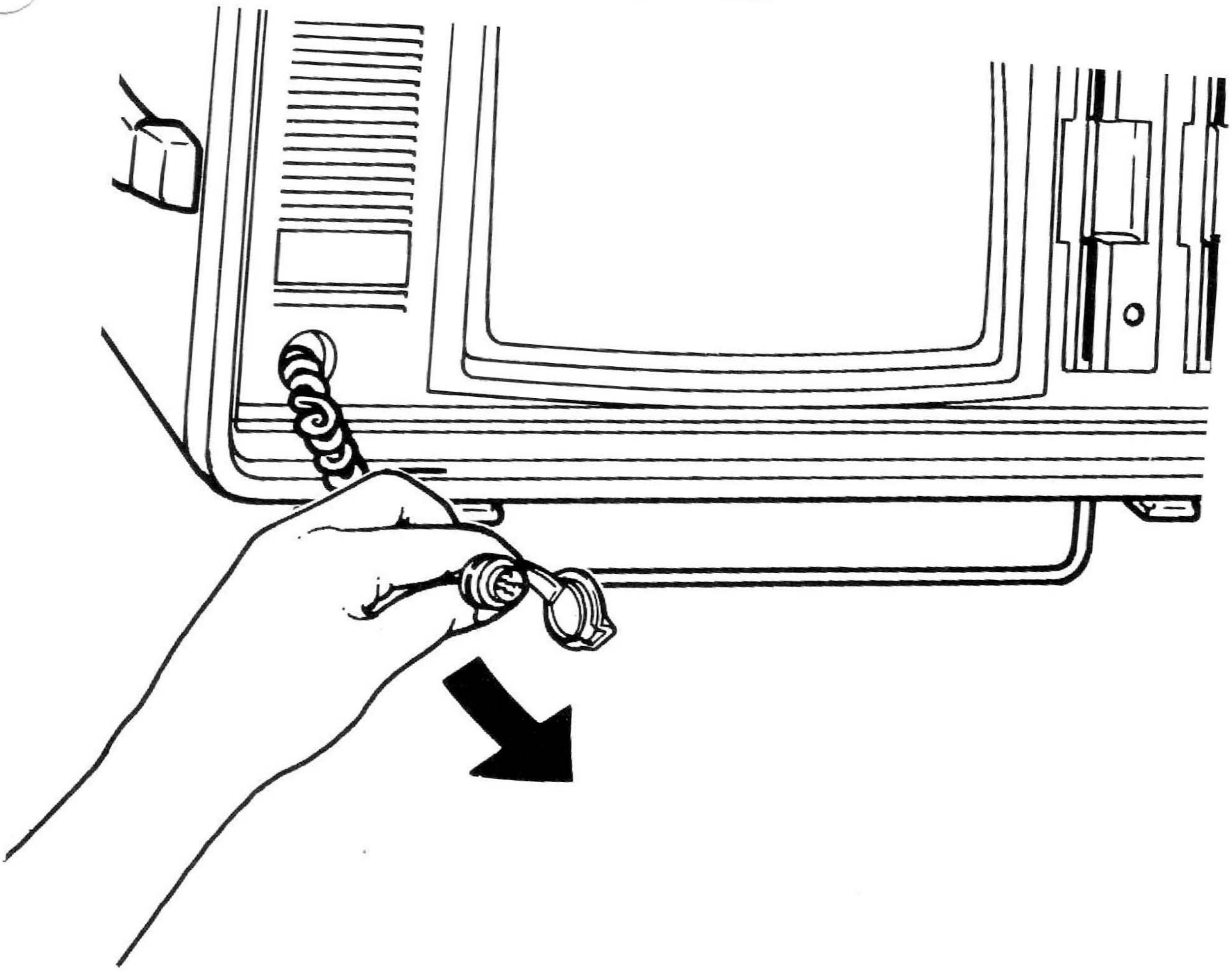


Fig. 3-11. CONNECTOR BEING PULLED OUT

The keyboard outlet for the connection is on the rear left upper corner of the back edge of the keyboard. It is protected by a plastic cover. Unsnap the cover.

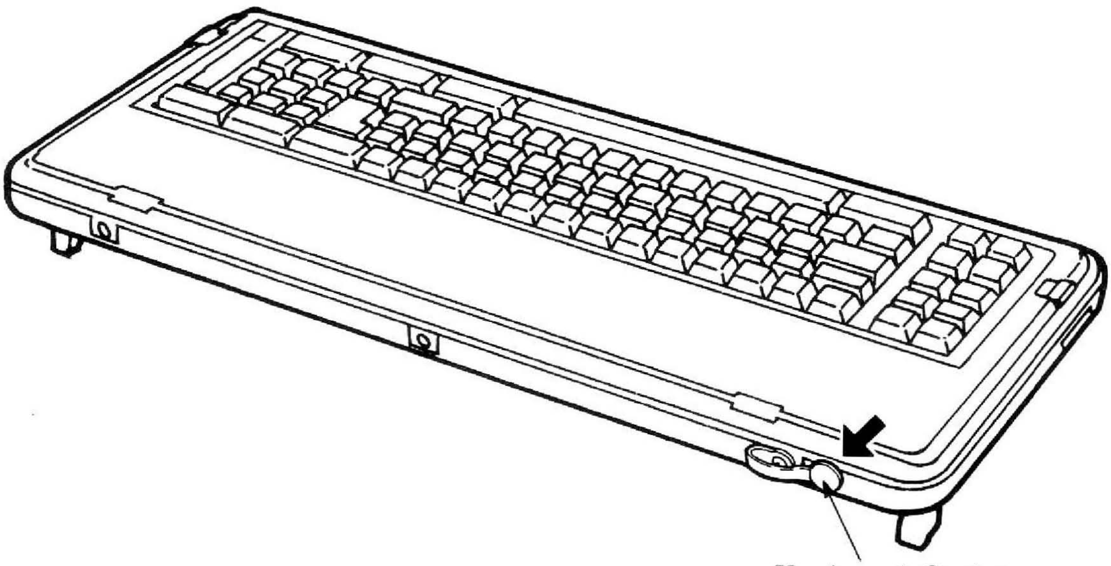


Fig. 3-12. THE LOCATION OF THE KEYBOARD OUTLET

The silver prongs fit into the keyboard outlet. Hold the connector so that the arrow on the black section of the connector is on the top. Firmly insert the cable into the outlet and turn the wheel surrounding the connector to fix it.

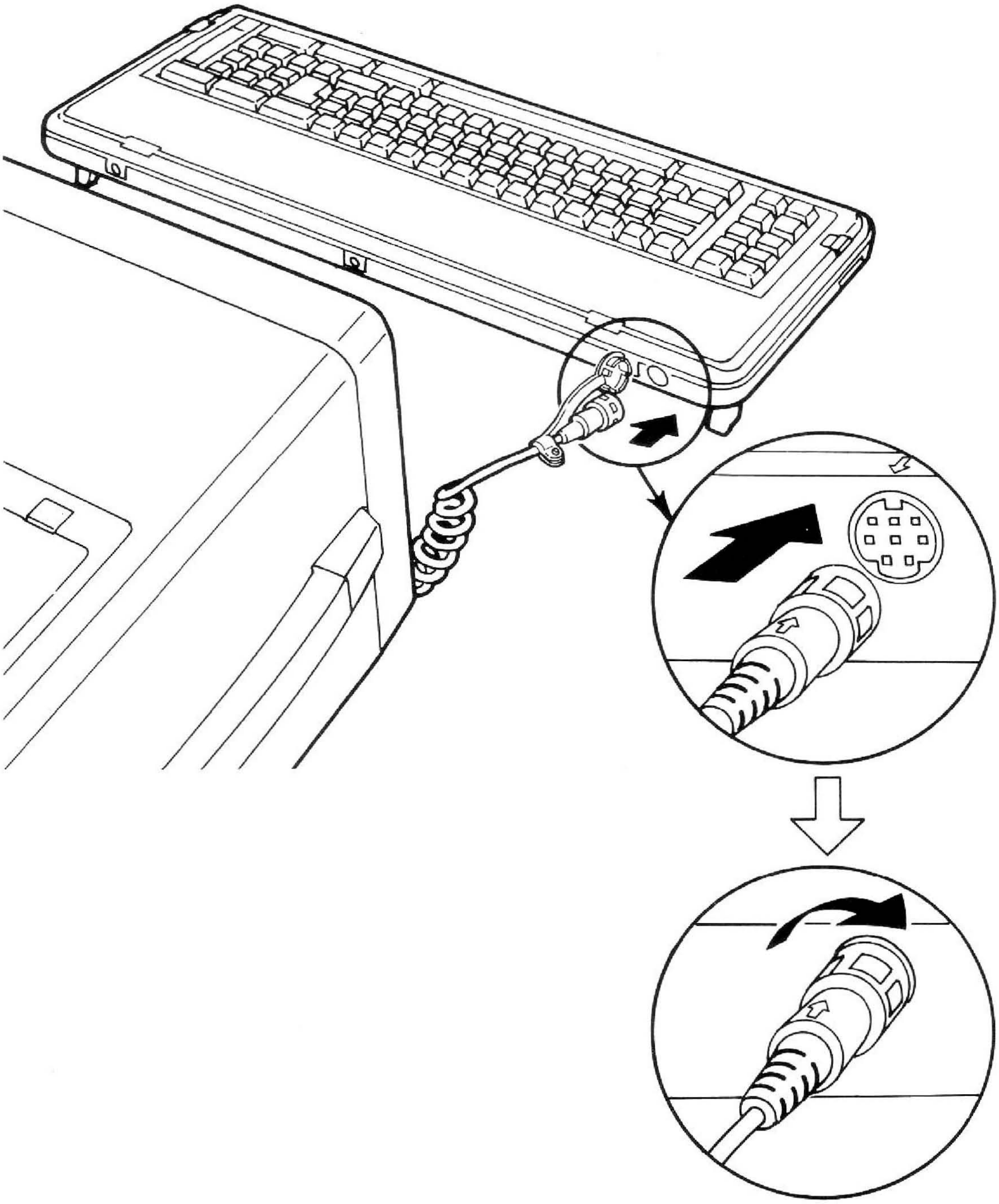


Fig. 3-13. THE INSERTION BETWEEN CONNECTOR AND OUTLET

SET UP

THE KEYBOARD UNIT

Before you begin to enter data into your Sr. Partner take a moment to acquaint yourself with the main features of your keyboard. Details of key usage will be covered in Chapter 4.

SET UP

Types of Keys

The **alphanumeric keys** on the keyboard resemble the layout of the keys on a standard typewriter. The keys can produce upper and lower case characters, numbers and the symbols above the numeric keys.

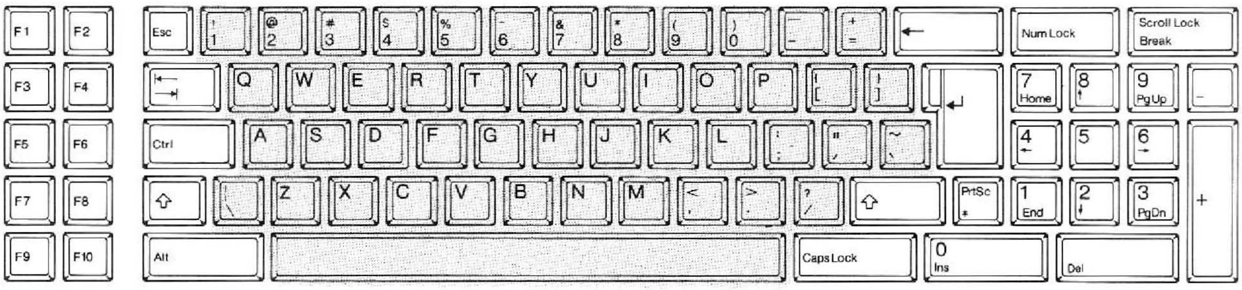


Fig. 3-14. ALPHANUMERIC KEYS

The **control keys** act to modify the alphanumeric keys. You may be familiar with the <BACKSPACE> key (indicated by a left arrow), the <SHIFT> keys (indicated by vertical arrows), the <TAB> key (indicated by right and left arrows), and the <Caps Lock> key from typing experience. Other control keys specify computer operations and are usually used in conjunction with another key. The <Esc>, <Ctrl>, <Alt>, <Num Lock>, <Scroll Lock> (<Break>) and <PrtSc> are examples of this type of control key.

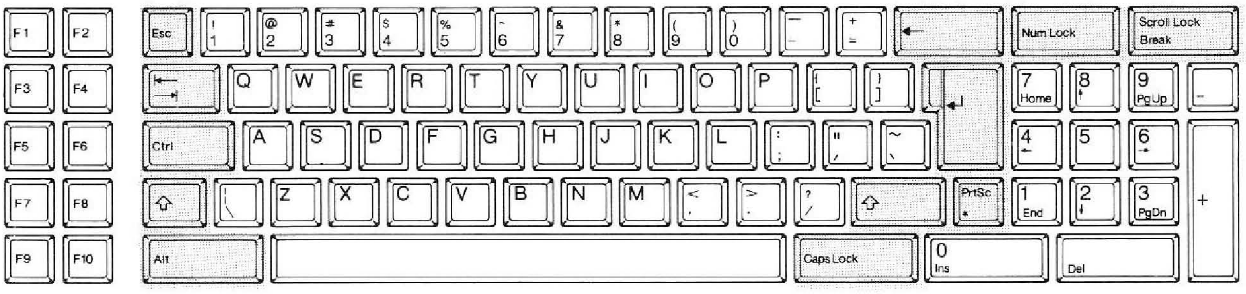


Fig. 3-15. CONTROL KEYS

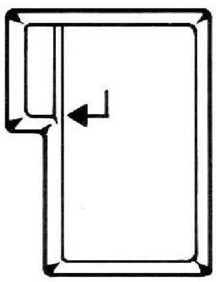


Fig. 3-16.
ENTER KEY

This is the <ENTER> key. IT IS THE MOST IMPORTANT KEY ON YOUR KEYBOARD. When giving your computer commands or instructions, you must press the <ENTER> key to indicate that you are finished typing in data and are ready to ENTER it into the computer's memory. The computer cannot process commands or perform any operations until you have pressed the <ENTER> key.

NOTE: The <ENTER> key is sometimes called the return key and it does move you to the next line on your screen. However, do not confuse this key with a carriage return on a typewriter.

When you type more information into the computer than can fit on one line, your computer will automatically "wrap around" (continue on the next line).

REMEMBER, pressing <ENTER> means you are ready for the computer to perform some action.

These **cursor/number pad keys** can be used in two ways.

When the <Num Lock> control key is pressed, these keys input numbers just like the keypad on a calculator. Because of their layout, these keys are useful if you will be inputting a great deal of numeric data. (You may also use the number keys in the typewriter section to input numeric information.)
When <Num Lock> is pressed again, the keys are in the cursor mode. The cursor is a flashing prompt on your screen which defines your current position. The arrows and symbols on these keys indicate the direction of movement of the cursor.

> **NOTE:** The <Num Lock> key is a **toggle key.**
> This means that it operates in one of two modes. Toggle key modes are lock in, UNTIL THE KEY IS PRESSED AGAIN. In this case the cursor/number pad keys are usually in the cursor mode. By pressing the <Num Lock> key you lock them in the number pad mode. Pressing <Num Lock> again, returns the keys to cursor mode.
>
> Several keys on your Sr. Partner keyboard operate as toggle keys.

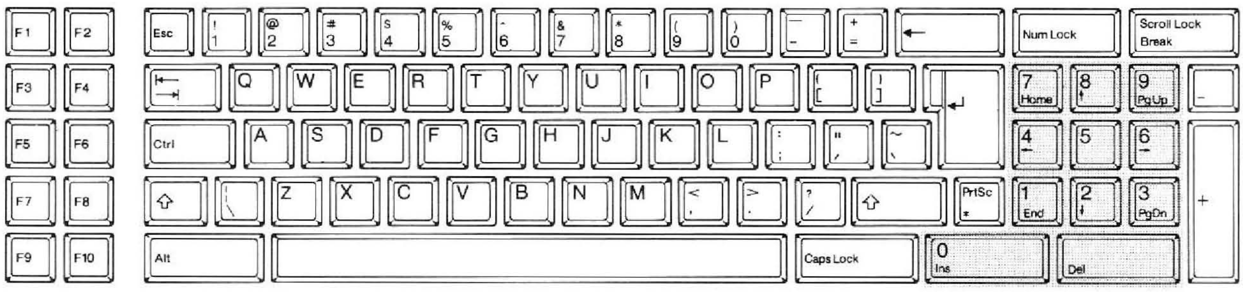


Fig. 3-17. CURSOR/NUMBER PAD KEYS

The **function keys** on the left side of the keyboard perform specific actions or enter specific information DEPENDING ON THE PROGRAMS you are using. Function keys are useful because they allow you to enter information using only one keystroke.

For example, pressing <F1> might move some text in a word-processing program.

In a spreadsheet application program <F1> might begin a calculation procedure.

When programming in BASIC, <F1> allows you to enter the LIST command without having to type out the entire command.

DOS also assigns operations to the function keys.

The manuals for your specific programs detail the operations of the function keys.

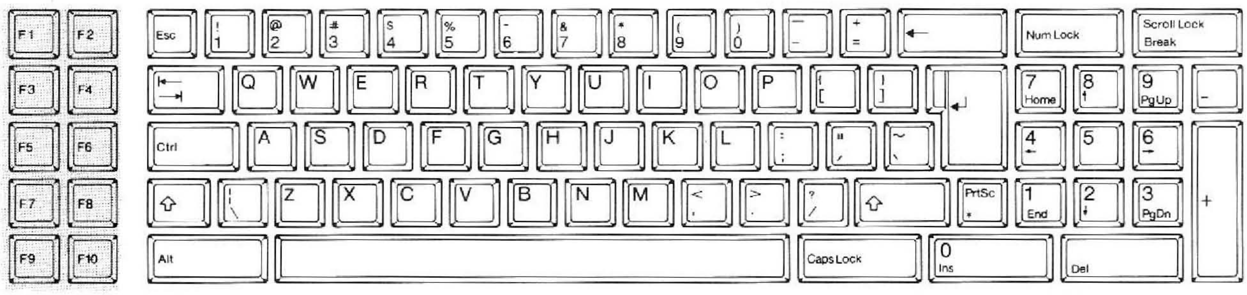


Fig. 3-18. FUNCTION KEYS

THE INTERNAL PRINTER

The last basic element in your Sr. Partner system is the internal printer. The printer is located under the top panel of your system unit.

SET UP

Setting up the printer

The two ribbed tabs release the protective cover for the printer compartment. Push these tabs forward, then slide back the first section of the panel (you will notice that the panel is hinged).

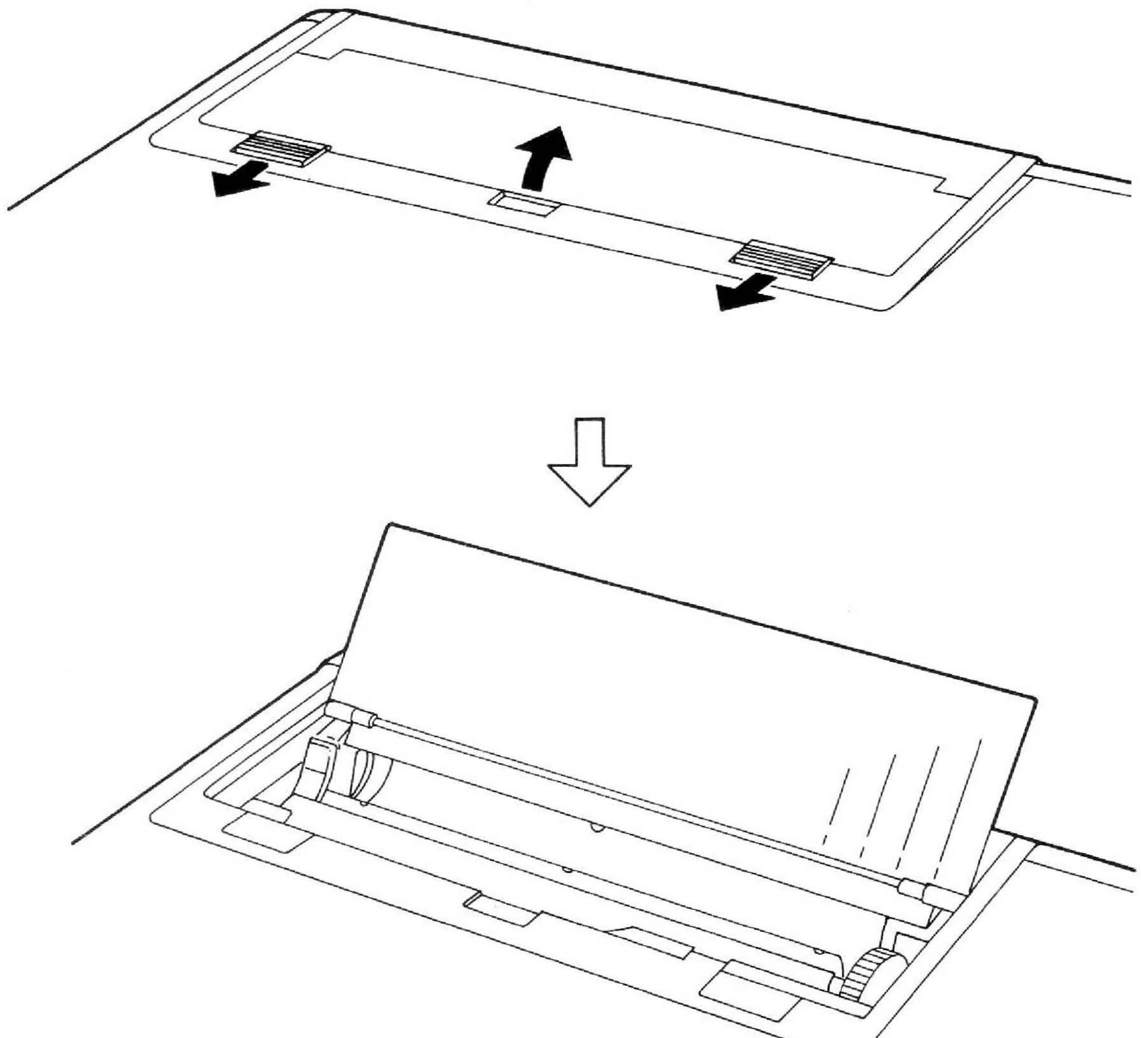


Fig. 3-19. REMOVING THE PROTECTIVE COVER FROM TOP PANEL

Fold back the remaining section of the top panel. This allows you access to the printer so that you may easily insert the paper.

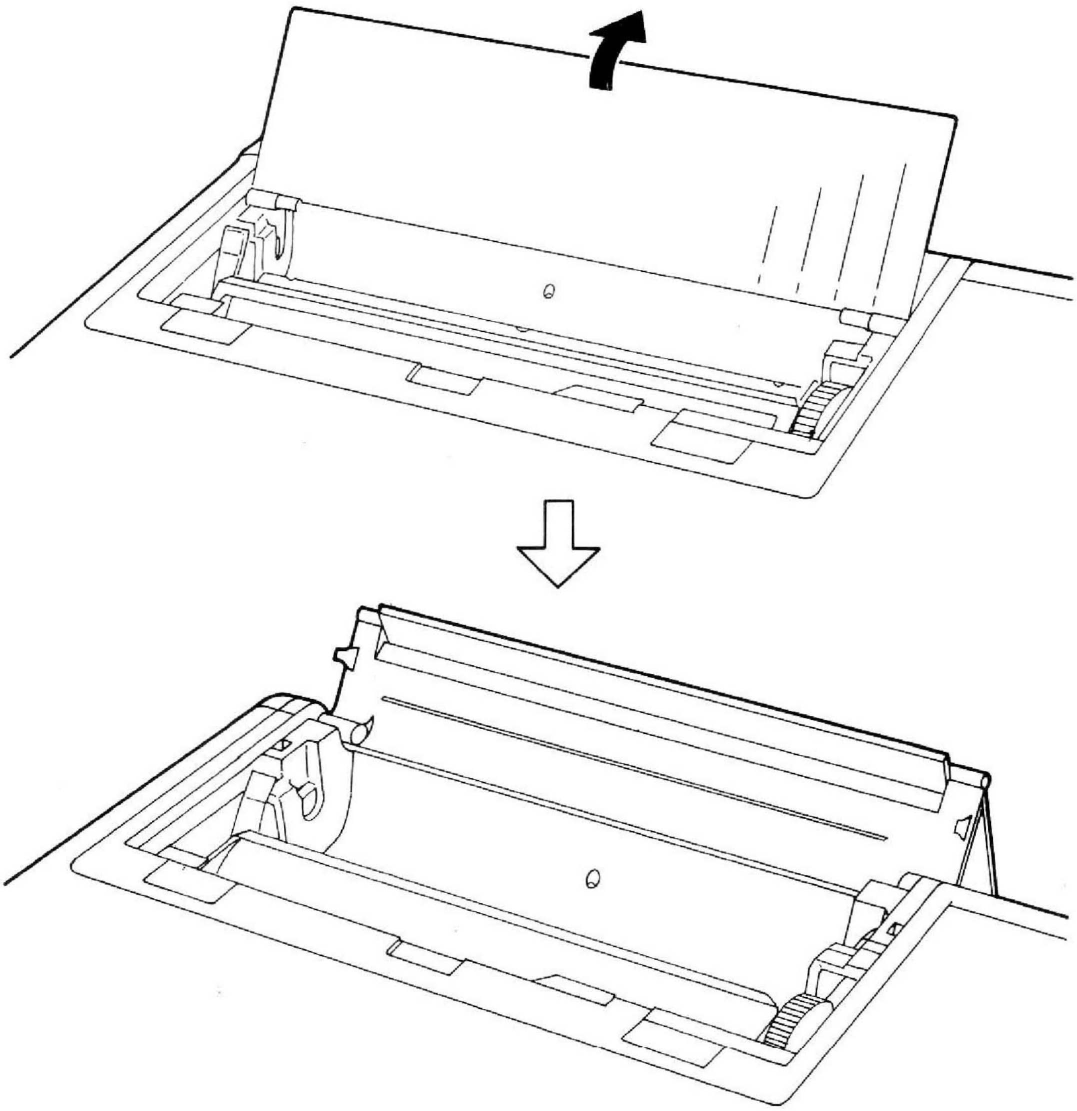


Fig. 3-20. PROTECTIVE COVER FOLDED BACK

The **paper feed knob** rolls the paper forward and is useful for loading the paper and moving it quickly through the machine.

The **line feed button** feeds the paper one line at a time. This button operates only when the computer is switched ON using the main power switch. If you hold down the button, the paper is continuously fed.

The **error indicator** tells you when an error has occurred which prevents the printing from continuing.

The **paper release knob** controls the tension by the **bail rod** on your paper. When pushed forward, it is in a fixed position. To release the paper push this knob to the back.

The **printer** itself is built into your Sr. Partner. This protects the printer and makes its use more convenient for you to operate. Because the printer is internal there are no connections necessary to begin operations.

The printer does not have a separate power switch. It is controlled by the Sr. Partner's main power switch.

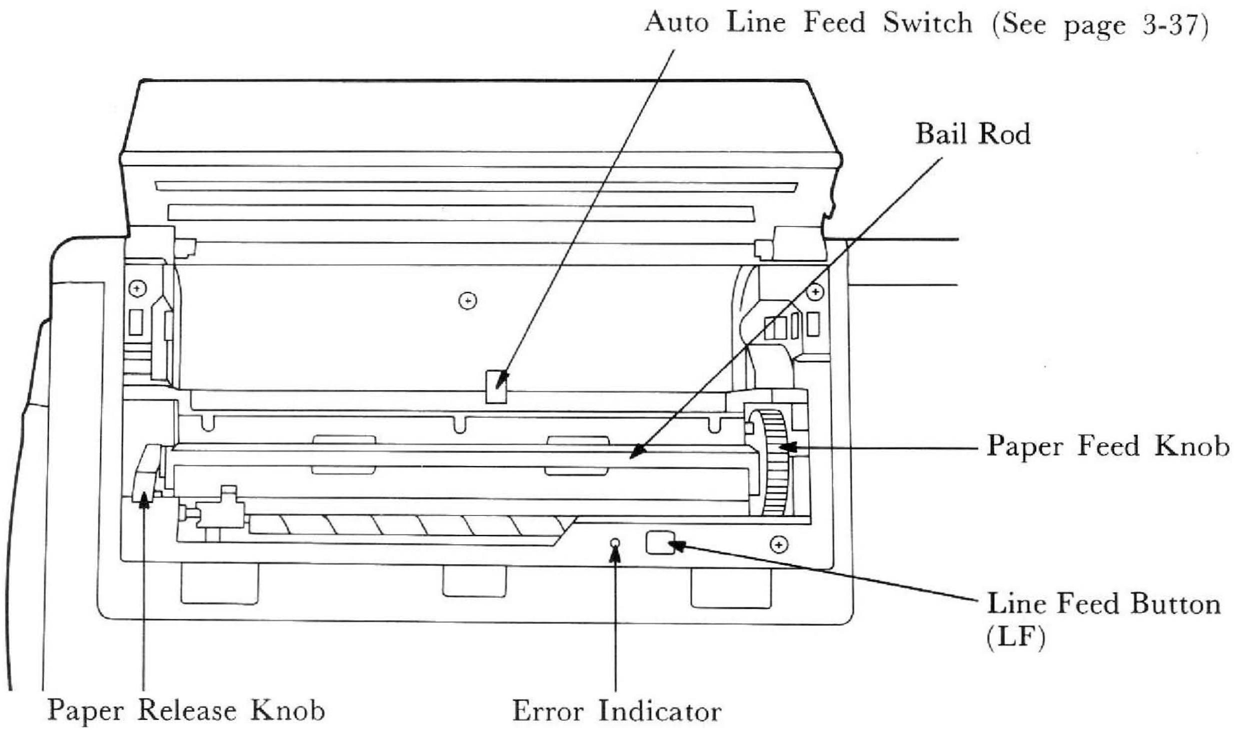


Fig. 3-21. TOP VIEW OF PRINTER

Inserting Paper Into the Printer

Included in your Sr. Partner system is a roll of thermal paper. Because your Sr. Partner uses a thermal process of printing, you must use the Panasonic thermal paper (RD-9671). YOU CANNOT USE ORDINARY OR OTHER THERMAL PAPER IN THE PRINTER.

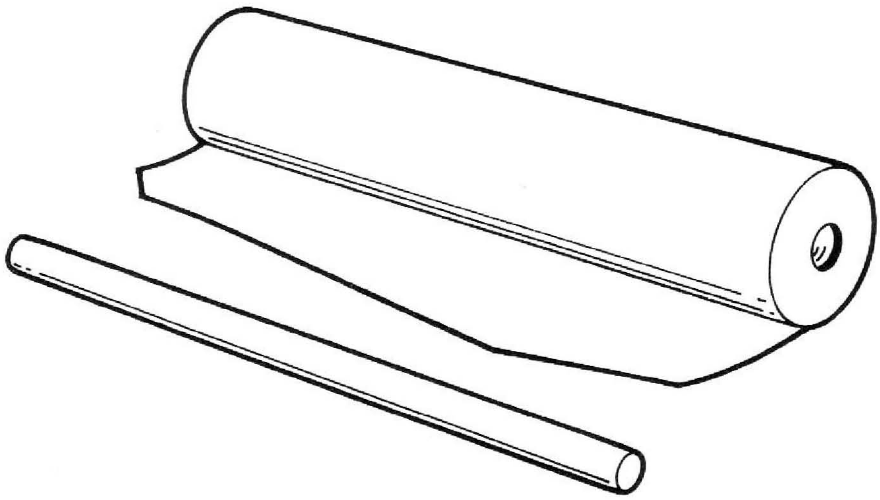


Fig. 3-22. PAPER ROLL (CUT EDGE), PAPER ROD

Push the paper release knob to the rear to release pressure on the print paper.

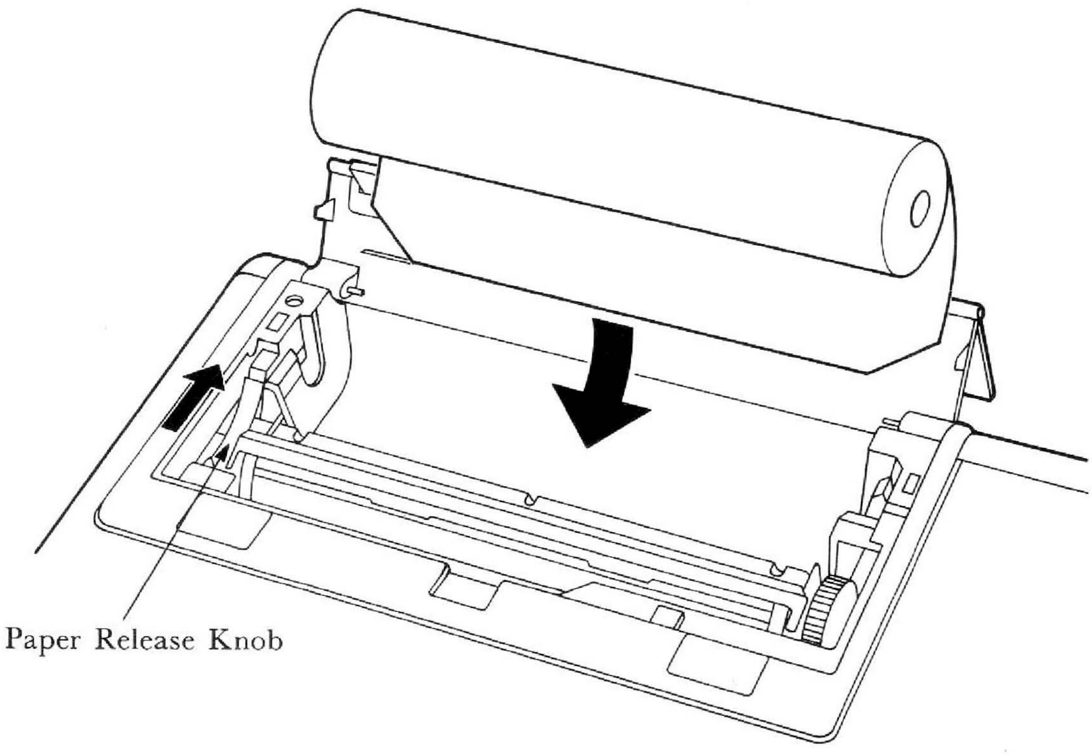


Fig. 3-23. PAPER RELEASE KNOB IN OPEN POSITION

Insert the paper as shown in Fig. 3-23, Fig. 3-24.

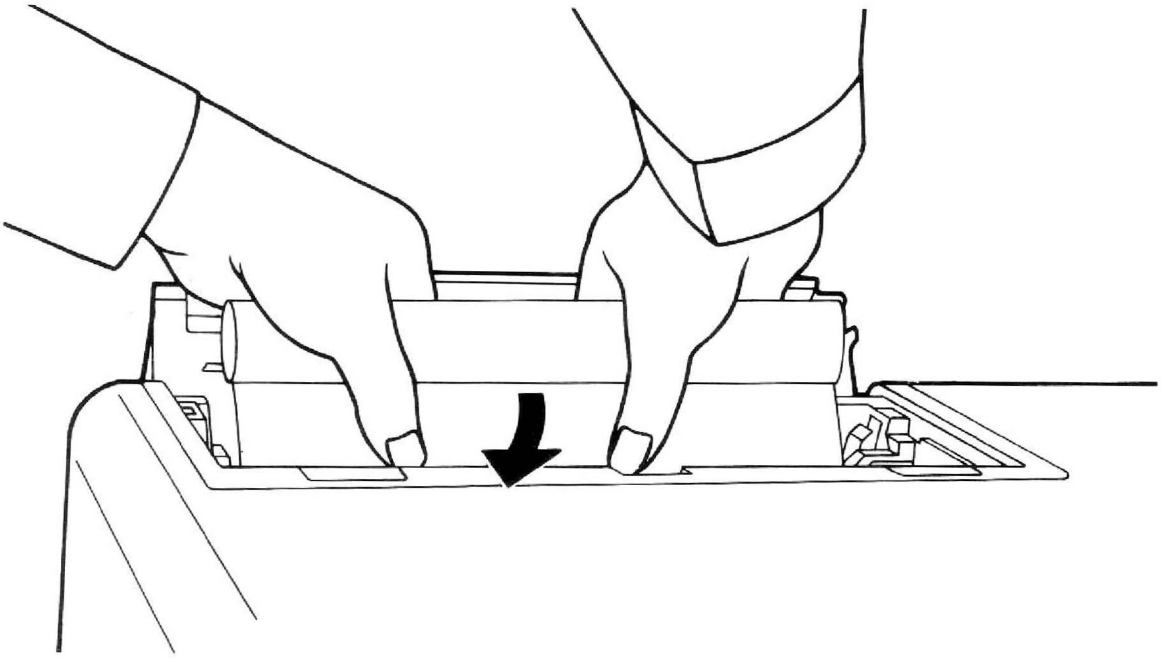


Fig. 3-24. INSERTING THE PAPER

When the edge of the paper reaches under the bail rod, pull up the edge of the paper.

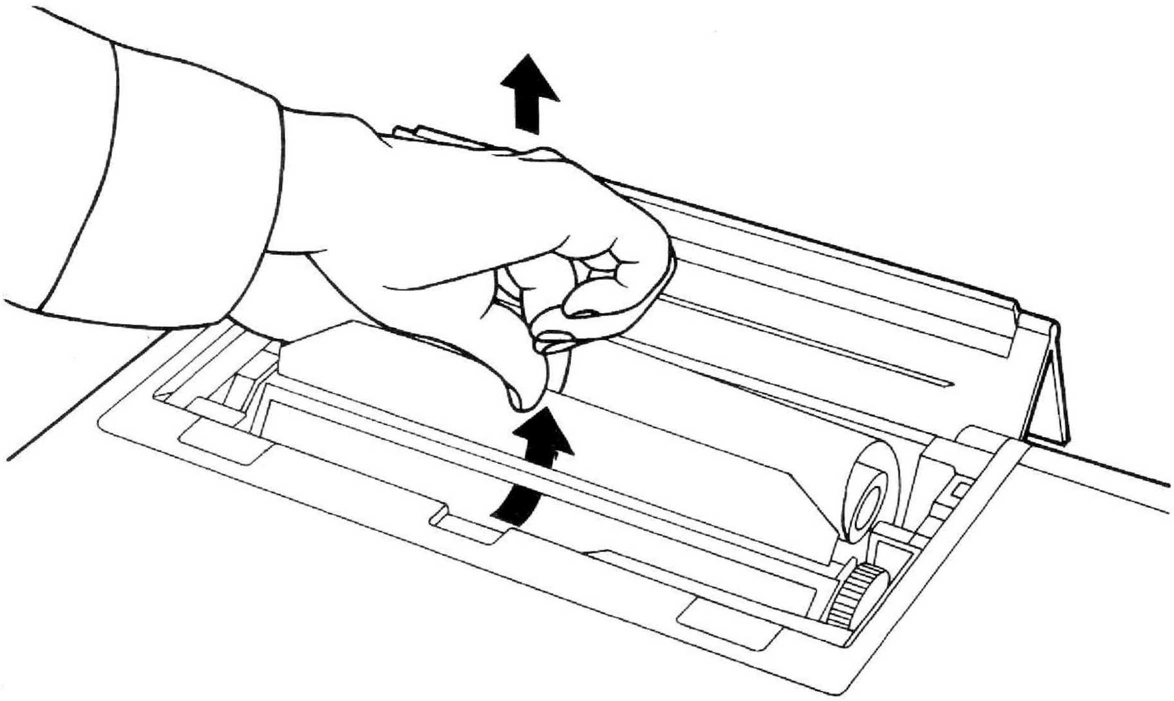


Fig. 3-25. PULLING UP THE PAPER

SET UP

Once the paper is visible, you can set the paper release knob in the fixed position.

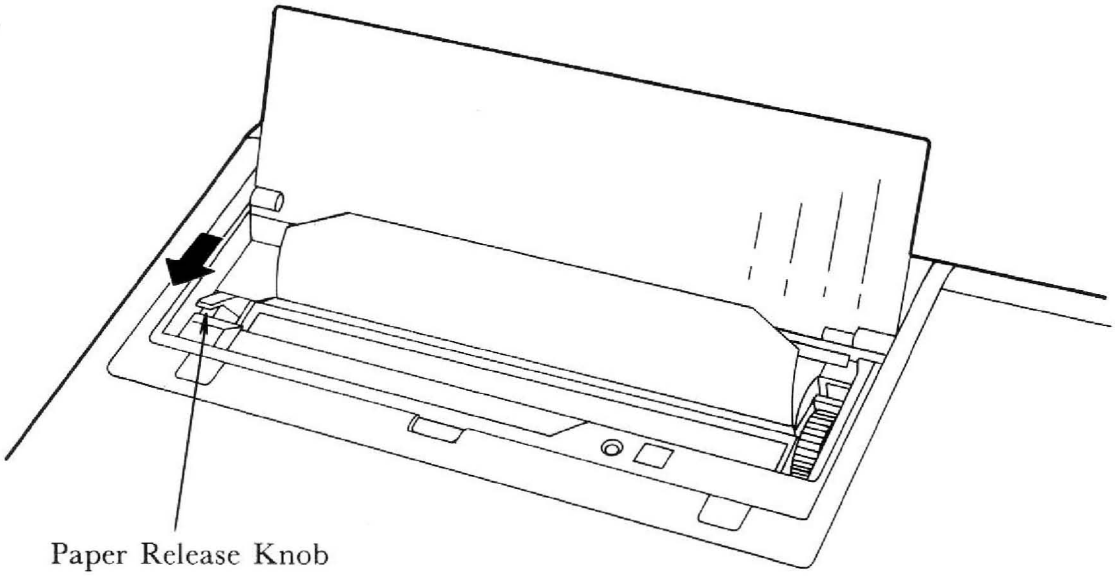


Fig. 3-26. SETTING PAPER RELEASE KNOB IN FIXED POSITION

Insert the paper rod into the roll of paper and place it in the shallow indentation on the left hand side of the printer compartment.

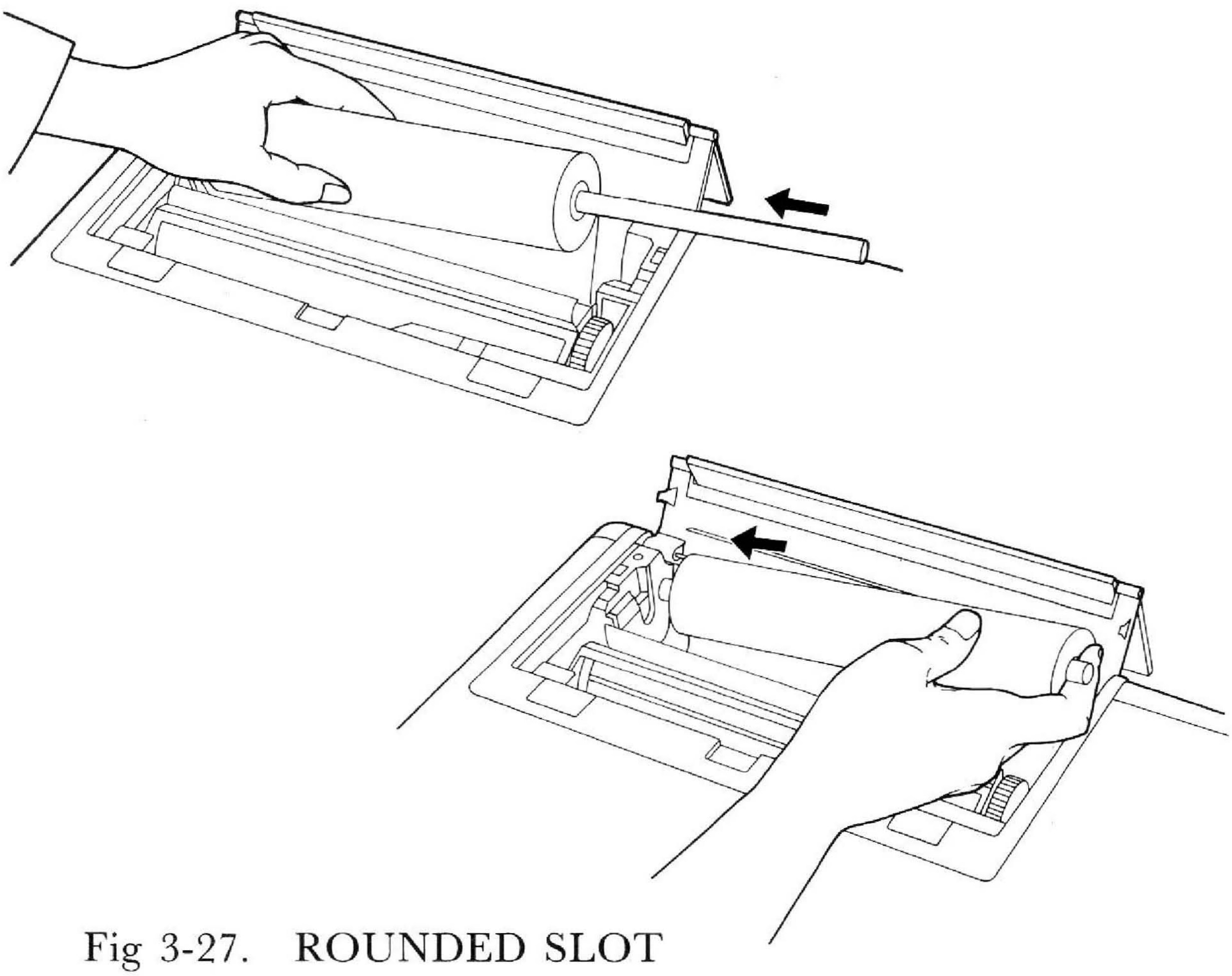


Fig 3-27. ROUNDED SLOT

Slide the rod downward until the paper is resting on the stop. It should be held firmly in place by the spring.

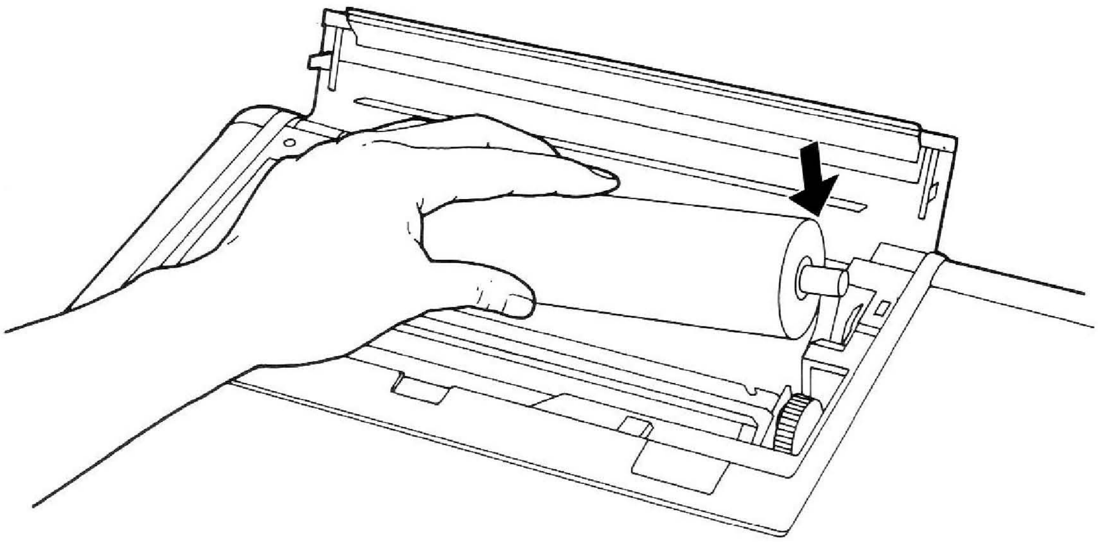


Fig. 3-28. SPRING ON RIGHT HAND EDGE

Continue feeding the paper using the paper feed knob.

NOTE: You may also feed the paper using the line feed button. However, this method works only when the computer is turned on and the paper is properly seated under the bail rod. It is most useful for advancing the paper one line at a time during printing operations.

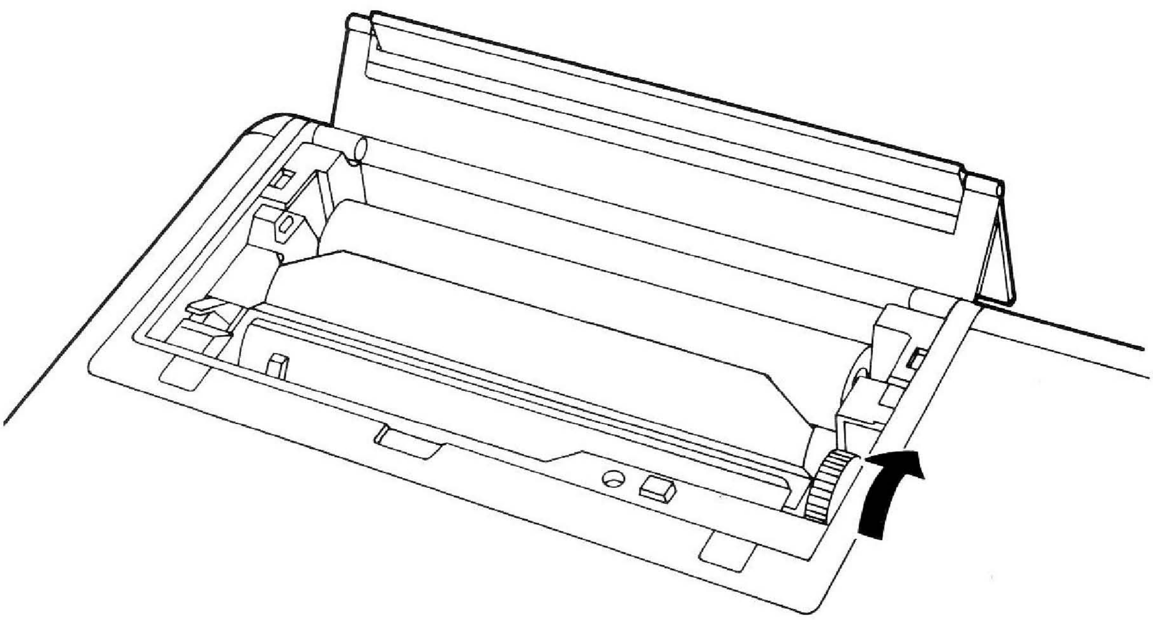
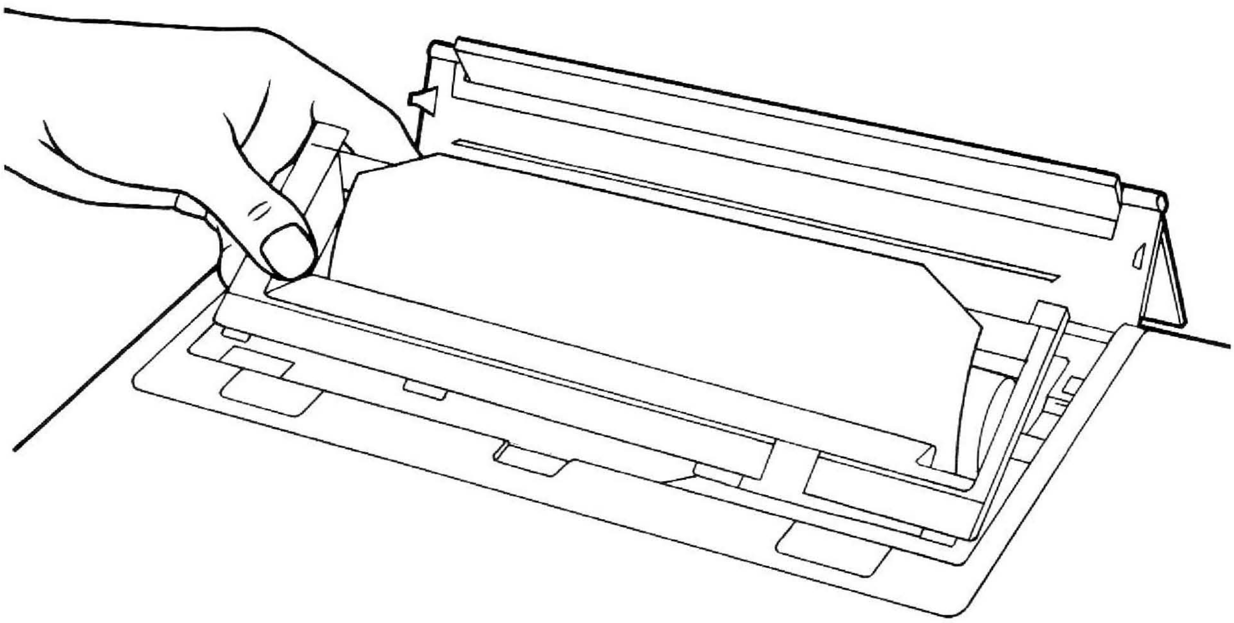


Fig. 3-29. PAPER FEEDING



When the paper is feeding correctly, put the plastic cover to its place. This cover prevents dirt and other objects from falling into the printer.

Fig. 3-30. PLASTIC COVER HOLDED

Fold the cover forward so that it is resting on the hinge. The top half of the cover now serves as a rest for the paper, feeding it up and over the cover to prevent jams when you are printing data. The printer is now ready for use. If your configurations is complete, turn to Chapter 4.

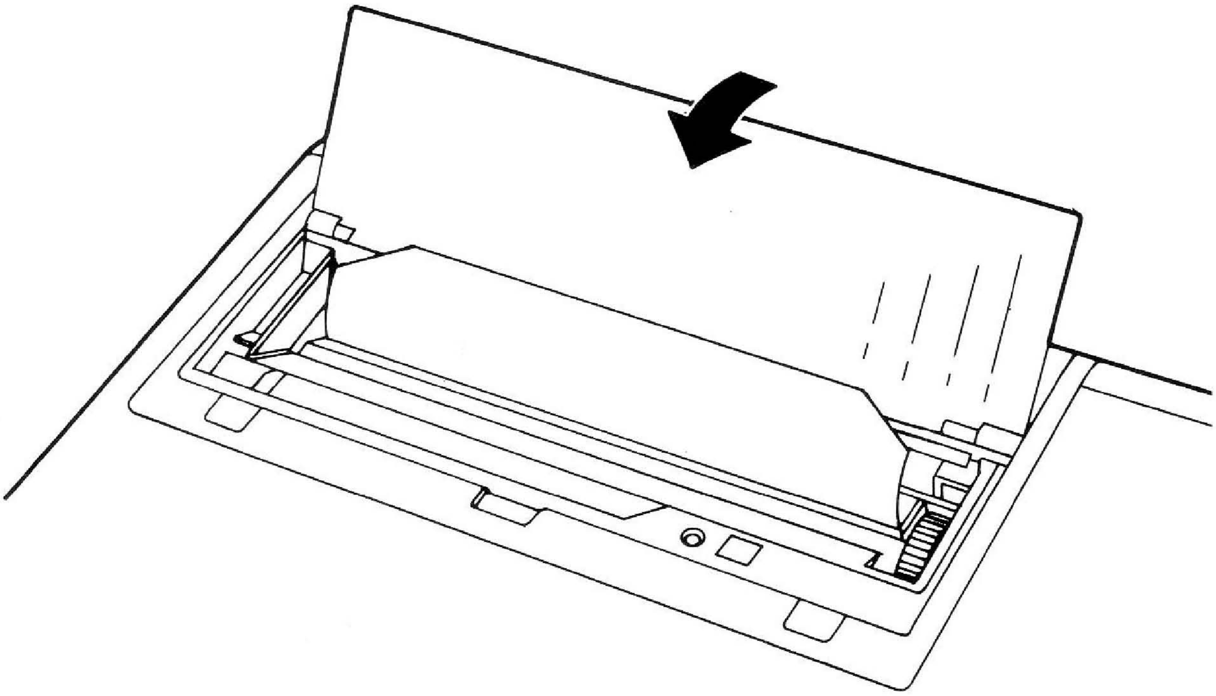


Fig. 3-31. COVER RETURNED TO HALF OPEN POSITION

SET UP

*** WARNING ***

Be careful that the lower edge of the top half of the cover must be under the plastic cover.

Removing Paper From the Printer

Before attempting to remove the paper from the printer, fold the printer cover all the way back and remove the plastic cover. Push the release knob to the back.

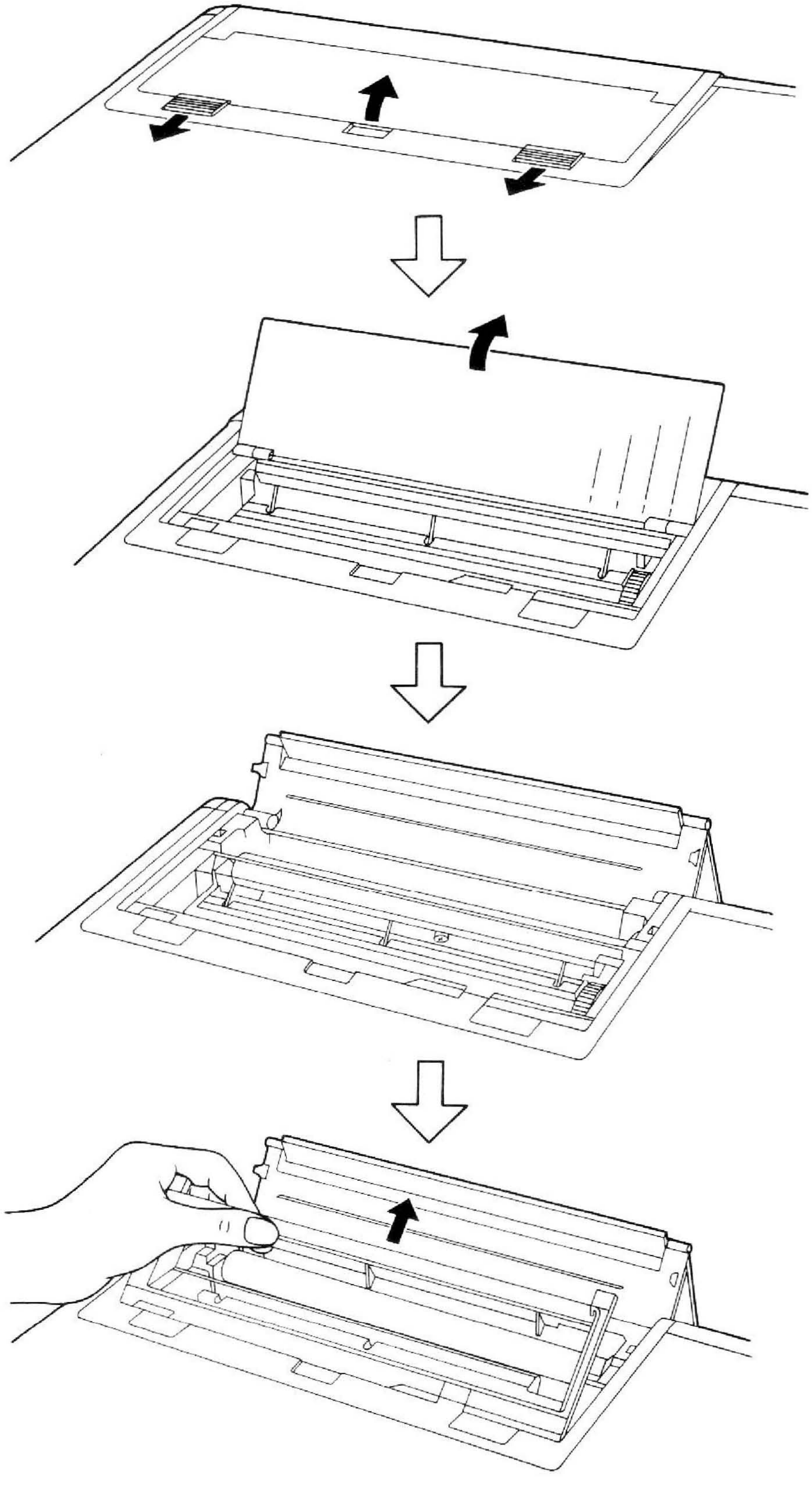


Fig. 3-32. PAPER IN POSITION

To remove paper from the printer, first push outward on the spring located at the right edge of the paper compartment. Keep pressure on the spring as you lift out the paper roll. Once the paper roll is clear of the spring, lift the left edge out of the indentation of the left hand side.

SET UP

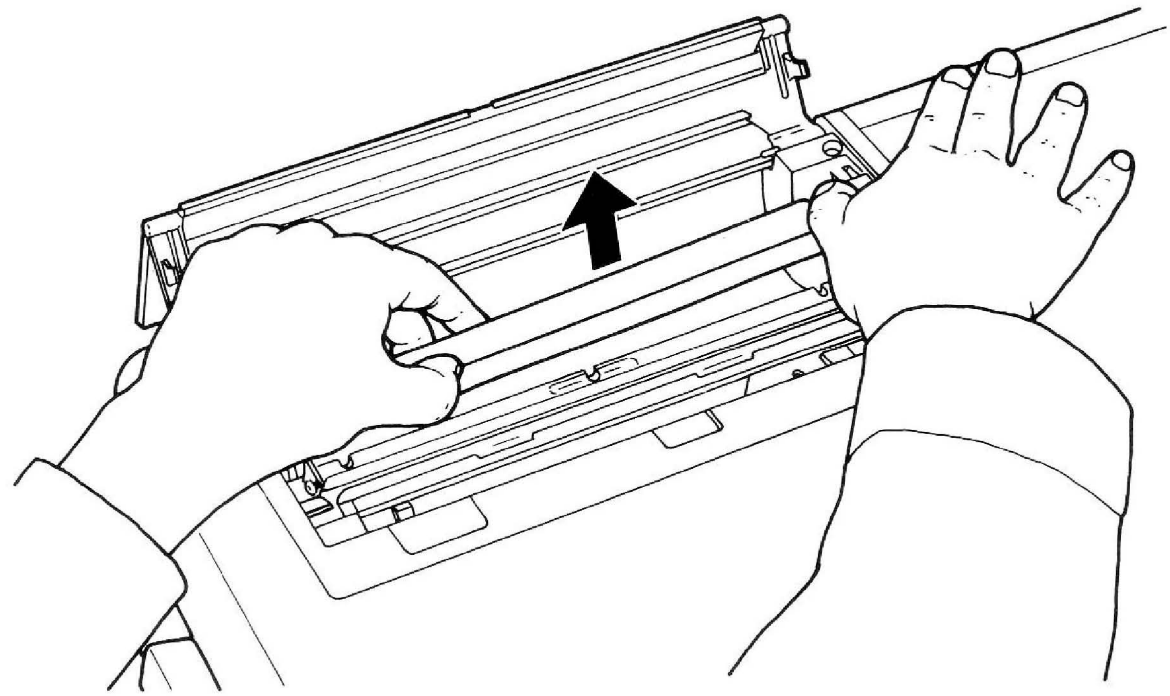


Fig. 3-33. PAPER BEING REMOVED

Push the paper release knob to forward, then place the plastic cover on the printer compartment. Return the printer cover to the closed position.

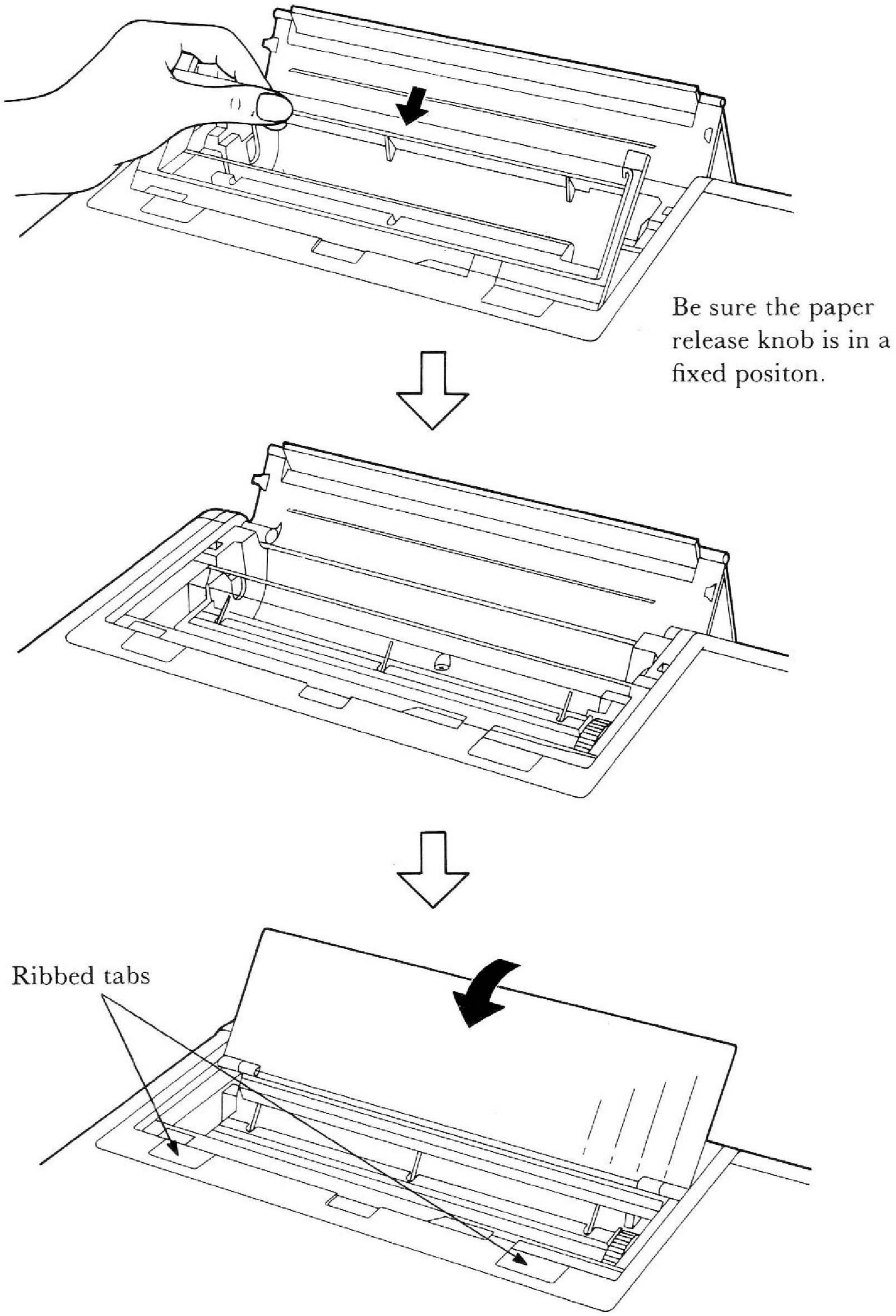


Fig. 3-34. PRINTER COVER CLOSED

Push two ribbed tabs to the back to fix the printer cover.

Printer Control Codes

You must tell the printer how you want a certain printout to be displayed. You can change many of the characteristics of the printing type and format. You set up the printer format through printer control codes.

You use BASIC commands to set the parameters of the printer output. Some experience with BASIC is necessary for you to understand the following examples.

> **NOTE:** Control codes and escape sequences of the internal printer are based on EPSON MX-80 printer.

To send control codes to the internal printer use this BASIC statement:

```
LPRINT CHR$ (ASCII #)
```

where ASCII # is one of the following.

ASCII #	Function
9	Tab (tabulation every 8 th space)
10	Line feed
12	Form Feed (change page)
13	Carriage return
14	Double width mode If carriage return (autolinefeed switch is on), line feed, form feed or printing by buffer-full is performed, this mode is automatically canceled.
15	Compressed mode
18	Cancel compressed mode
20	Cancel double width mode
24	Clear printer buffer
27	Begin Escape Sequence
127	Clear printer buffer and cancel double width mode

Printer Escape Sequences

The printer escape sequences list the first and second parameters in determining output format. In BASIC the escape sequence statement takes this style:

```
LPRINT CHR$ (27); CHR$ (Code#1); CHR$ (Code#2)
```

CHR$ (27) begins the escape sequence:

<Esc> and <_> Underline mode

```
LPRINT CHR$ (27);CHR$ (45);CHR$ (0)
LPRINT CHR$ (27);CHR$ (45);CHR$ (1)
```

1 begins the underline mode, 0 cancels it.

<Esc> and <0> Set line feed 1/8 inches

```
LPRINT CHR$ (27);CHR$ (48)
```

<Esc> and <1> Set line feed 7/85 inches

```
LPRINT CHR$ (27);CHR$ (49)
```

<Esc> and <2> Set line feed 1/6 inches (See <Esc> and <A>)

```
LPRINT CHR$ (27);CHR$ (50)
```

<Esc> and <3> Set line feed n/255 inches (multiple lines, $1 \leqq n \leqq 255$)

```
LPRINT CHR$ (27);CHR$ (51);CHR$ (n)
```

<Esc> and <6> Print using Character Set 2

```
LPRINT CHR$ (27);CHR$ (54)
```

<Esc> and <7> Print using Character Set 1

```
LPRINT CHR$(27);CHR$(55)
```

<Esc> and <<>	Return to left margin for next line
	LPRINT CHR$(27);CHR$(60)
<Esc> and <@>	Returns the printer to its initial state
	LPRINT CHR$(27);CHR$(64)
<Esc> and <A>	Set the line feed to n/85 inches ($1 \leqq n \leqq 85$)
	To change the variable line feed, you must send <Esc> and <2> to the printer.
	LPRINT CHR$(27);CHR$(65);CHR$(n)
<Esc> and <C>	Set page length to n lines (max 127 lines)
	LPRINT CHR$(27);CHR$(67);CHR$(n)
	Set page length to n inches
	LPRINT CHR$(27);CHR$(67);CHR$(00);CHR$(n)
<Esc> and <E>	Set printer to Emphasized mode
	LPRINT CHR$(27);CHR$(69)
<Esc> and <F>	Cancel Emphasized mode
	LPRINT CHR$(27);CHR$(70)
<Esc> and <G>	Set printer to Double Strike mode
	LPRINT CHR$(27);CHR$(71)
<Esc> and <H>	Cancel Double Strike mode
	LPRINT CHR$(27);CHR$(72)
<Esc> and <J>	Set line feed n/255 inch (one line, $1 \leqq n \leqq 255$)
	LPRINT CHR$(27); CHR$(74); CHR$(n)

<Esc> and <K>

Change from Text mode to Bit Image Graphics mode (Normal-density)

```
LPRINT CHR$(27);CHR$(75);CHR$(n1);CHR$(n2)
```

n_1, n_2 represents the number of bit image data. n_1 is the remainder of the number of the data over 256, and n_2 is its quotient. The number of the data must be less than 480.

<Esc> and <L>

Change from Text mode to Bit Image Graphics mode (Dual-density). Operation is same as <Esc> <K> except that the maximum number of data is 960.

<Esc> and <N>

Set Skip Perforation function
n (number of lines to skip) must range from 1 to 127

```
LPRINT CHR$ (27);CHR$ (78);CHR$(n)
```

<Esc> and <O>

Cancel Skip Perforation function

```
LPRINT CHR$ (27);CHR$ (79)
```

<Esc> and <U>

Unidirectional printing

```
LPRINT CHR$ (27);CHR$ (85);CHR$ (1)
LPRINT CHR$ (27);CHR$ (85);CHR$ (0)
```

1 begins unidirectional printing, 0 cancels it.

<Esc> and <W>

Set printer to Double width mode

```
LPRINT CHR$ (27);CHR$ (87);CHR$ (1)
LPRINT CHR$ (27);CHR$ (87);CHR$ (0)
```

1 begins Double Width printing, 0 cancels it.

SET UP

Setting Autolinefeed Switch

You can set autolinefeed by the switch inside the internal printer compartment.

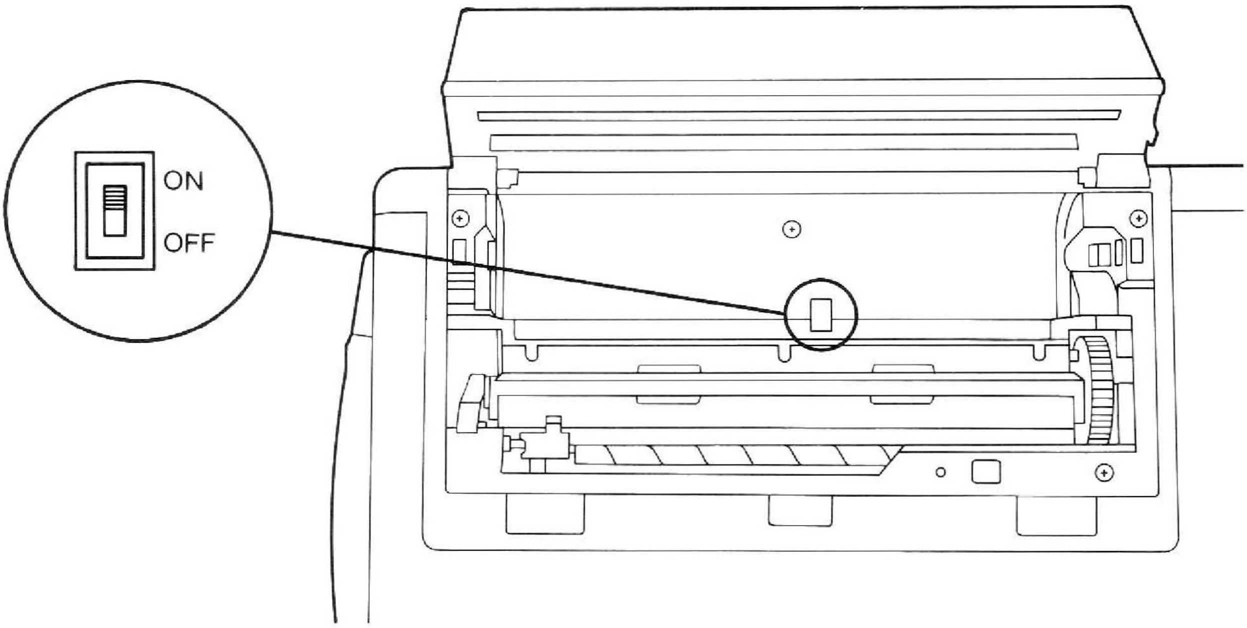


Fig. 3-35. SETTING AUTOLINEFEED SWITCH

If you set the swtich to ON position, the printer performs carriage return and linefeed by CR (Carriage Return) code. OFF position is a default and the printer performs only carriage return by CR code.

If you have changed the switch, reset the printer by pressing <Alt> <Esc> or restarting your System Disk (<Ctrl> <Alt> <Del>).

Special keys for the printer

<SHIFT>+<PrtSc> Print what is on the screen

<Ctrl>+<PrtSc> Print what you are typing

<Alt>+<Esc> Reset the printer

<Alt>+<PrtSc> Switch the printer to an external printer

Character Sets

CHARACTER SET 1

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

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	128 NUL	129	130	131	132	133	134	135	136	137 HT	138 LF	139	140 FF	141 CR	142 SO	143 SI
9	144	145	146 DC2	147	148 DC4	149	150	151	152 CAN	153	154	155 ESC	156	157	158	159
A	160 á	161 í	162 ó	163 ú	164 ñ	165 Ñ	166 ª	167 º	168 ¿	169 ⌐	170 ¬	171 ½	172 ¼	173 ¡	174 «	175 »
B	176 ░	177 ▒	178 ▓	179 │	180 ┤	181 ╡	182 ╢	183 ╖	184 ╕	185 ╣	186 ║	187 ╗	188 ╝	189 ╜	190 ╛	191 ┐
C	192 └	193 ┴	194 ┬	195 ├	196 ─	197 ┼	198 ╞	199 ╟	200 ╚	201 ╔	202 ╩	203 ╦	204 ╠	205 ═	206 ╬	207 ╧
D	208 ╨	209 ╤	210 ╥	211 ╙	212 ╘	213 ╒	214 ╓	215 ╫	216 ╪	217 ┘	218 ┌	219 █	220 ▄	221 ▌	222 ▐	223 ▀
E	224 α	225 β	226 Γ	227 π	228 Σ	229 σ	230 μ	231 τ	232 Φ	233 Θ	234 Ω	235 δ	236 ∞	237 ∅	238 ε	239 ∩
F	240 ≡	241 ±	242 ≥	243 ≤	244 ⌠	245 ⌡	246 ÷	247 ≈	248 °	249 ∙	250 ·	251 √	252 ^{n}	253 ^{2}	254 ■	255 SP

SET UP

CHARACTER SET 2

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

SET UP

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	128 ç	129 ü	130 é	131 â	132 ä	133 à	134 å	135 ç	136 ê	137 ë	138 è	139 ï	140 î	141 ì	142 Ä	143 Â
9	144 É	145 æ	146 Æ	147 ô	148 ö	149 ò	150 û	151 ù	152 ÿ	153 ö	154 ü	155 ¢	156 £	157 ¥	158 ₧	159 ƒ
A	160 á	161 í	162 ó	163 ú	164 ñ	165 Ñ	166 ª	167 º	168 ¿	169 ⌐	170 ¬	171 ½	172 ¼	173 ¡	174 «	175 »
B	176 ░	177 ▒	178 ▓	179 │	180 ┤	181 ┤	182 ┤	183 ┐	184 ┐	185 ┤	186 │	187 ┐	188 ┘	189 ┘	190 ┘	191 ┐
C	192 └	193 ┴	194 ┬	195 ├	196 ─	197 ┼	198 ├	199 ├	200 └	201 ┌	202 ┴	203 ┬	204 ├	205 ─	206 ┴	207 ┬
D	208 ┴	209 ┼	210 ┬	211 └	212 └	213 ┌	214 ┌	215 ┼	216 ┼	217 ┘	218 ┌	219 █	220 ▄	221 ▌	222 ▐	223 ▀
E	224 α	225 β	226 Γ	227 π	228 Σ	229 σ	230 µ	231 τ	232 Φ	233 Θ	234 Ω	235 δ	236 ∞	237 ∅	238 ε	239 ∩
F	240 ≡	241 ±	242 ≥	243 ≤	244 ⌠	245 ⌡	246 ÷	247 ≈	248 °	249 ■	250 -	251 √	252 ∩	253 ²	254 ■	255 SP

OPTIONS

SET UP

There are several options available for your Sr. Partner system. With these options you can customize your system so that your Sr. Partner reflects your needs.

Built in RAM Board allows you to add additional RAM memory to your Sr. Partner. Your first RAM expansion will add 64K to the exisiting 256K of your system. Subsequent additions can be made in 64 K increments up to a total fo 512K! See your Memory Expansion Guide.

You may also add an additonal disk drive if you have Sr. Partner with one disk drive. This very wise investment allows you to access two disks without stopping to interchange the floppies. Not only do two drives increase your access to information retrieval and storage, they make many simple household tasks, such as copying, much more efficient. See your Panasonic dealer for information on adding a drive B to your Sr. Partner.

If you desire a color monitor, with larger display area and increased resolution you may add a RGB monitor to your system. For details on color monitors, see your dealer.

If you add options to be installed inside your Sr. Partner, see the next section for installing, and the individual manuals accompanying the option for specific operating instructions.

INSTALLATION OPTIONAL BOARDS

NOTE: Unless you are experienced with computers, Panasonic recommends that your dealer install and test the optional board before you take delivery of your Sr. Partner.

If you are doing the installation yourself, you will need a Phillips screwdriver.

Preparation

STEP 1	BE SURE ALL POWER IS OFF. Disconnect the Sr. Partner from any electrical outlets. Unplug the AC cord from the rear panel.
STEP 2	Attach the keyboard to the system unit for protection of the front panel.

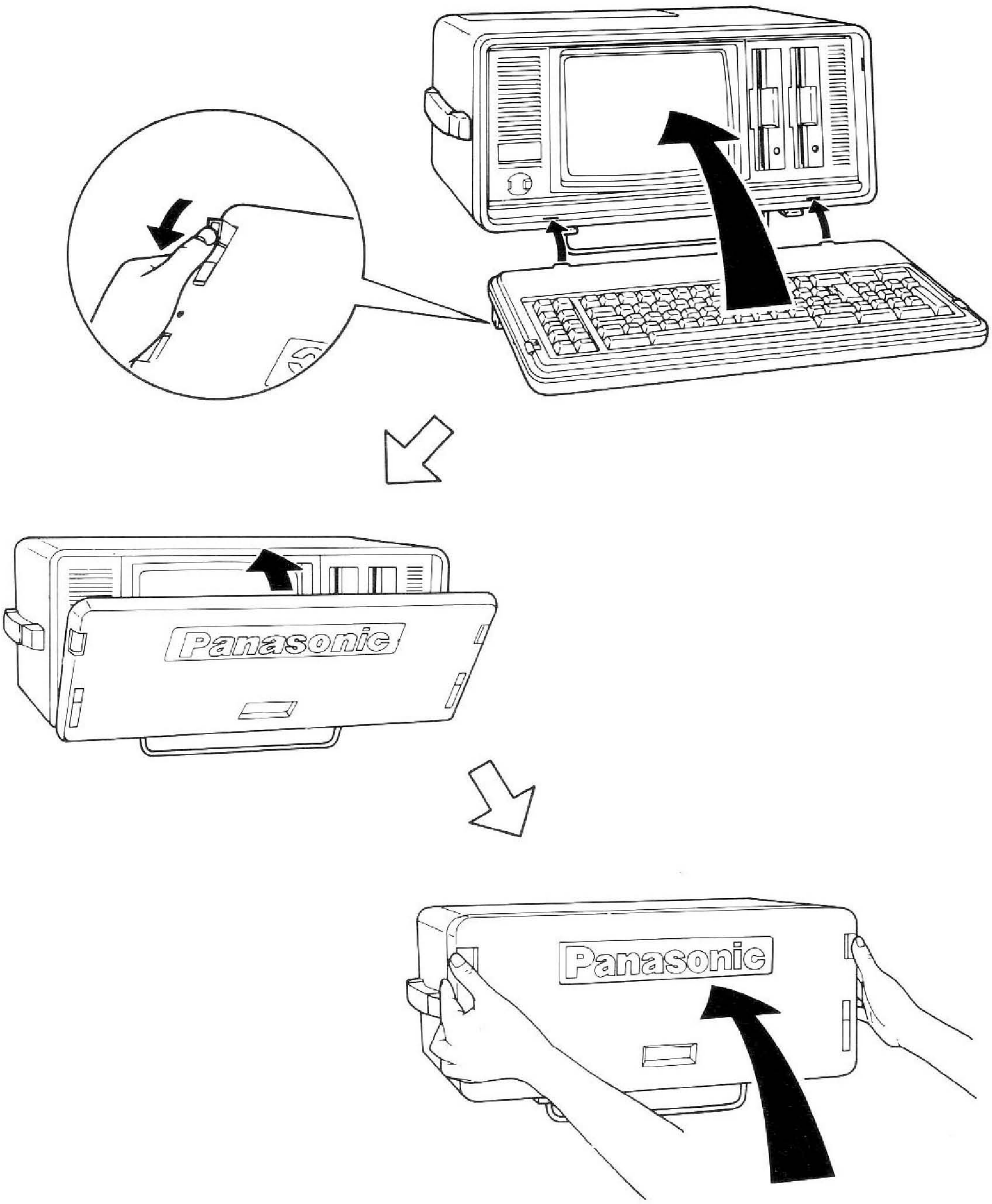


Fig. 3-36. ATTACHING THE KEYBOARD

STPE 3 Fold the metal rod over the keyboard unit.

Removing the Printer Cover

STEP 4 The two ribbed tabs release the protective cover for the printer compartment. Push these tabs forward, then slide back the first section of the printer cover. Fold the cover all the way back and remove the plastic cover.

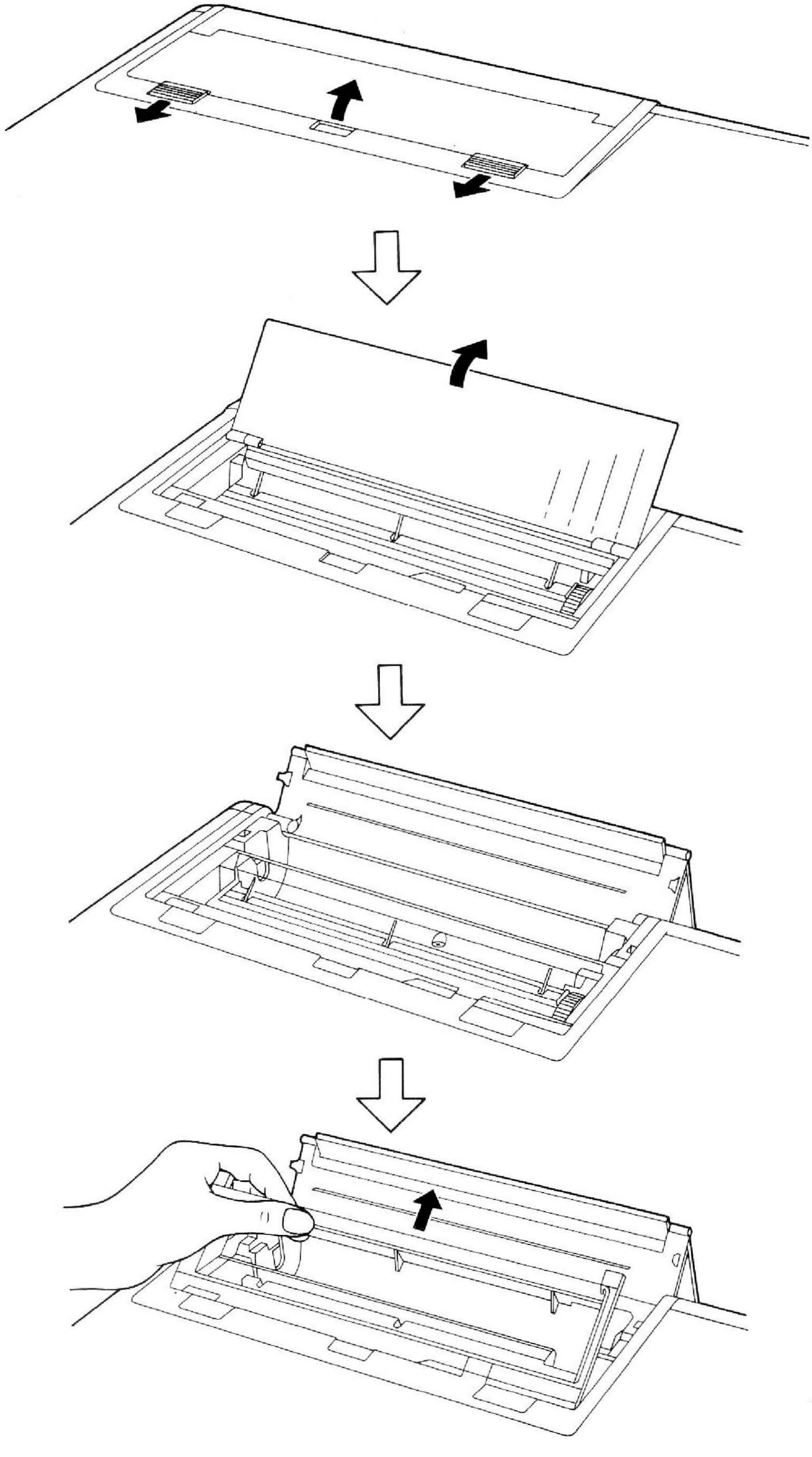


Fig. 3-37. OPENING OF PRINTER COMPARTMENT

SET UP

STEP 5 You will see four screws attaching the printer cover to the system unit.

STEP 6 Using a Phillips screwdriver, remove these four screws.

NOTE: You will be removing several sets of screws during this procedure. The screws are not interchangeable. Be sure to store the screws in a safe location and in an easily identifiable pattern.

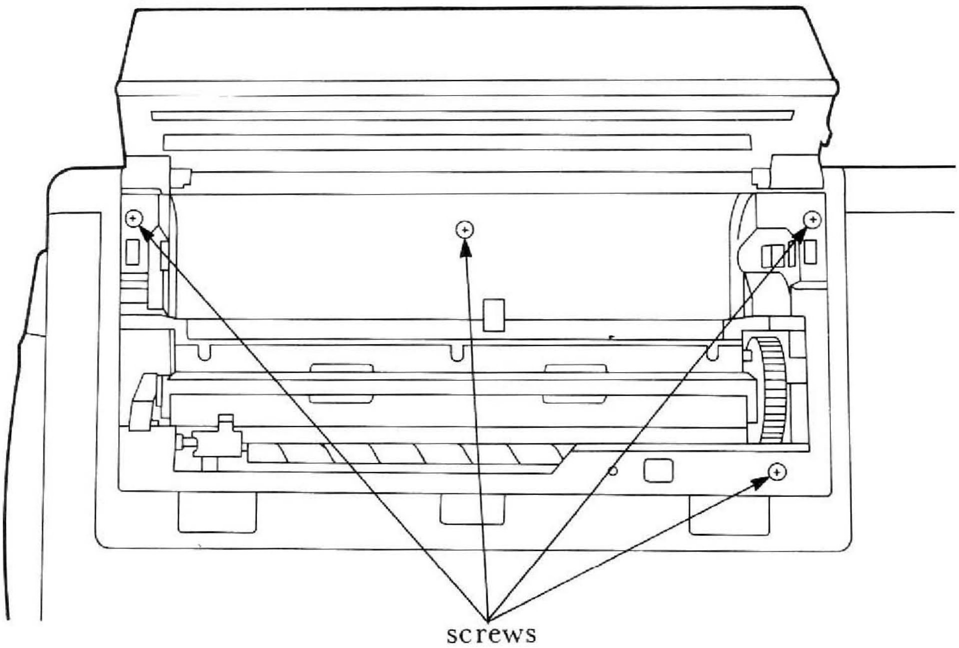


Fig. 3-38. LOCATION OF SCREWS

STEP 7 Gently lift the printer cover. Be careful not to pull on the wires connected to the printed circuit board on the inside of the cover.

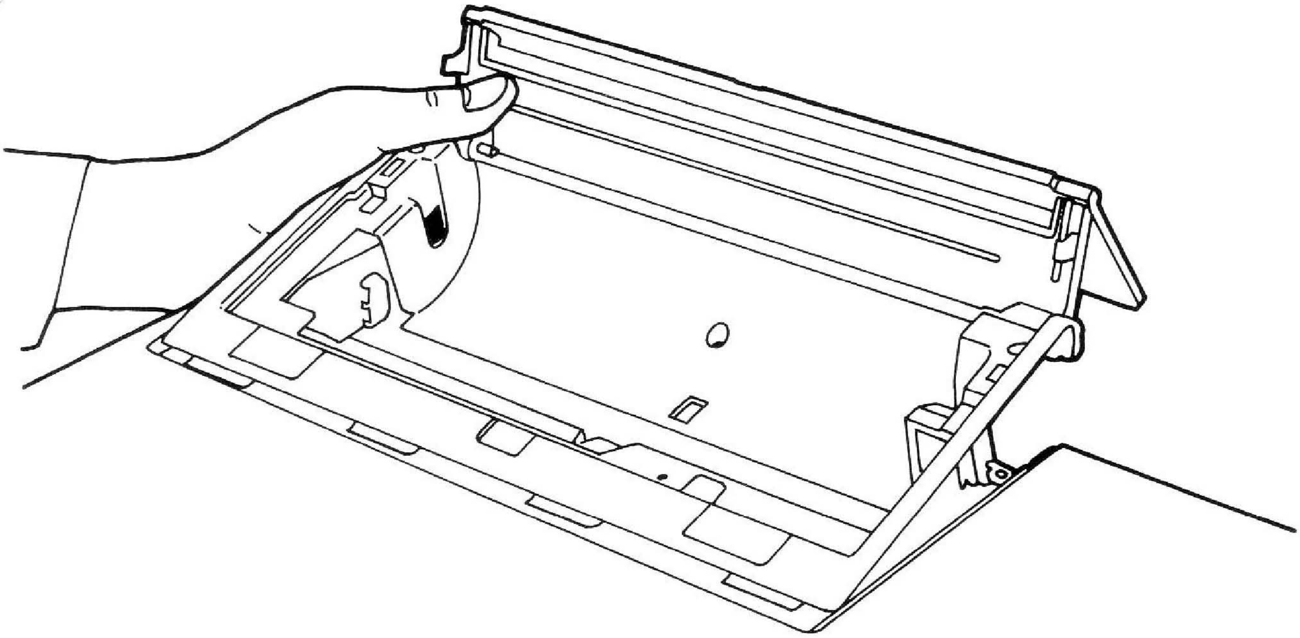


Fig. 3-39. LIFT THE PRINTER COVER

STEP 8 Disconnect the cable from the printed circuit board. The printer cover will now lift free of the system unit.

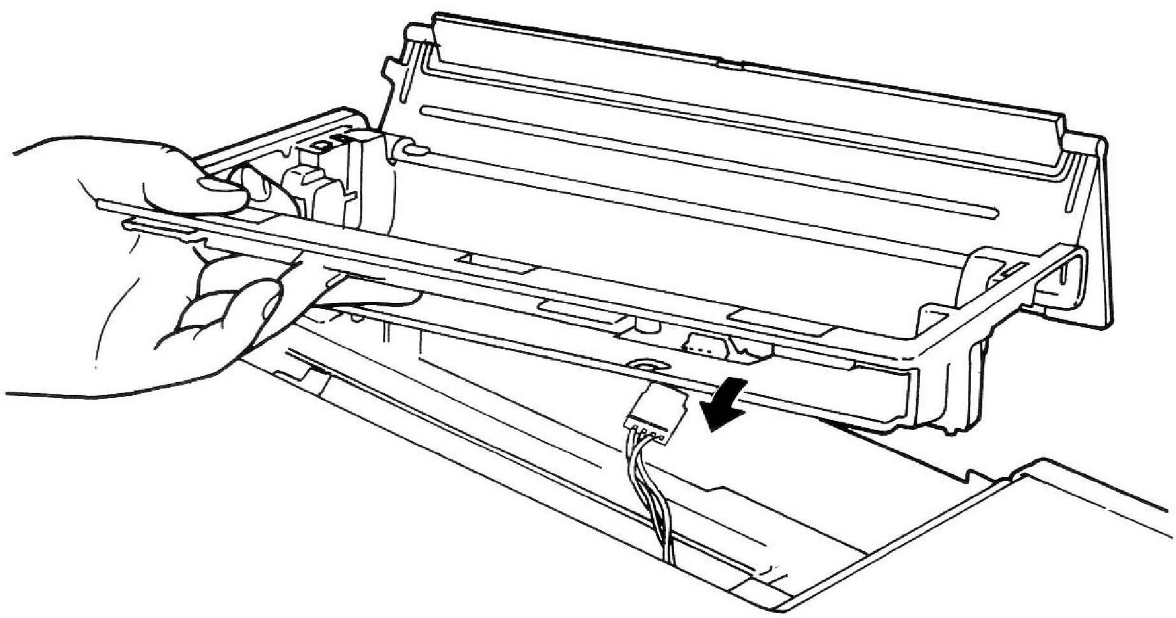


Fig. 3-40. REMOVAL OF PCB CONNECTORS

Removing the Rear Panel Cover

STEP 9 Remove the six screws which secure the rear panel cover and remove the cover. This will expose the internal aluminum panel. (Don't remove this internal aluminum panel.)

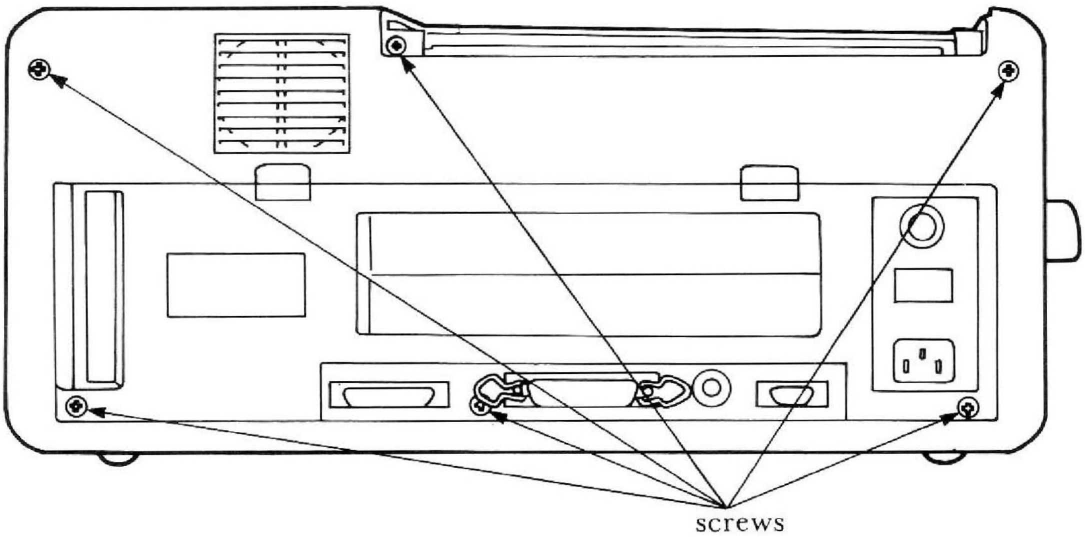


Fig. 3-41. REAR PANEL'S SCREWS

Removing the Outer Case Cover

STEP 10 Turn over the machine so that the bottom panel is facing you.

STEP 11 Grasping the case on the sides, slide it up until it clears the internal section of the system unit.

SET UP

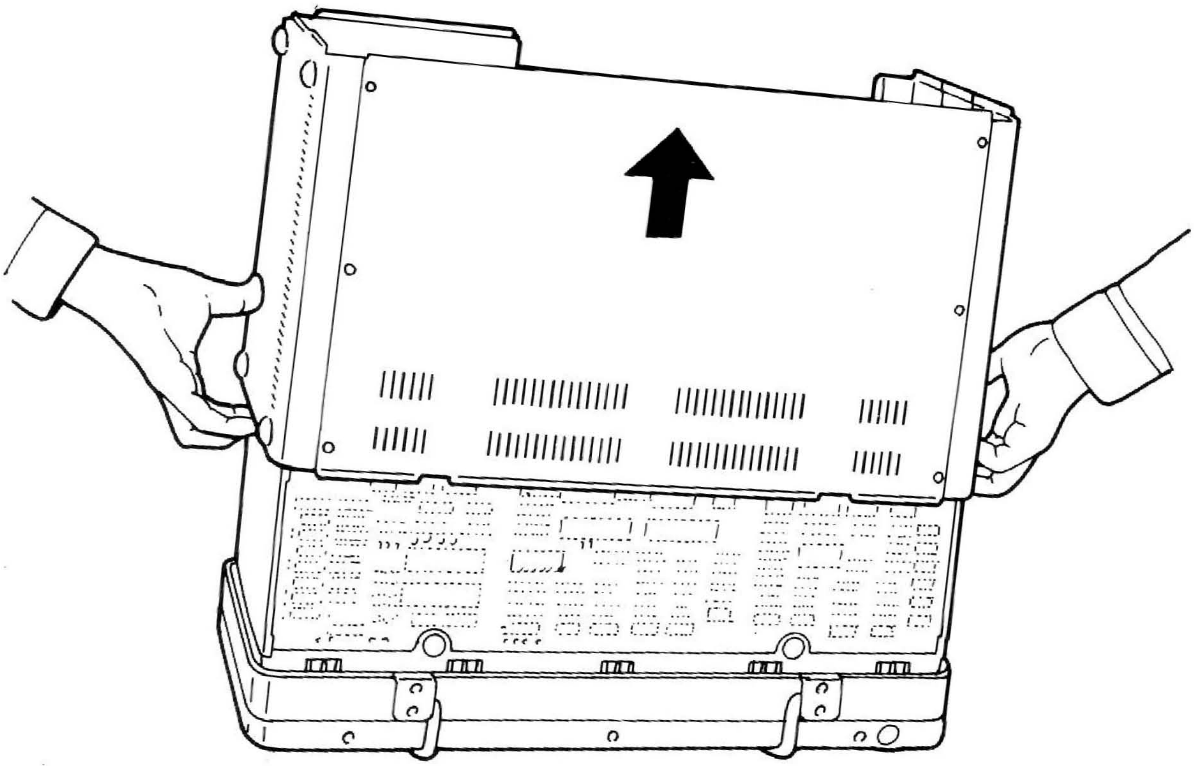


Fig. 3-42. OUTER CASE BEING SLID UPWARDS

SET UP

Inside the System Unit

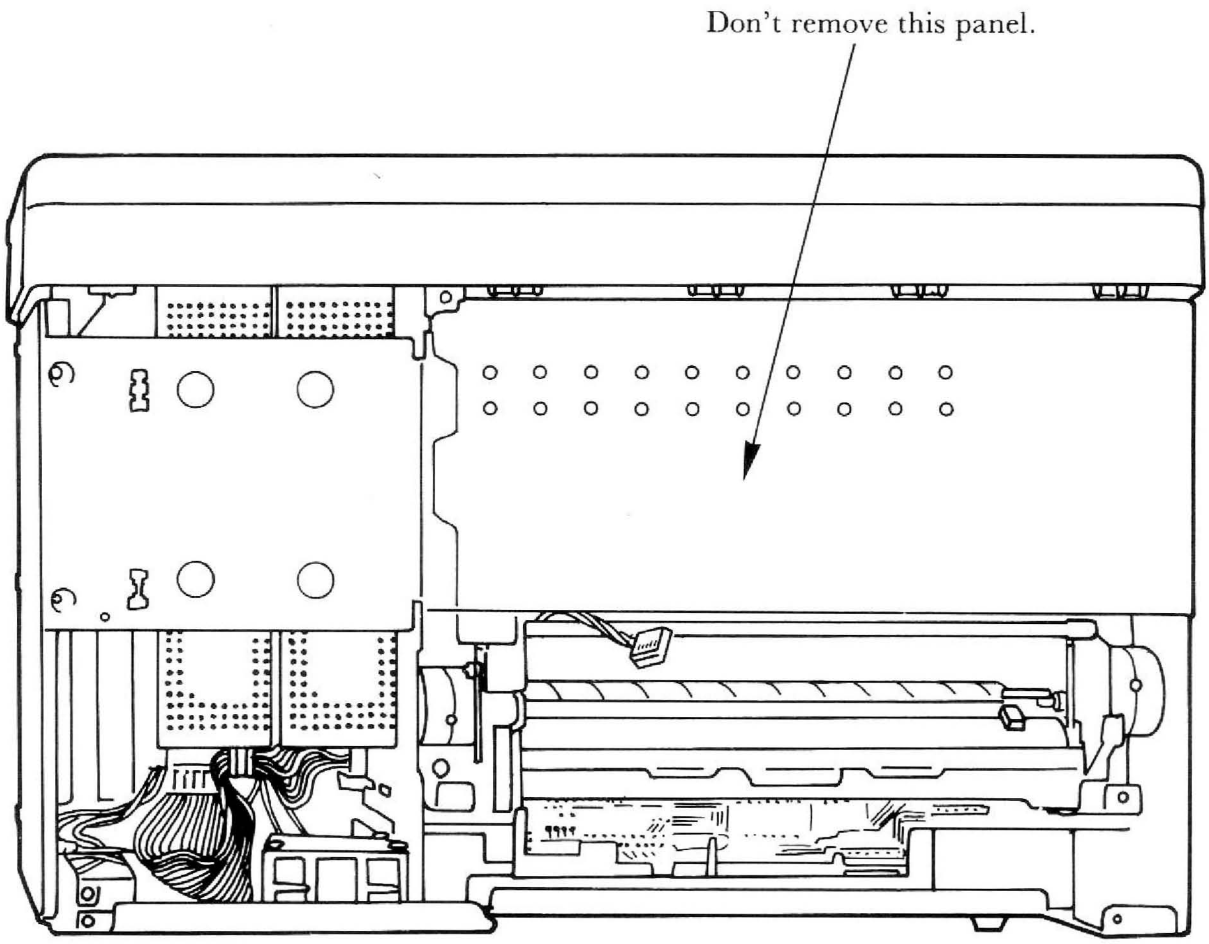


Fig. 3-43. INSIDE OF THE SYSTEM UNIT

Setting the dip swtich on the main board (if necessary)

The dip switch that sets/clears several functions is located near the disk drive shown below.

If necessary, change the dip switch according to the table on the following page. It is easier to change it BEFORE inserting an optional board.

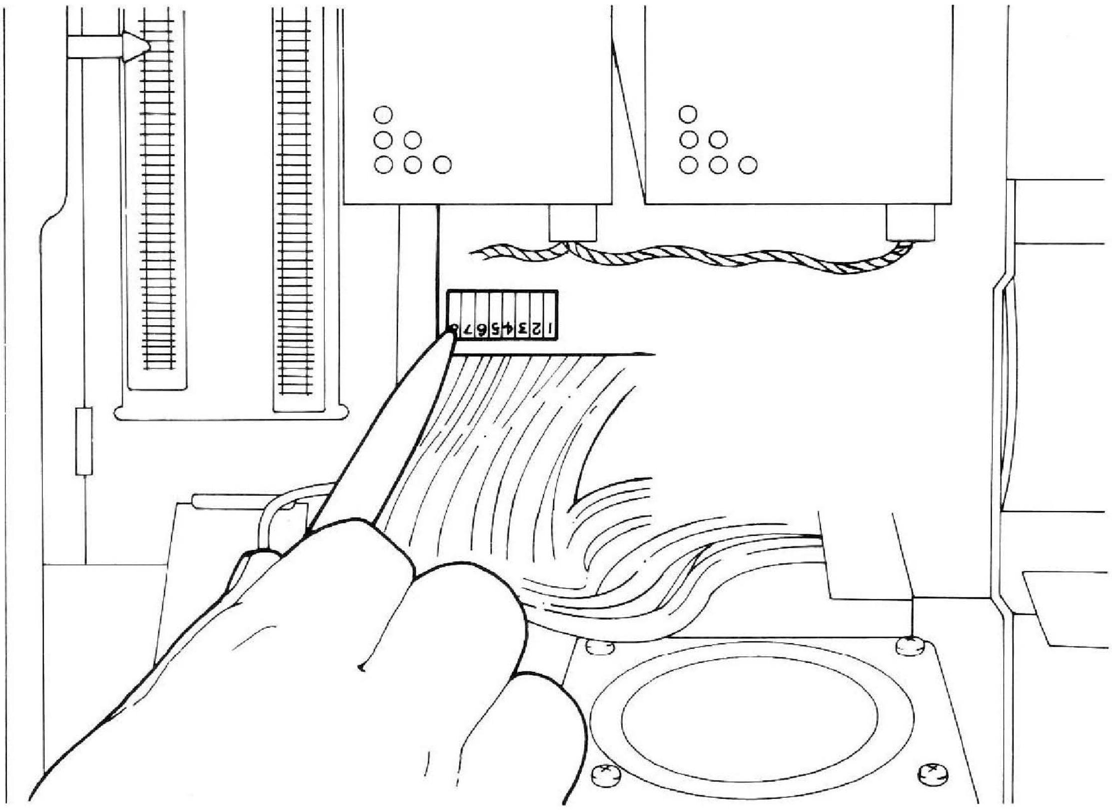


Fig. 3-44 SETTING THE SWITCHES

Function		1	2	3	4	5*	6	7	8
RAM size** (Total Size)	128K	ON	OFF	ON	ON	—	—	—	—
	256K	ON	OFF	OFF	ON	—	—	—	—
	320K	ON	ON	ON	OFF	—	—	—	—
	384K	ON	OFF	ON	OFF	—	—	—	—
	448K	ON	ON	OFF	OFF	—	—	—	—
	512K	ON	OFF	OFF	OFF	—	—	—	—
8087 implemented		—	—	—	—	—	OFF	—	—
8087 not implemented		—	—	—	—	—	ON	—	—
1 disk drive		—	—	—	—	—	—	—	OFF
2 disk dirves		—	—	—	—	—	—	—	ON
80 characters per line (VDT)		—	—	—	—	—	—	OFF	—
40 characters per line (VDT)		—	—	—	—	—	—	ON	—

Setting dip switch on the main board

*5 is reserved for future expansion.

**If you use Panasonic RAM Board, you can expand your memory. Refer to the operation manual attached to the RAM Board.

Installing optional board

STEP 1 Remove the four corner screws and the two screws securing the top panel of the disk drive unit. (If you have installed a Drive B there are also two screws holding this unit.)

SET UP

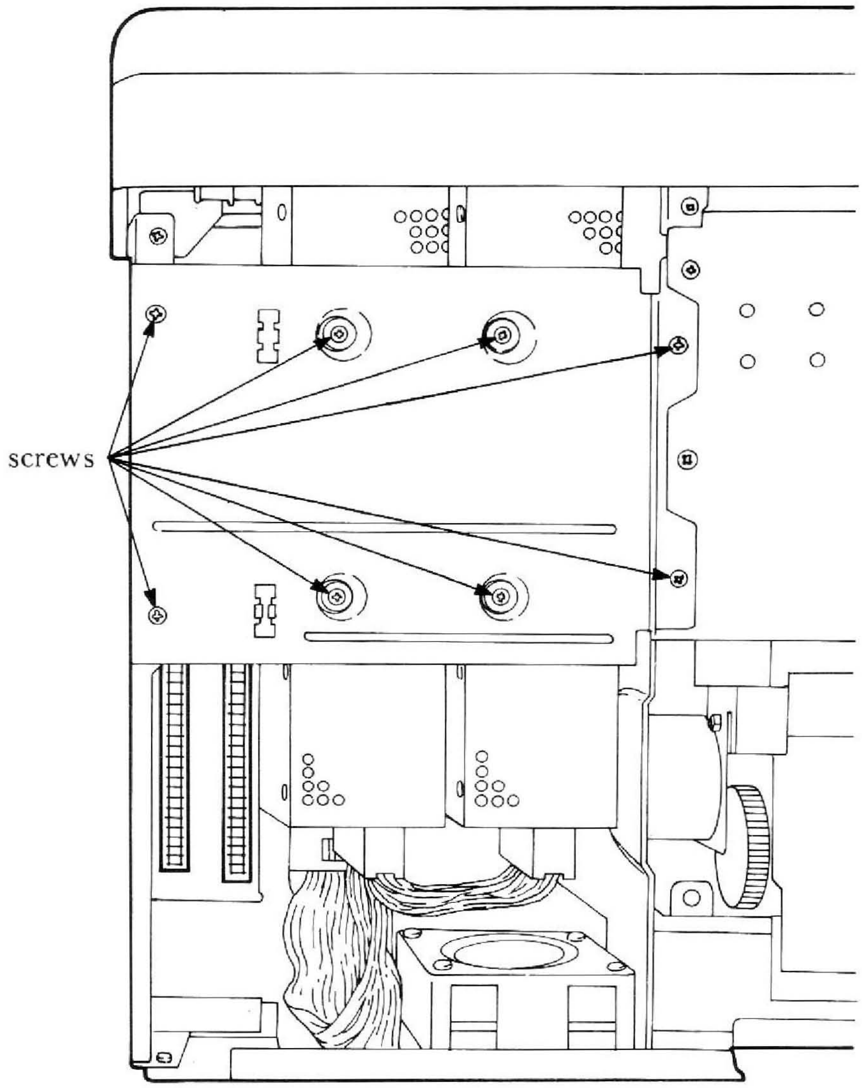


Fig. 3-45. TOP VIEW OF DISK DRIVE AREA

SET UP

STEP 2 You will insert an optional board into one of two connectors. We recommend you insert it into the connector near the disk drive because another connector (near the side panel) has no slot for an external cable connector at the edge of an optional board.

STEP 3 Hold the optional board so that the component side is to the left and the soldered side to the right.

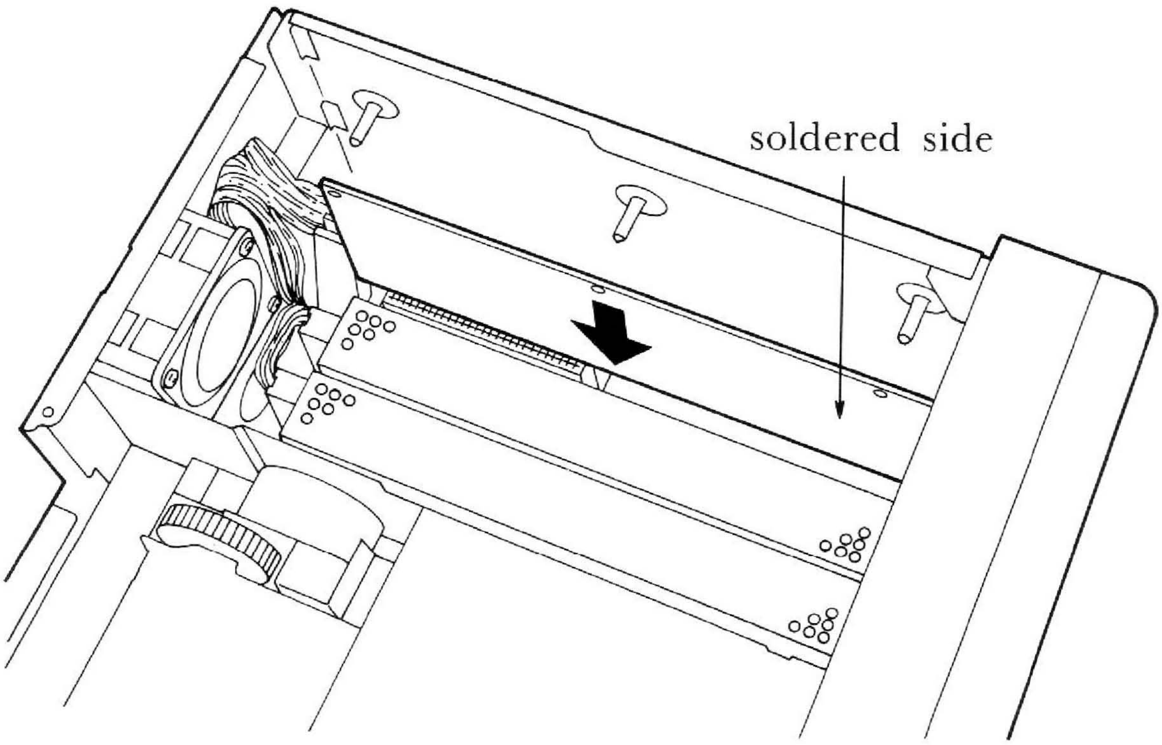


Fig. 3-46. CORRECT ORIENTATION OF OPTIONAL BOARD

STEP 4 Insert the board in the connector. Gently slide in board and push it into the connection on the floor. If the board has an connector for an external cable, you have to remove the protector of the optional board slot by unscrewing.

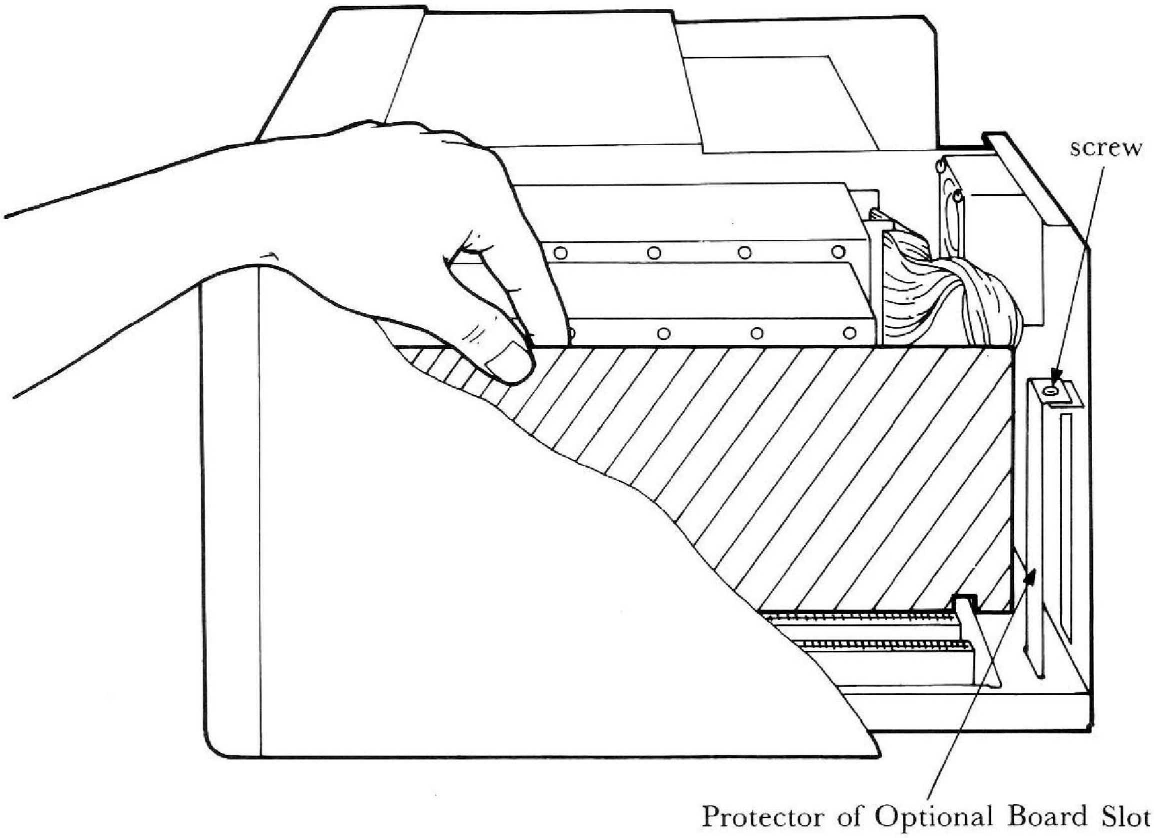


Fig. 3-47. INSERTING BOARD

STEP 5 Replace the screws which hold the disk drive and disk drive top plate in place.

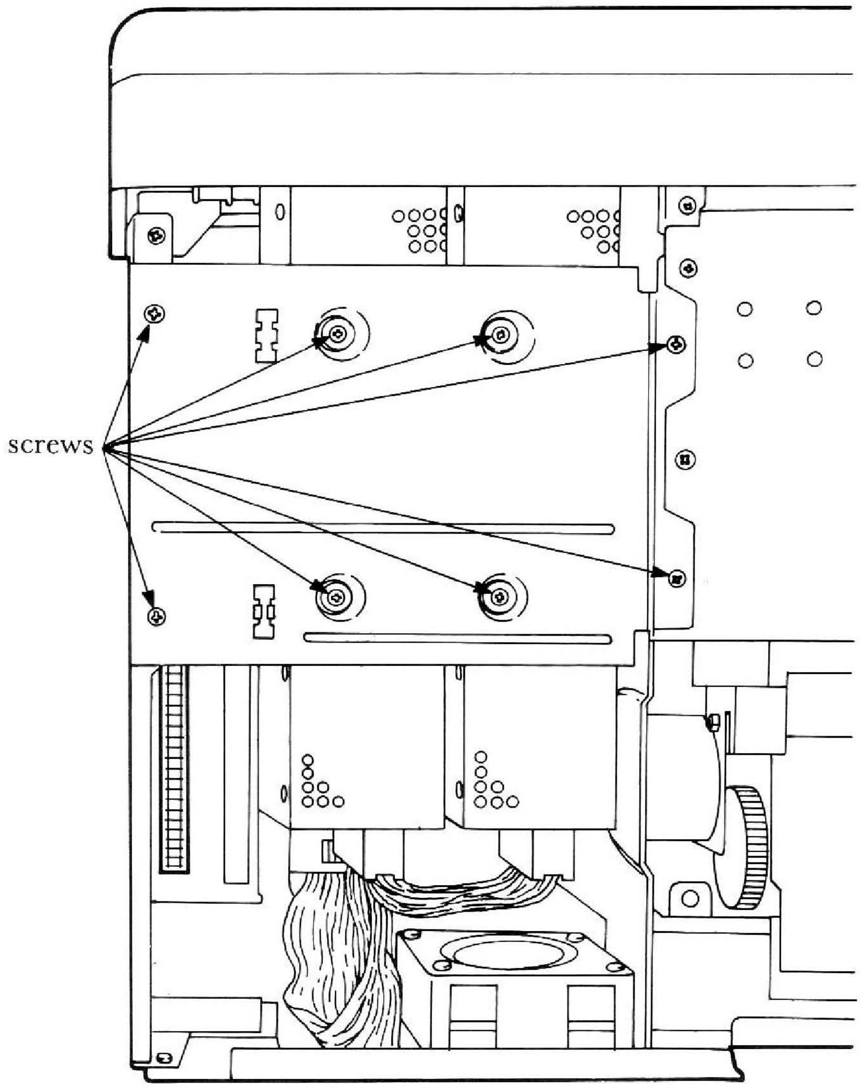


Fig. 3-48. DISK DRIVE COVER PLATE

Replacing the Outside Case of the System Unit

STEP 1 Replace the outer case. Slide the case down until it is firmly seated.

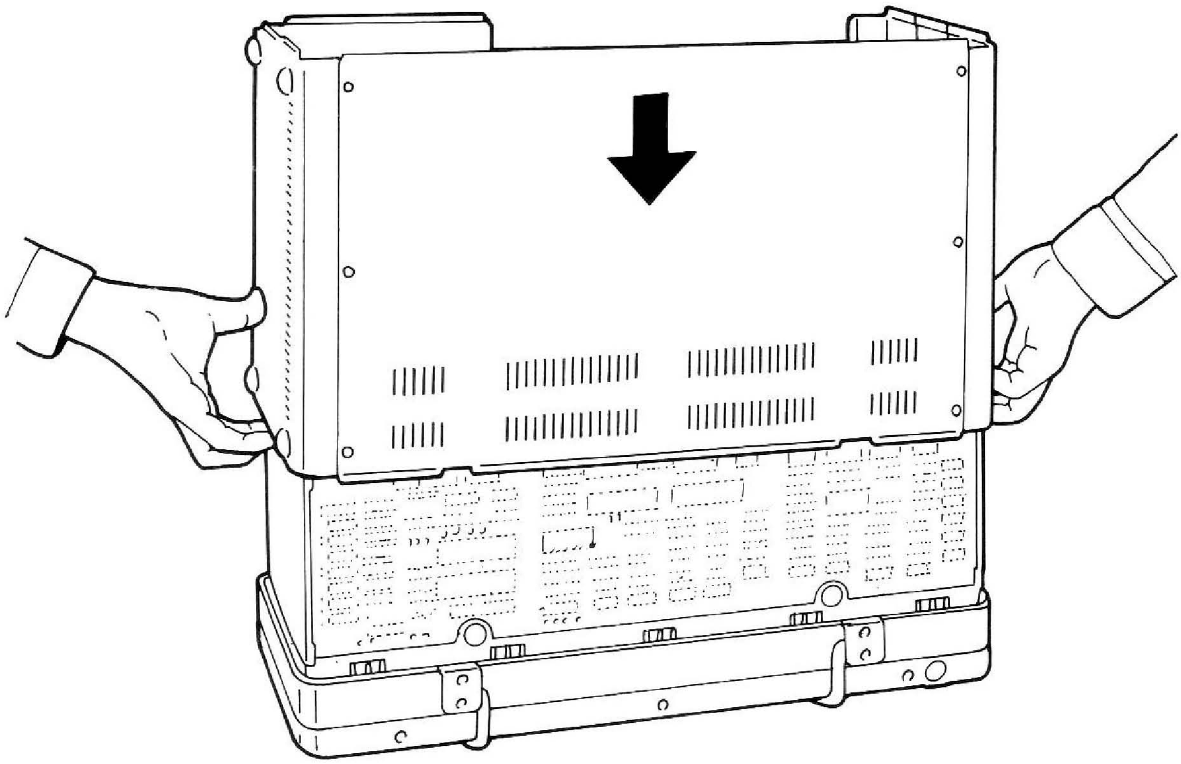


Fig. 3-49. SLIDING OUTER CASE BACK ON

SET UP

STEP 2 Reattach the back panel. You will replace six screws.

*** **WARNING** ***

Before reattaching the back panel, stand holders of the parallel port as shown.

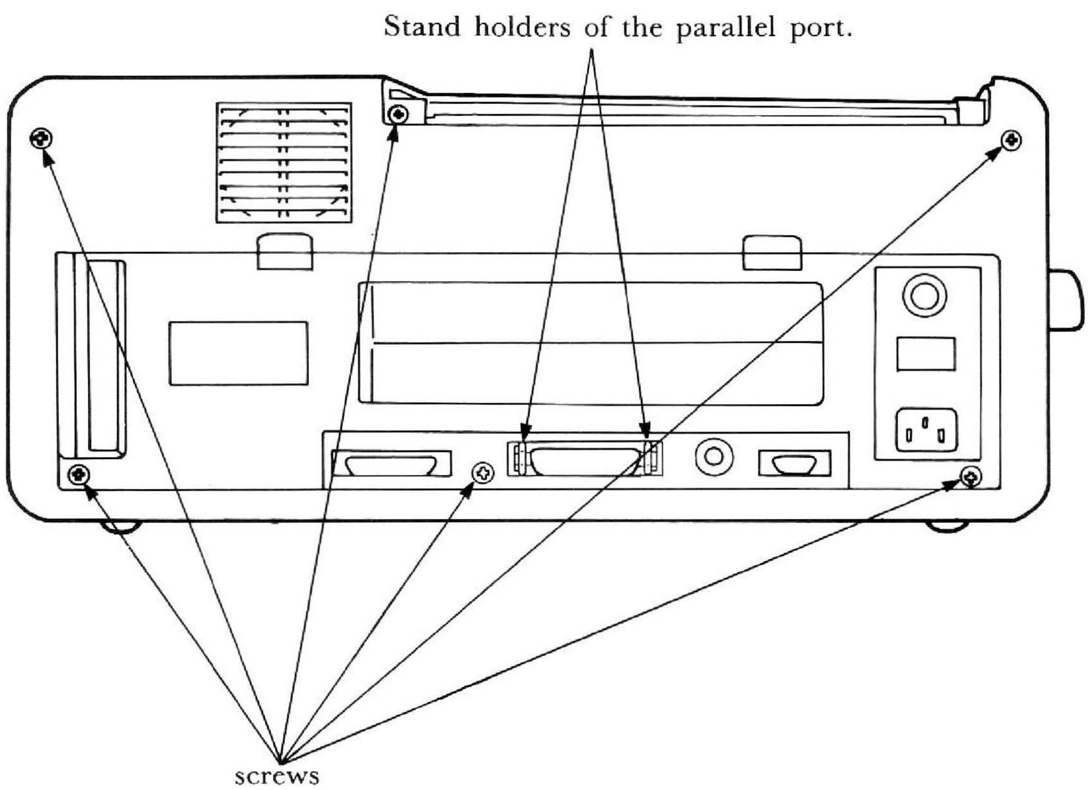


Fig. 3-50. REPLACING BACK PANEL

STEP 3 Reconnect the printer cable to the printer cover.

SET UP

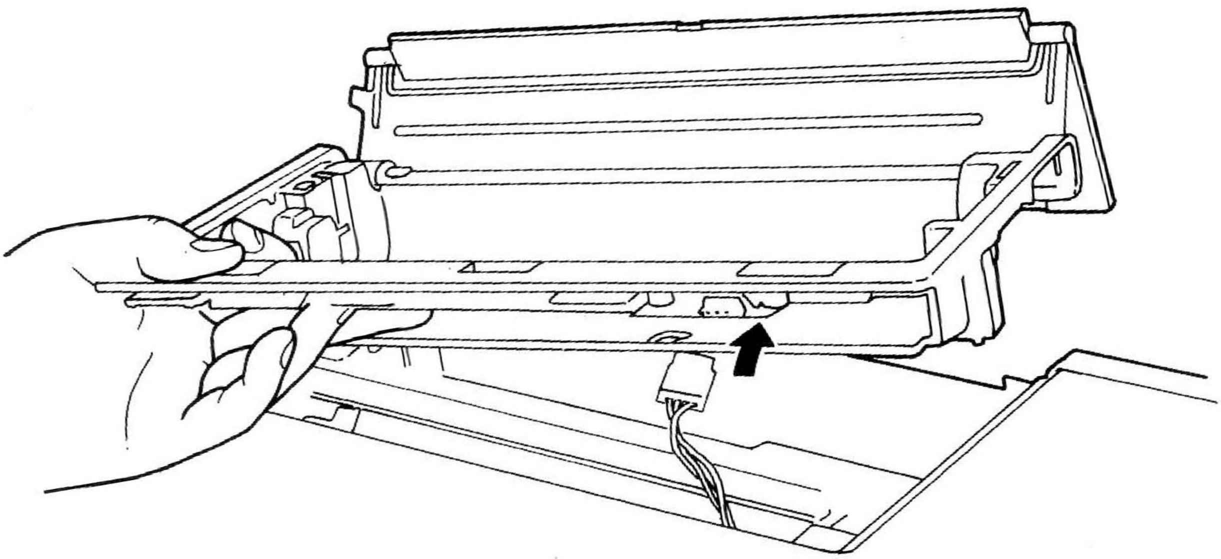


Fig. 3-51. PRINTER PCB BOARD.

STEP 4 Replace the printer cover. Reinsert and replace four screws holding the printer cover in place.

*** **WARNING** ***

Before replacing the printer cover, be sure the paper release knob is in open position.

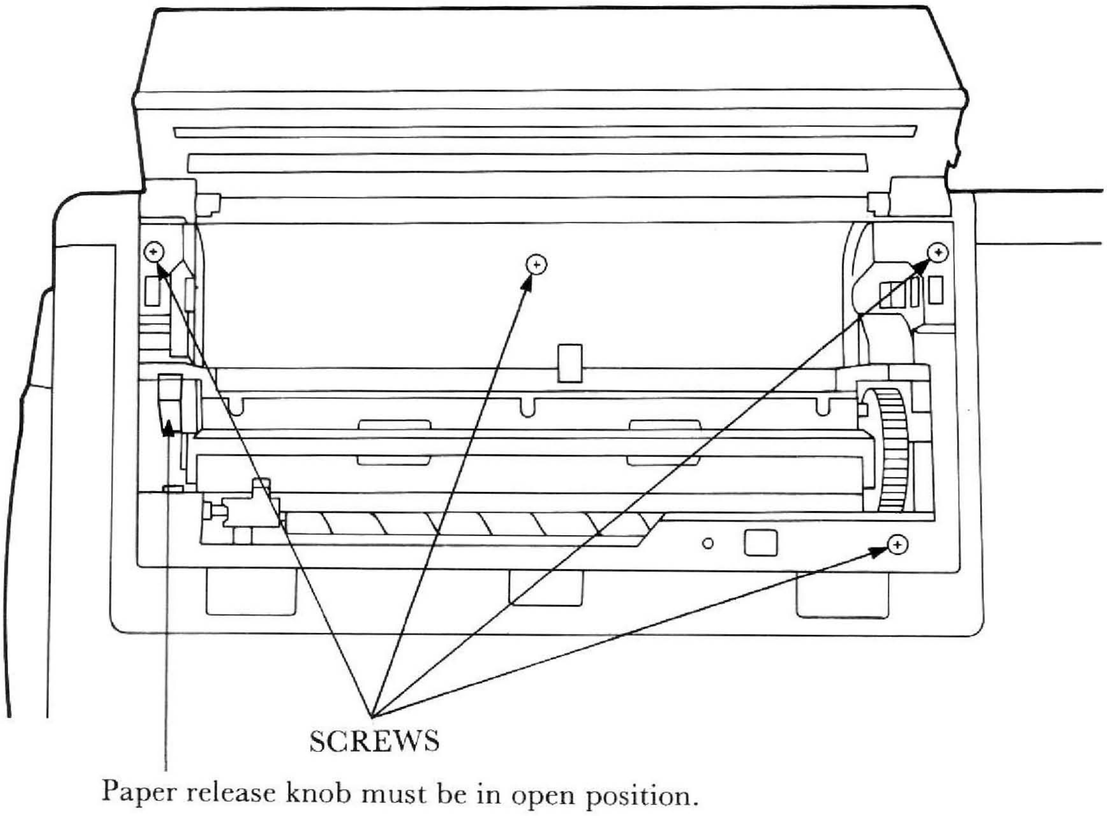


Fig. 3-52. REPLACING PRINTER COVER.

Your upgraded Sr. Partner is now ready to use! Simply reattach the keyboard, plug in the power cord to the rear of the unit and the wall outlet. Switch On the Main Power Switch to begin computing.

CHAPTER 4
OPERATING THE SYSTEM

OPERATION

OPERATION

KEYBOARD USAGE

Many of the keys of your Sr. Partner keyboard are familiar to you, if you often use a typewriter. Beware, however! Although some keys appear to be identical to typewriter keys, in some cases their function is different. The following section details keyboard usage in general. Study the keyboard now to familiarize yourself with it's operation.

> **NOTE:** Specific programs can affect the operation of the keyboard. Check your Operating System and Application Program manuals for specific keyboard functions.

Alphanumeric Keys

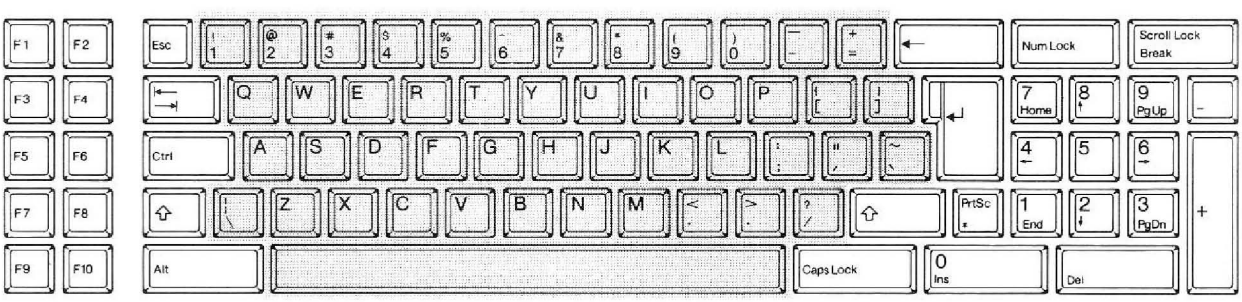
A–Z

Alphabet keys. These keys operate in the same manner as typewriter keys. Normally, these keys enter letters in the lower case. Use <SHIFT> (indicated by vertical arrows) to enter uppercase letters.

1–0

Number keys. These keys operate in the same manner as typewriter keys. Use <SHIFT> keys to enter the symbols above the numbers.

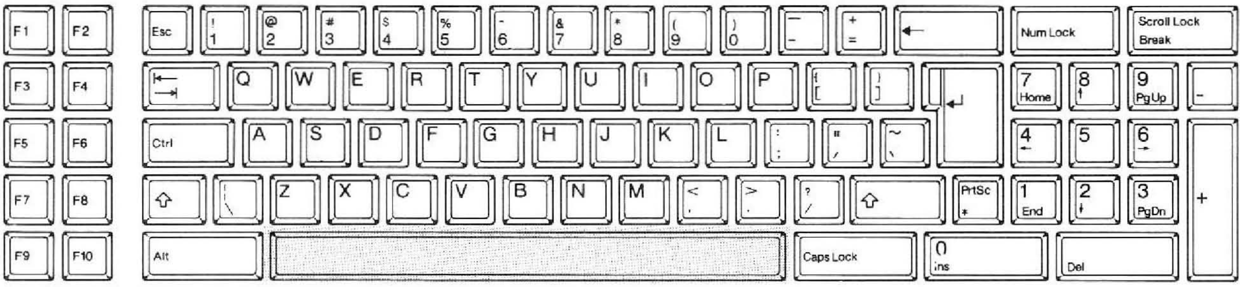
Special Character keys. These keys are used for punctuation and mathematical representation. Use <SHIFT> to enter the symbols on the upper half of the keys.



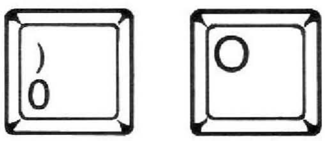
Fgi. 4-1. ALPHANUMERIC KEYS

Keys That Can Confuse You

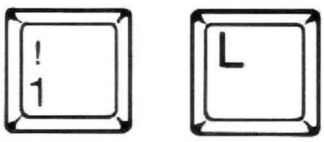
The Spacebar. Press the spacebar to move the cursor to the right.



NOTE: This key DOES NOT OPERATE like a spacebar on a typewriter. On a typewriter you use the spacebar to get to a specific location on the page. Nothing is altered on your typewritten page. The spacebar on your Sr. Partner also moves your location, but to the computer you have entered a new character, a blank character (invisible to you). Using the spacebar is like typing over a word. IT ERASES THE CHARACTERS AS IT MOVES, AND REPLACES THEM WITH BLANK CHARACTERS.



The Letter O and the Number 0. Although these keys are interchangeable on the typewriter, they are different characters to your computer. You can differentiate these two keys because the 0 has a diagonal line running through it.



The Lowercase L and the Number 1. Like the O and 0, these keys mean quite different things to your computer. Be sure you enter the correct key.

Control Keys

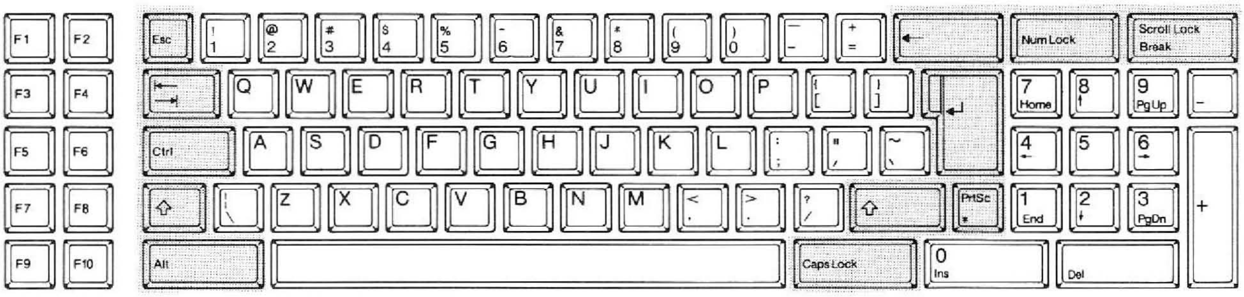
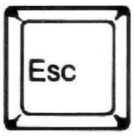


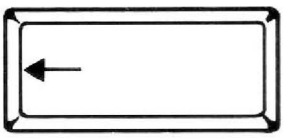
Fig. 4-2. KEYBOARD

OPERATION

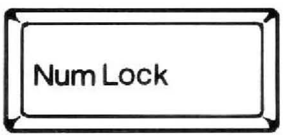
The control keys are arranged around the typewriter area. Keys are explained beginning with the top row.



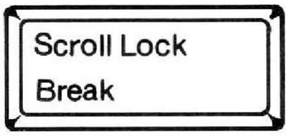
The Escape key. Like many control keys, the functions of <Esc> changes according to the program being run. See your Operating System or Application Program manual for the specific function of this key.



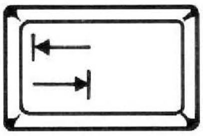
The Backspace key. This key erases the character the cursor is right and moves the cursor one space to the left. Use this key for making corrections. ?



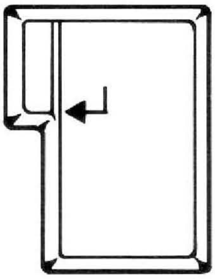
The Number Lock key. Use this key to lock the numeric keypad into the numeric mode. Pressing <Num Lock> again returns the numeric keypad to the cursor control mode.



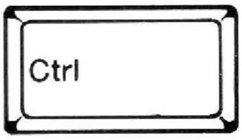
The Scroll Lock/Break key. Like the <Esc> key, the specific function of <Scroll Lock/Break> is defined in your Operating System or Application Program manual. This key will be denoted by <Break> in this manual.



The Tab key. Use this key to move forward or backward to the next tab stop. To set tab, see your Operating System or Application Program manual.

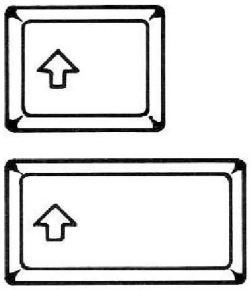


The ENTER key. THIS IS THE MOST IMPORTANT KEY ON YOUR KEYBOARD. In DOS and BASIC, this key signals the computer that you are finished entering data and are waiting for the computer to perform some action or give some response. Refer to each specific manual for an explanation of the <ENTER> function in the program you are using.

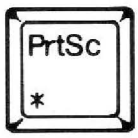


The Control key. <Ctrl> is always used in conjunction with other keys to perform specific operations. The function of this key varies according to the program definitions. See your Operating System or Application Program manual for details.

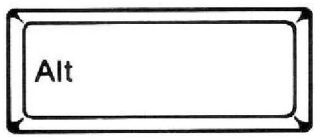
NOTE: When you see the notation <Ctrl> <S>, for example, you must press <Ctrl> and hold it down while pressing <S> key.



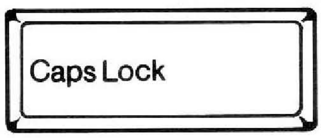
The SHIFT keys. There are two <SHIFT> keys on your keyboard. To enter uppercase letters and the symbols on the upper half of the number and punctuation keys, press <SHIFT> and while holding it down press the character desired. See also <Caps Lock> below.



The Print Screen key. Use <PrtSc> in conjunction with <SHIFT> to print the information currently displayed on the screen. Pressing <PrtSc> without <SHIFT> enters the asterisk character.



The Alternate key. For information on this key, see your Operating System or Application Program manual.



The Caps Lock key. <Caps Lock> lets you decide which case you would prefer as the normal entry mode. The first time you press <Caps Lock> you lock in the uppercase mode. You get lowercase letters by pressing <SHIFT>. The second time you press <Caps Lock> you lock into the lowercase mode. Pressing <SHIFT> then enters uppercase letters.

NOTE: <Caps Lock> only affects the letter keys. No matter what the mode of the <Caps Lock> key, the symbols above the numbers and the upper symbols on the punctuation keys require the use of <SHIFT>.

Numeric Keypad

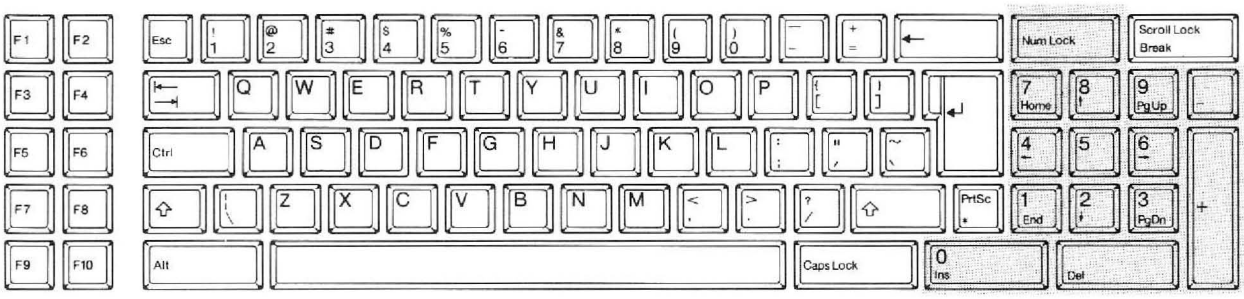
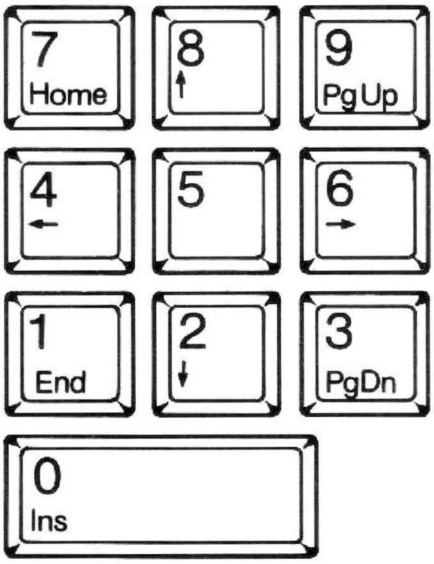
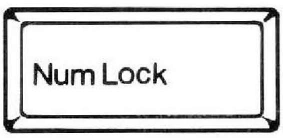


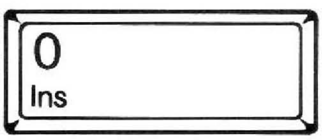
Fig. 4-3. KEYBOARD



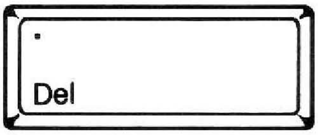
The Number keys. Usc these keys to enter numeric data.



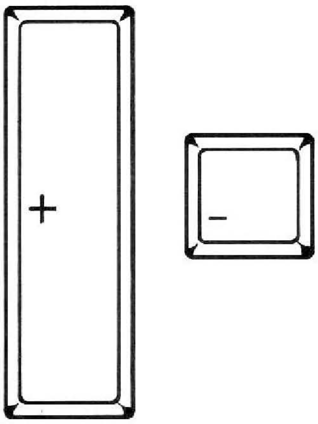
The Number Lock key. This is the control key used in conjunction with the numeric keypad. Press <Num Lock> once and you are locked in the numeric mode. Press <Num Lock> again and you toggle back to the cursor control mode.



The Insert key. Use <Ins> to insert characters in the middle of a line. When you use <Ins> characters are inserted BEFORE THE UNDERLINED CHARACTER. Press <Ins> again to return to normal operation.



The Delete key. Use <Del> to remove the character underlined by the cursor. When you use <Del> the current character and its space are deleted. All remaining characters to the right of the deleted position move one space to the left.

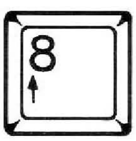


The Plus and Minus keys. These keys have no programmed function. They do display a + or − on the screen.

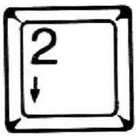
Cursor Control

The cursor is a small blinking underline on the display which tells you your position on the screen. The cursor indicates the location of the next character you enter. When you begin to enter data, the cursor moves ahead of your typing and indicates the next available space. The cursor is also used in editing information. When using <Ins> the cursor indicates the space BEFORE which the character will appear. When using <Del> the cursor underlines the character to be deleted.

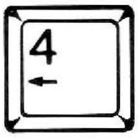
> **NOTE:** Remember <Num Lock> determines the current mode of the numeric keypad. If you are locked in the numeric mode, you must press <Num Lock> before using these keys for cursor control.



The Cursor Up key. Moves the cursor one line up.

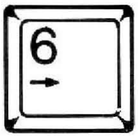


The Cursor Down key. Moves the cursor one line down.



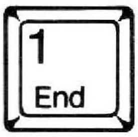
The Cursor Left key. Moves the cursor one position to the left. Some Application Programs delete the character.

OPERATION



The Cursor Right key. Moves the cursor one position to the right.

The Cursor Home key. Moves the cursor to the top left corner of the screen.



The Cursor End key. Moves the cursor to the bottom left of the screen.

The Page Up key. <PgUp> is a program controlled key, that is it may or may not be operable in the program you are using (DOS does not support <PgUp>). When supported, <PgUp> allows you to "scroll" backwards in your file, looking at one "page" at a time. Check the specific manual for the program being executed for details on <PgUp> Operation.

The Page Down key. Like <PgUp>, <PgDn> is a program controlled key. When supported, <PgDn> allows you to "scroll" forward in your file, one "page" at a time. Check your Application Program manual for further details on <PgDn> operation.

Some of these keys are available in the specific programs. See your Operating System or Application Program manual.

Function Keys

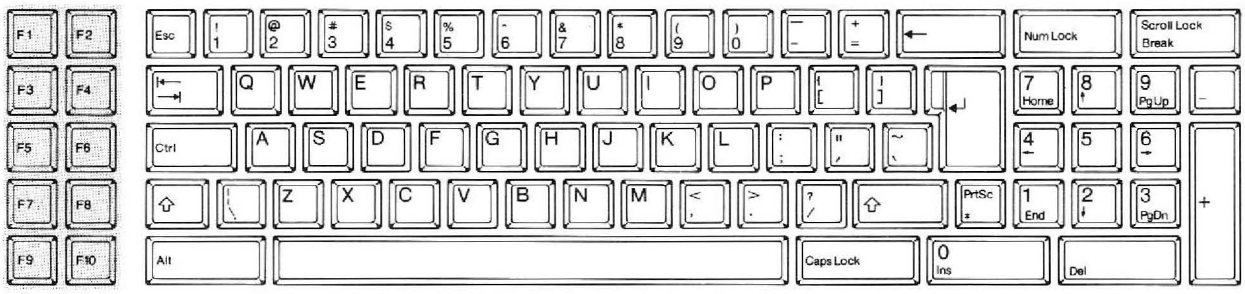


Fig. 4-4. KEYBOARD

The function keys are always under program control. For details on function key operations see your Operating System or Application Program manuals.

USING THE DISK DRIVE

Disks

Disks are small coated and used to read in programs and data, and write out information you wish to save from RAM. Because disks provide the instructions to the computer and supply the necessary data for programs, they are a very vital link in your Sr. Partner system.

OPERATION

*** **WARNING** ***

BEFORE HANDLING OR USING ANY DISK SEE THE SECTION BELOW ON DISK CARE

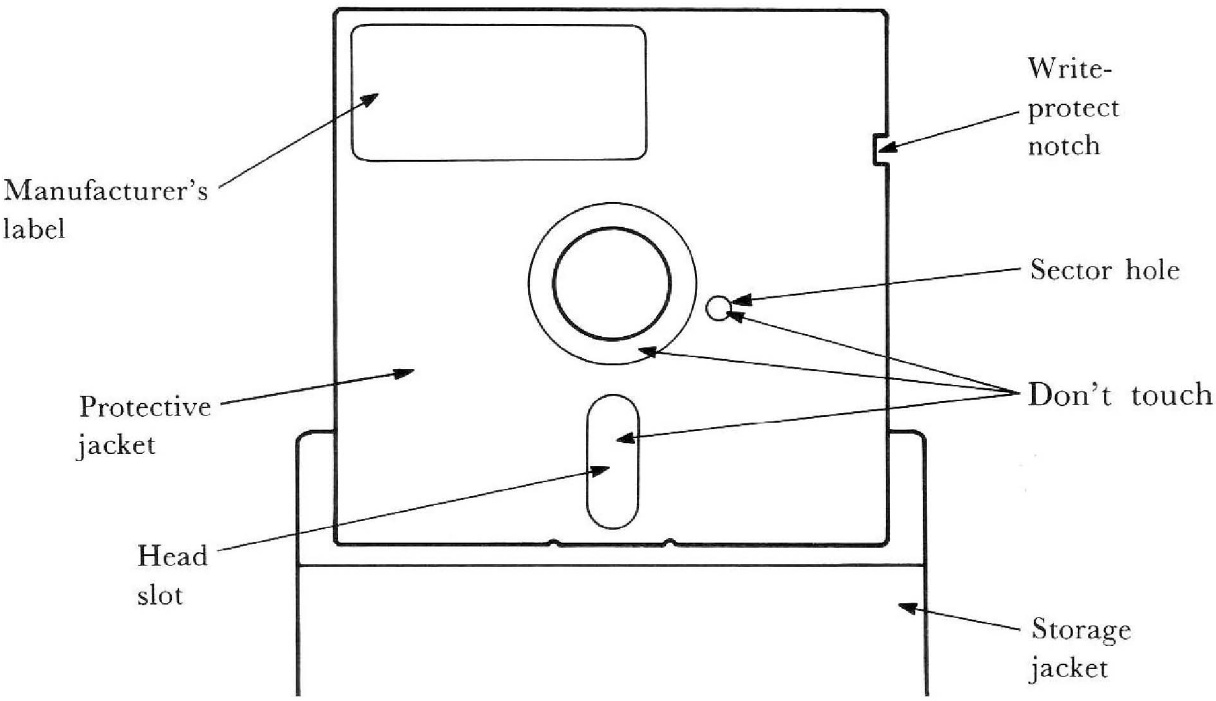


Fig. 4-5. DISK

Disks, or "floppy disks" come in their own **protective jacket**. Do not attempt to remove this permanent covering, it is designed to protect the delicate coating on the disk.

There are three exposed areas on the disk. YOU MUST NEVER TOUCH THESE AREAS AS VALUABLE DATA WILL BE ALTERED OR DESTROYED.

Around the center hole is a small strip of exposed recording surface. Located to the side is a small **sector hole**. These areas help to align the disk when it is inserted into the drive.

The **head slot** provides the areas which is "read" by the recording head in the drive unit. The drive head, by passing very close to the surface of the disk as it rotates, decodes the particles imbedded in the coating on the disk. In the same way the head "lays down" new information when you write to a disk.

The **write-protect notch** prevents you from accidentally losing data by recording over existing information. When this notch is covered, you cannot write to a disk. An adhesive tab for covering the write protect notch is supplied with purchased disks. The System Disk attached the Sr. Partner and some purchased programs are write protected when you receive them.

The **manufacturer's label** identifies the type of disk and the maker.

The **storage jacket** provides protection for the disk when you are not using it.

Purchasing Disks for Your Sr. Partner

Your Sr. Partner uses 5¼ inch double-sided, double-density, soft-sectored disks. This type of disk can store approximately 360,000 characters (also termed 360K).

Double-sided disks can store information on both sides. This does not mean that you must turn over your disks, however. Information is automatically stored on both surfaces.

OPERATION

Double-density disks allow data to be packed together in a more economical pattern. They are called double-density because they can store twice the amount of information of the original disks.

How Information is Stored on a Disk

Information is stored on disks in concentric circles called tracks.

Tracks are divided into specific areas called sectors. Information is read or written one sector at a time.

The recording head on your disk drive both reads and writes information to the disk. The stationary head transfers or records data as the disk spins.

The space on a disk is measured in bytes. Each byte holds one character. The disk used by your Sr. Parther can hold approximately 360K bytes.

When information is stored on a disk it is assigned a specific side, track and sector location. This information helps the computer locate specific data or identify unused locations.
You do not need to know the physical location of your files on a disk. To access a file, you simply type in the file's name, the operating system keeps track of its location.

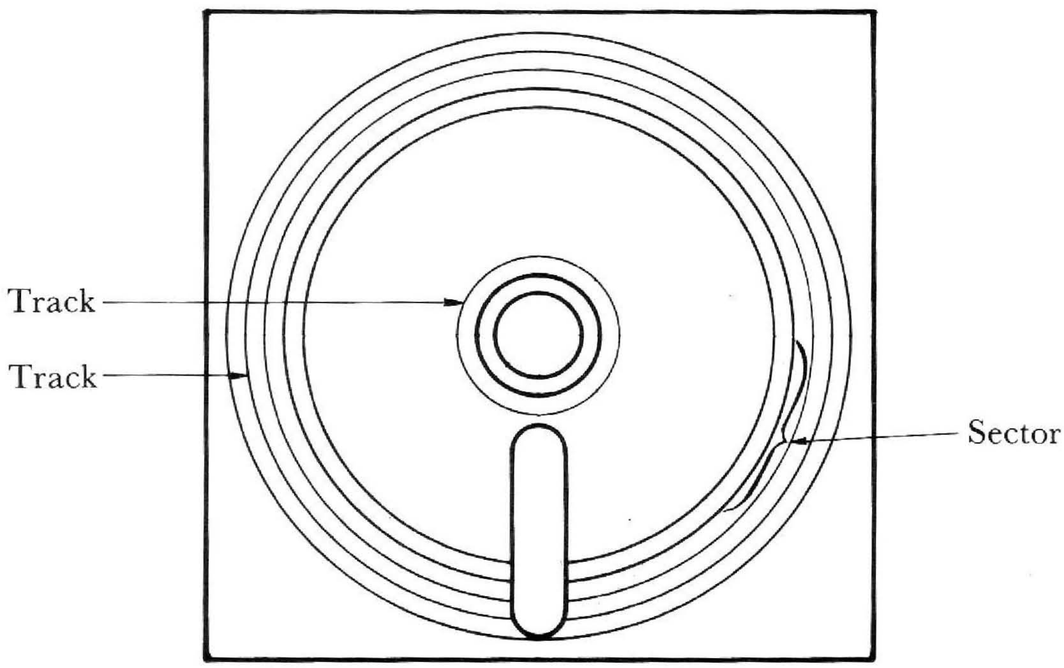


Fig. 4-6. TRACKS AND SECTORS WITHIN SECTORS

Caring For Disks

Your disks are probably the most fragile component in your computer system. Any damage to the delicate surface of the disk can result in lost or damaged information. ALWAYS HANDLE YOUR DISKS WITH CARE!

The recording head of the disk drive must have a clean surface to read and write information accurately. Never touch the exposed surfaces of the disk (around center hole, sector hole and the recording head slot).

Fingerprints, hair, dirt, tobacco, food and any type of scratches can cause you to lose valuable data. And imperfections on the disk can damage the recording head as well.

Even the smallest impressions on the surface of the disk can alter data. Keep pencils, ballpoint pens, coffee cups and paper clips away from your disks. WHEN LABELLING DISKS, USE ONLY SOFT, FELT-TIP PENS AND WRITE ONLY ON THE LABEL SURFACE.

Since the surface of the disk is covered with a magnetic coating, other magnetic fields can wreak havoc with stored information. Keep disks away from magnetic field sources such as telephones, dictation equipment, x-rays and calculators.

Avoid extremes of heat and humidity. Keep disks out of direct sunlight. Sunlight coming in through a window can heat up disks very quickly!

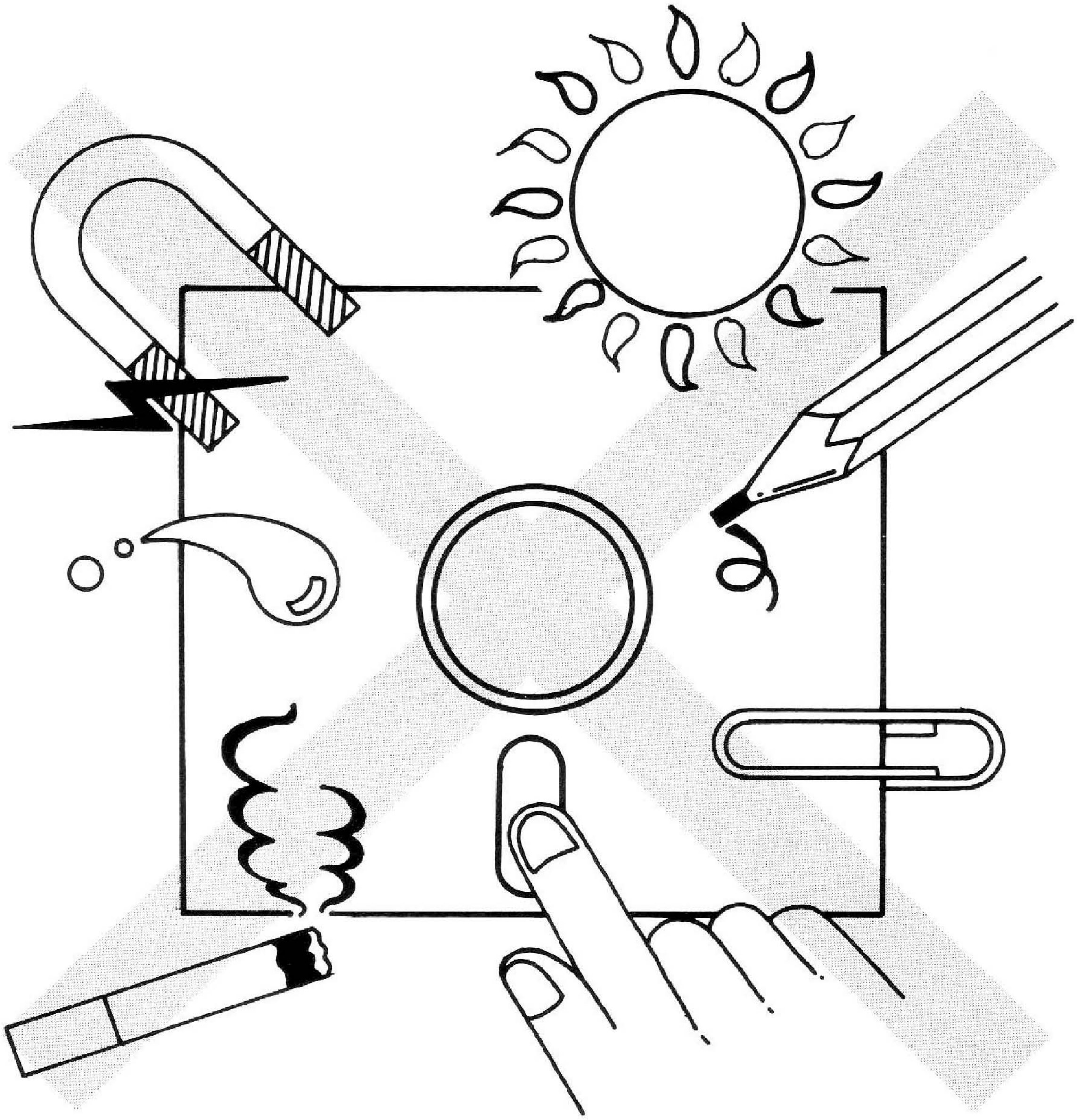


Fig. 4-7. CARING FOR DISKS

OPERATION

Storing Disks

You should never leave disks lying around. They may become lost in a pile of papers, accidentally bent or damaged or have something heavy put down on top of them. WHEN NOT IN USE DISKS SHOULD BE STORED IN THEIR PROTECTIVE JACKET AND PLACED IN A SAFE LOCATION.

Store infrequently used disks in storage containers designed for this purpose. Related groups of disks can be easily located using this type of storage.

When stored disks are standing on edge, be sure they are not bent or sagging.

Keep stored disks in a dry, cool place. Avoid extremes of heat and humidity, especially sunlight streaming in through an office or car window!

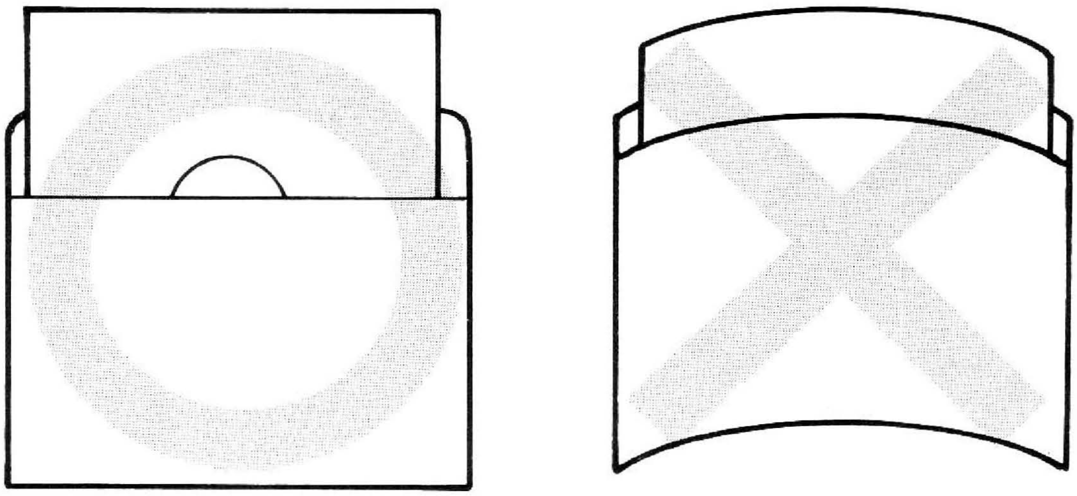


Fig. 4-8. CORRECT STORAGE TECHNIQUES

The Write-Protect Notch

Usually your computer will both read from and write to a disk. When disks contain important information which you do not want to change, you must write protect the disks.

Some disks (your System Disk and purchased application program disks) cannot be altered. They are permanently write protected. However when you copy these disks, or create personal disks which you do not want changed, you must write protect them.

Purchased disks contain write protect tabs in their packaging. These adhesive tabs are glued over the write protect notch to prevent the recording head from writing information to the disk. These disks are then READ ONLY. Remove the tab to return the disk to both read and write status.

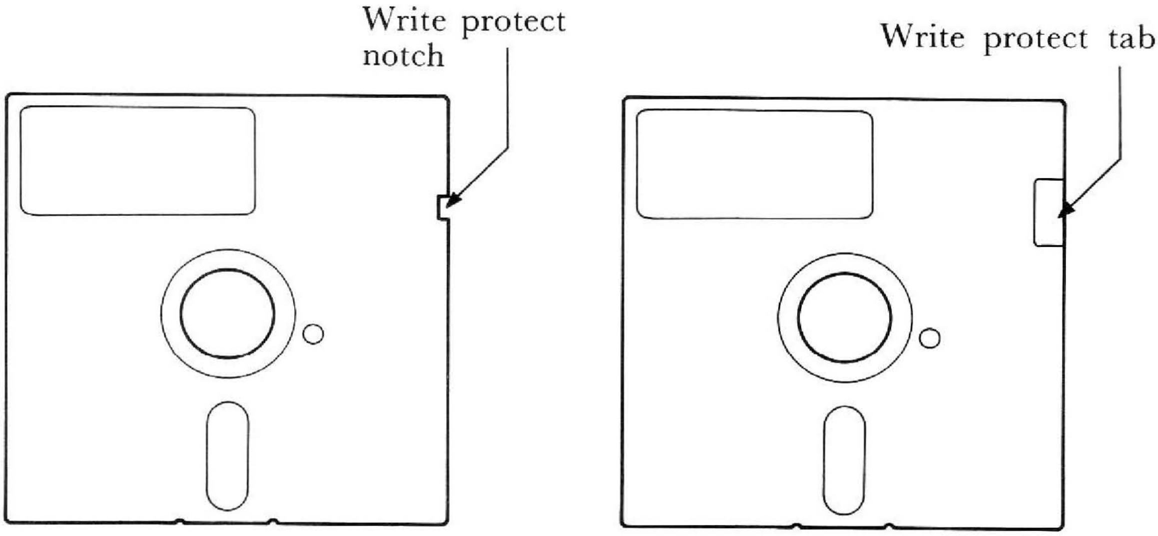


Fig. 4-9. WRITE PROTECT TAB AND WRITE PROTECT NOTCH

BRINGING UP THE SYSTEM

The System Disk

The System Disk contains the operating instruction for your Sr. Partner. It is located in the jacket on the inside rear cover of this manual.

*** **WARNING** ***

Until you make a copy of this disk it is the only method of operating your Sr. Partner. Handle this disk with extreme care.

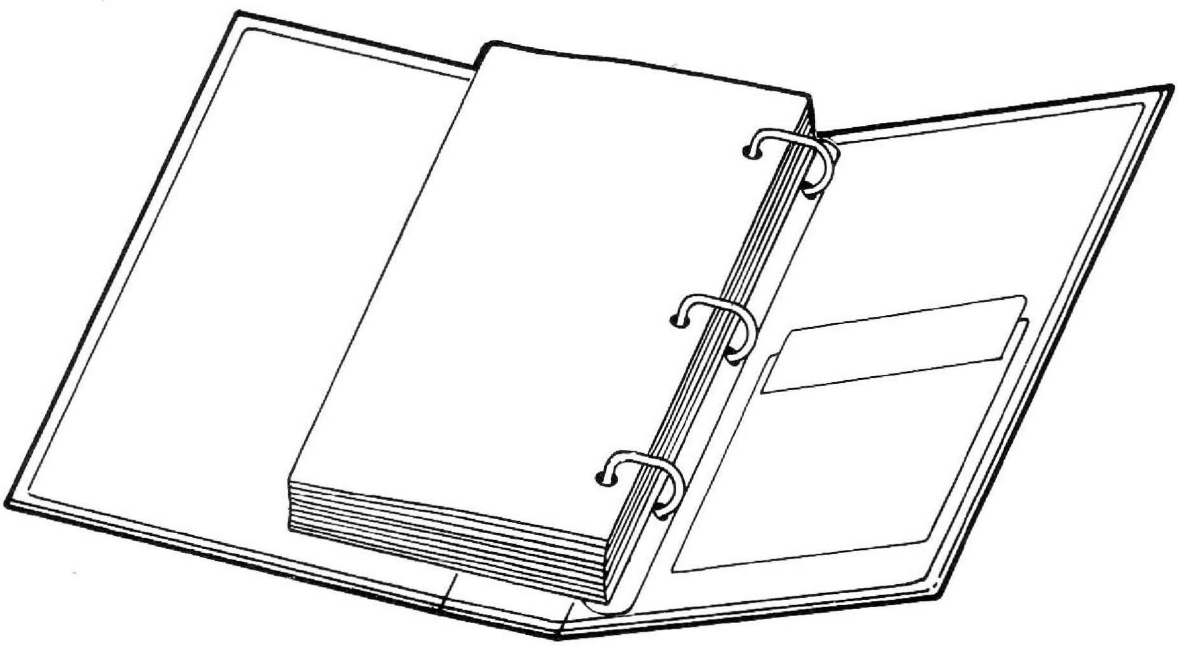


Fig. 4-10. SYSTEM DISK

Inserting Disks

This section contains just inserting and removing a disk. Booting the System Disk is discribed on page 4-30.

STEP 1 Be sure the main power switch is in the OFF position.

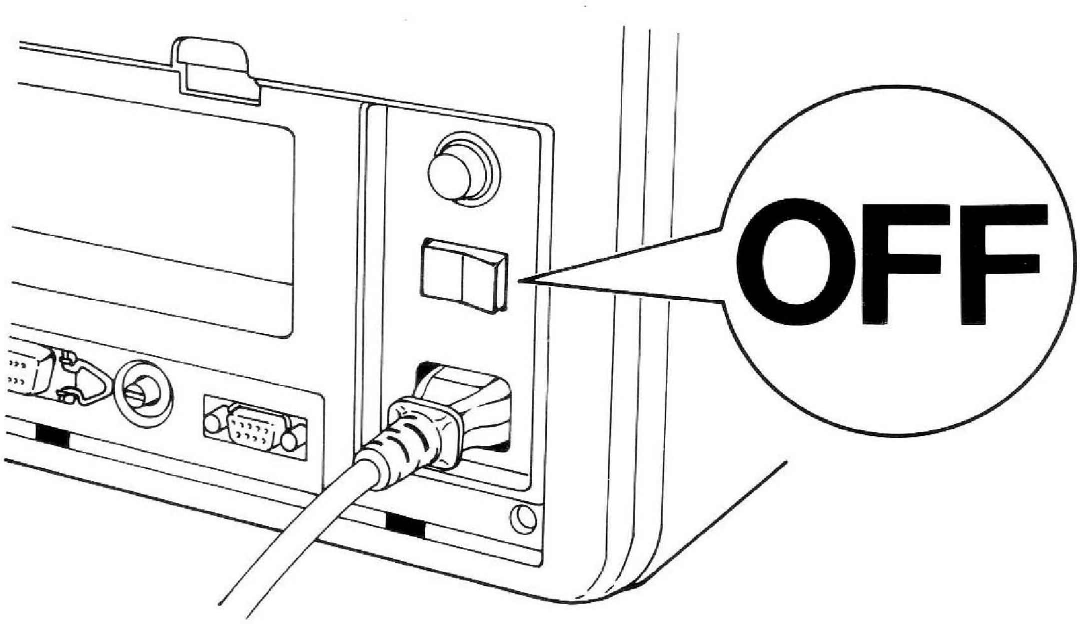


Fig. 4-11. POWER SWITCH

OPERATION

STEP 2 If the door of the disk drive is closed, push the load lever in and to the left to expose the drive slot.

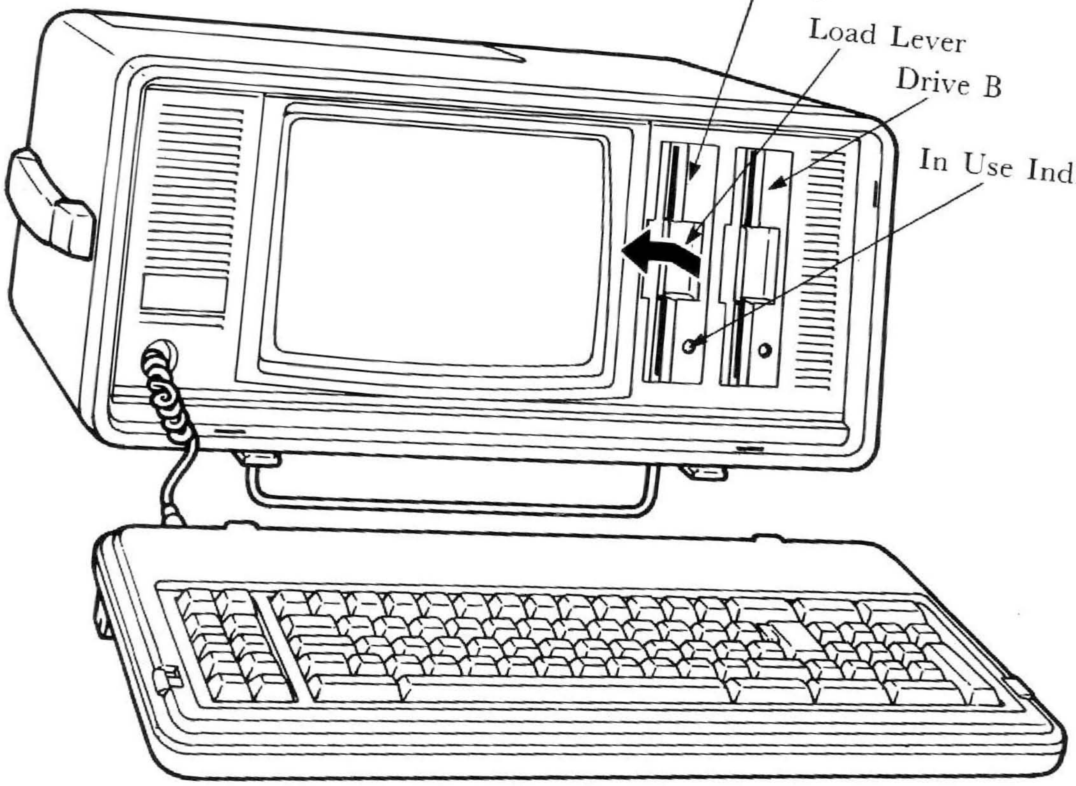


Fig. 4-12. LOAD LEVER

*** **WARNING** ***

Remove the head protection sheet from the drive. This head protection sheet protects the delicate recording head while your Sr. Partner is in shipment.

STEP 3 Hold the disk in your right hand with your thumb on the manufacturer's label. The label should be facing left. The edge with the head slot enters the drive first.

NOTE: You must insert the System Disk in Drive A, the left drive if you have two drive units. Drive A is the primary drive of your Sr. Partner system. YOU MUST BOOT (TURN ON) DOS FROM DRIVE A.

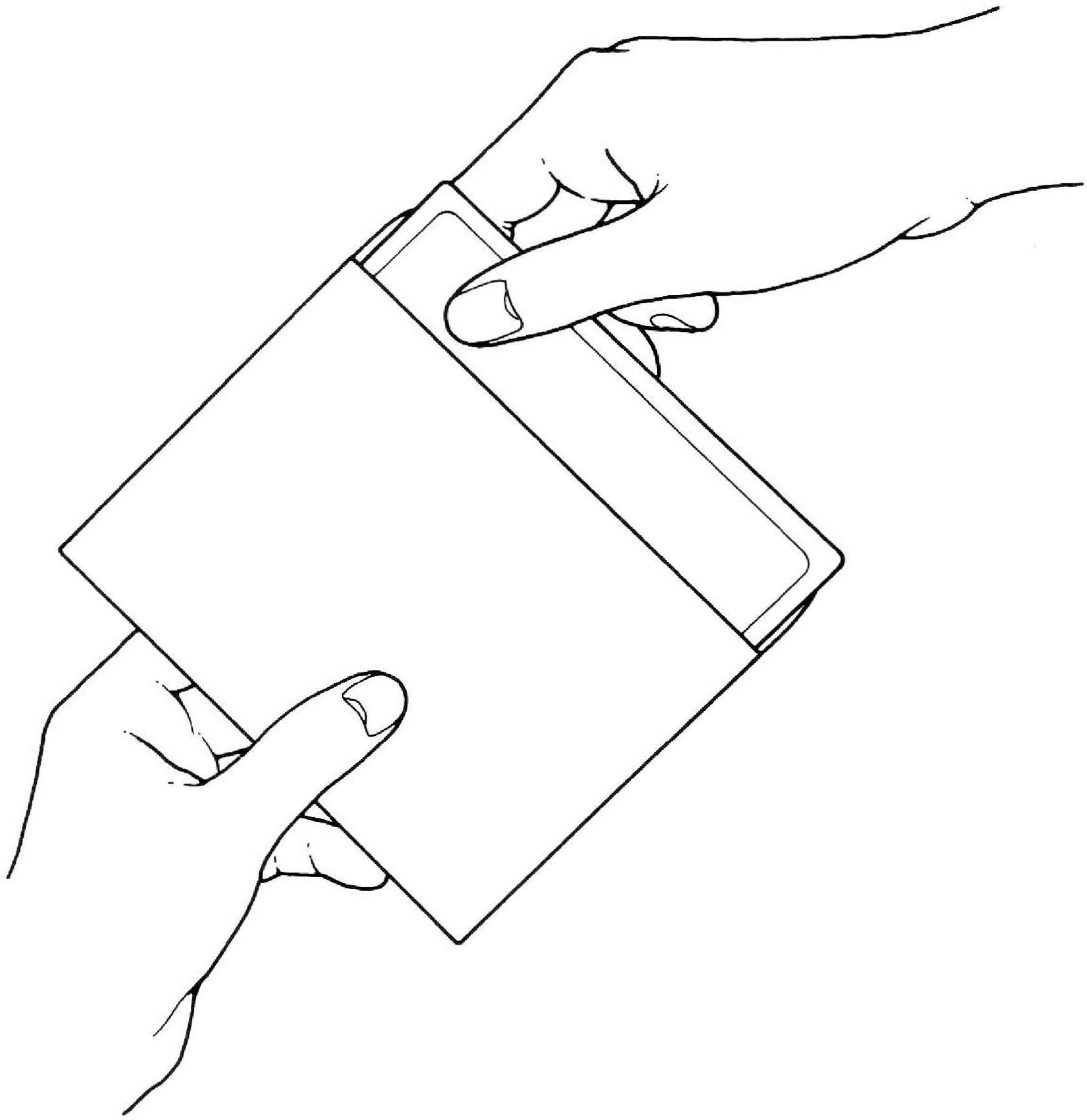


Fig. 4-13. CORRECT HOLDING TECHNIQUE OF DISK

STEP 4 As you insert the disk it will come to rest against a solid stop. DO NOT FORCE THE DISK BEYOND THE STOP. Once the stop is reached the disk is completely inserted.

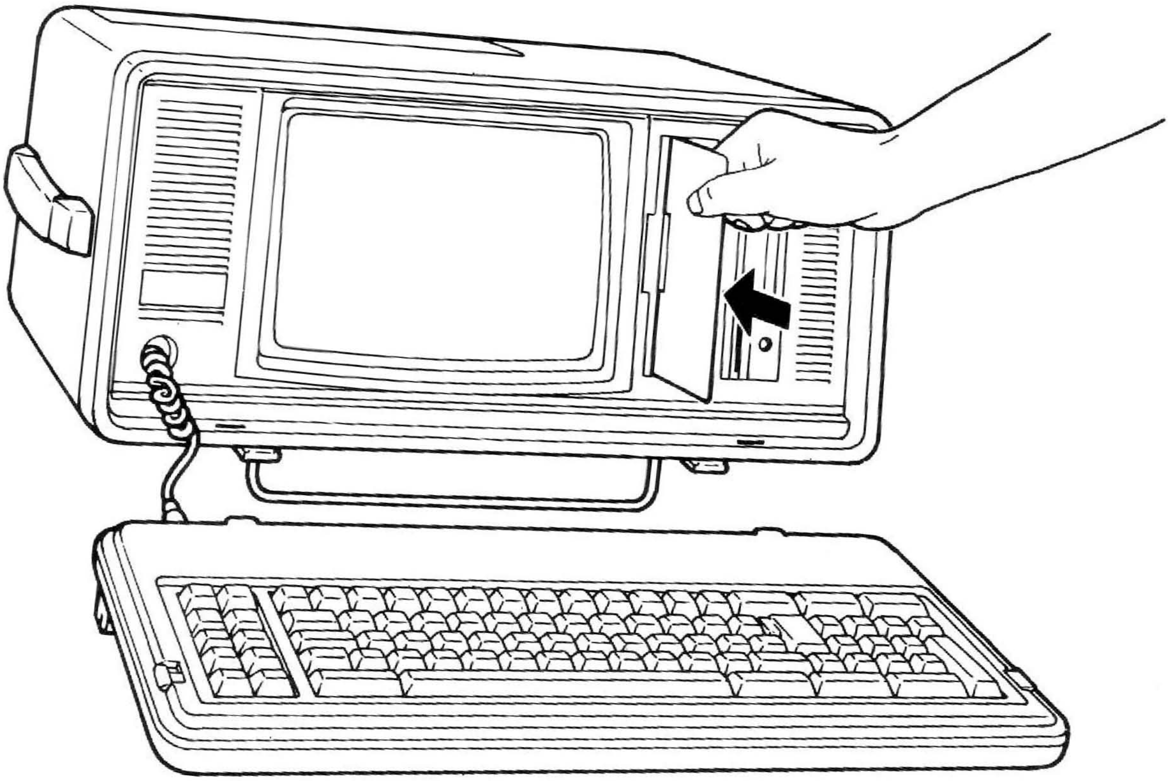


Fig. 4-14. INSERTION OF DISK

STEP 5 The drive cannot operate unless the door is locked. Using your thumb, push the lever to the right to lock the drive.

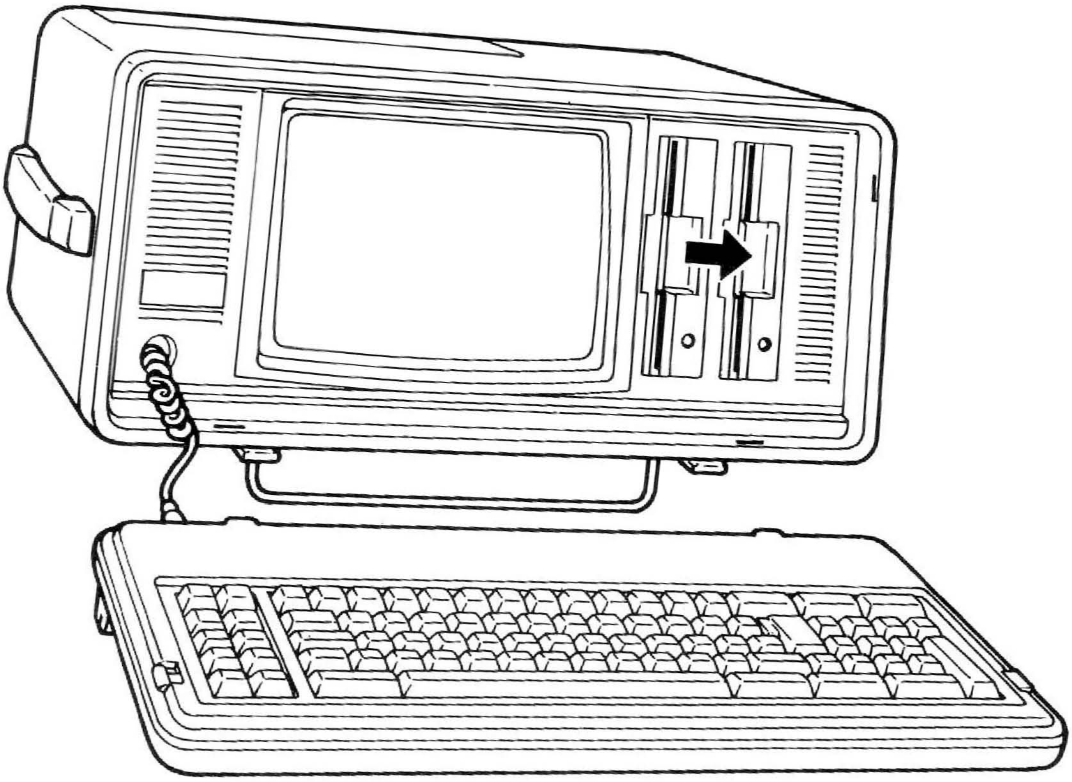


Fig. 4-15. LOCKING DRIVE

Removing Disks

When you are finished using a disk, remove it carefully.

*** **WARNING** ***

Never remove a disk when the drive in use indicator is lit up. You may damage the drive and/or the disk and lose valuable information.

OPERATION

STEP 1 Push the lever in and to the left. The door will open and the disk will come slightly out of the drive.

STEP 2 Grasp the label with your right hand and remove the disk.

STEP 3 Return the disk to its storage jacket.

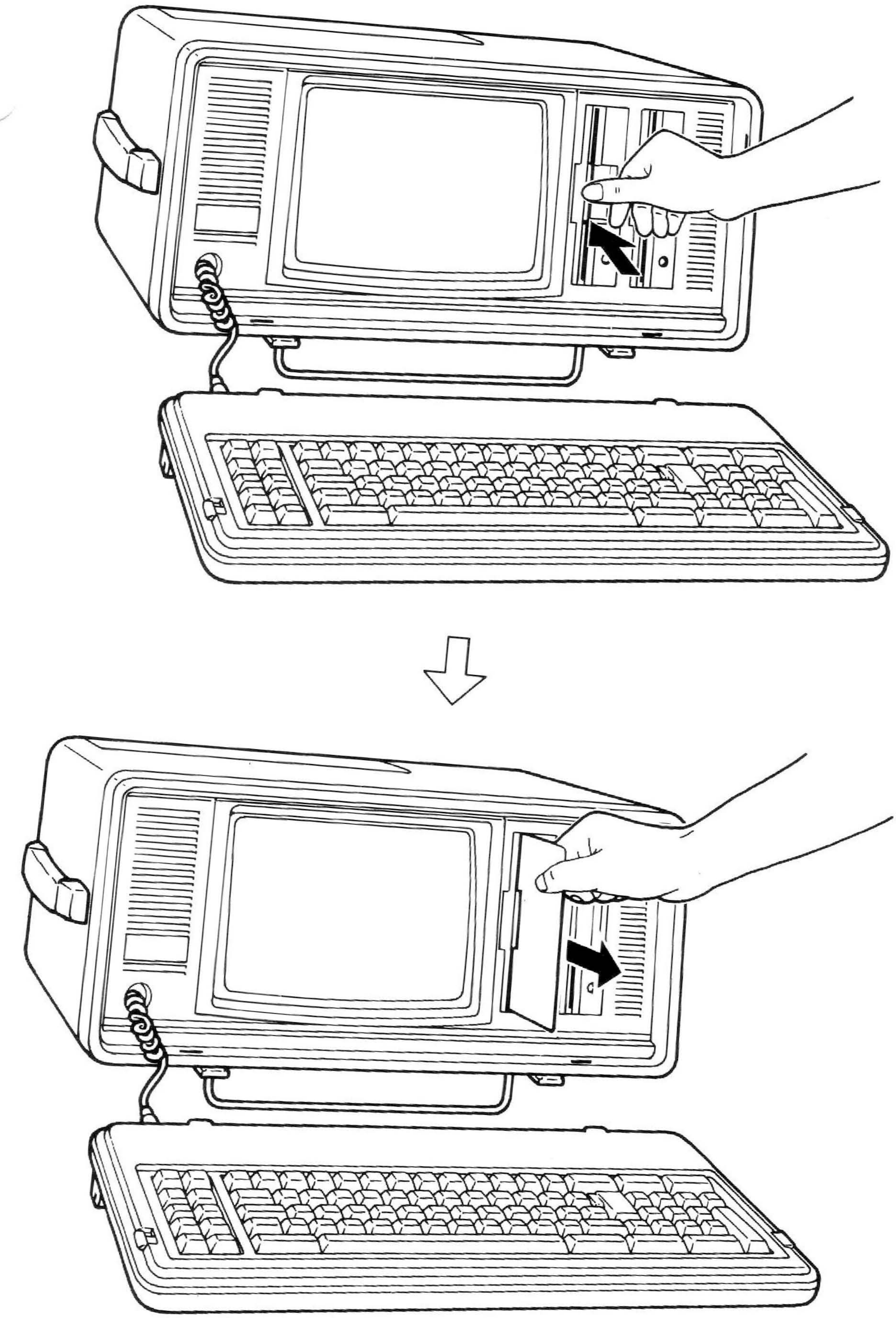


Fig. 4-16. LOCK IN OPEN POSITION

Turning on the Power

*****WARNING*****

You must turn on the main power switch **before inserting the System Disk.**

Be sure the computer is plugged into a wall socket and contains NO **disk** or a **head protection sheet**. Now turn on the main power switch.

OPERATION

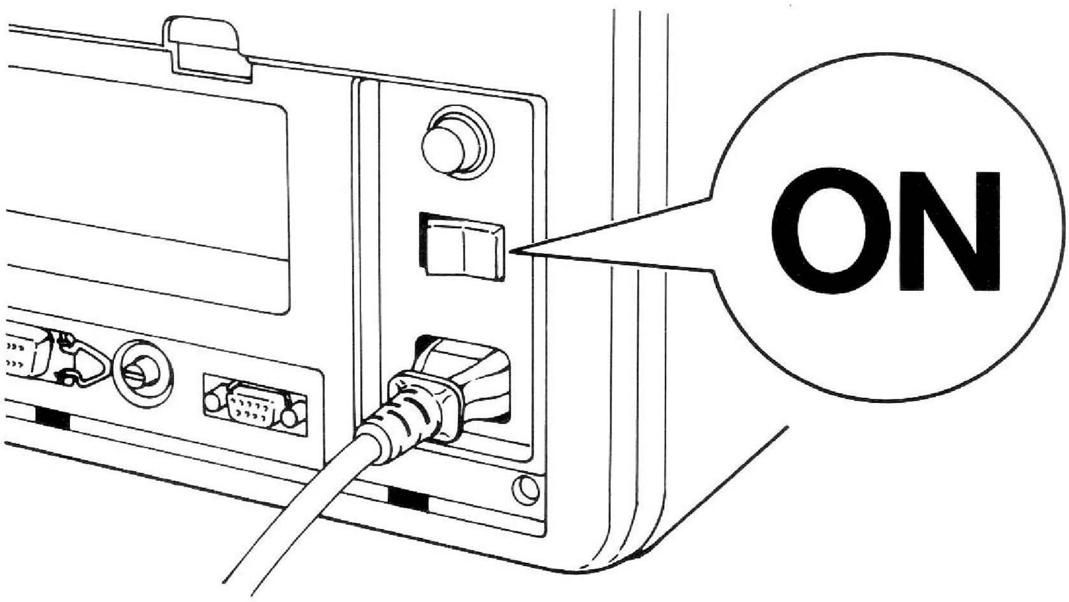


Fig. 4-17. POWER ON

Insert the System Disk into Drive A (It is labeled "A". Refer to Fig. 4-18.). You can insert the System Disk at any time.

1. After one short beep is heard...
2. While the cursor is blinking in the upper left corner of the screen...
3. While the in-use indicator of the disk drive is lit up...
4. When the message Disk boot error Replace and strike any key when ready appears on the screen.

The Flashing Cursor

Immediately after the beep, the cursor will blink in the upper left hand corner of the screen. The cursor is an indicator to you of your position on the screen.

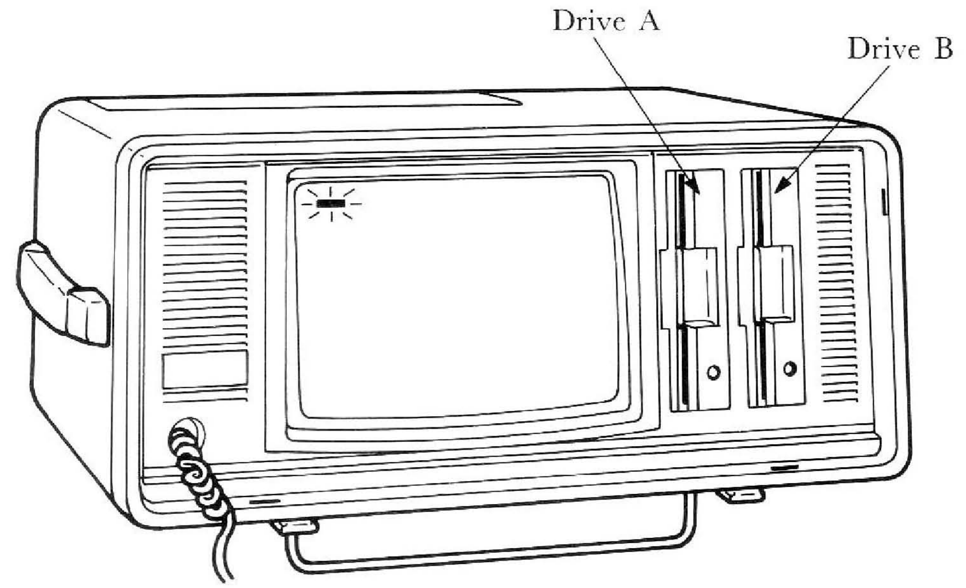


Fig. 4-18. THE FLASHING CURSOR

The In Use Indicator

Everytime you bring up the system it cycles through a series of internal tests to be sure its memory and all its parts are in working order. You hear this testing as a strange combination of clicks and whorls. DO NOT PANIC, the sounds tell you that the computer is working just fine.

The Sr. Partner is reading its instructions from the System Disk. You will hear the disk drive transferring information. The **In Use Indicator** lights up when the computer is accessing a disk, either reading from it or writing to it.

> REMEMBER: NEVER UNLOCK THE DRIVE DOOR OR ATTEMPT TO REMOVE A DISK WHEN THE IN USE INDICATOR IS ON!

OPERATION

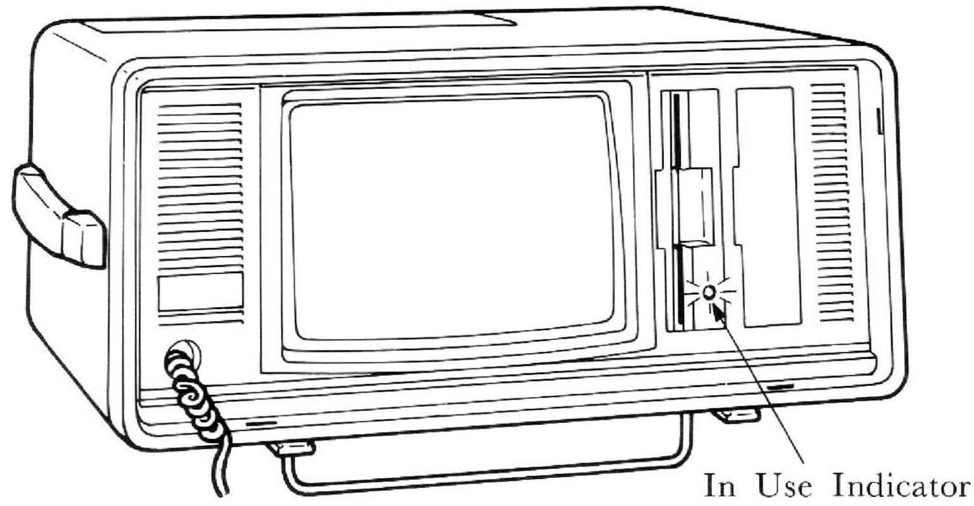


Fig. 4-19. THE IN USE INDICATOR

The Opening Screen

When the computer completes it self-test, this message will appear on the screen:

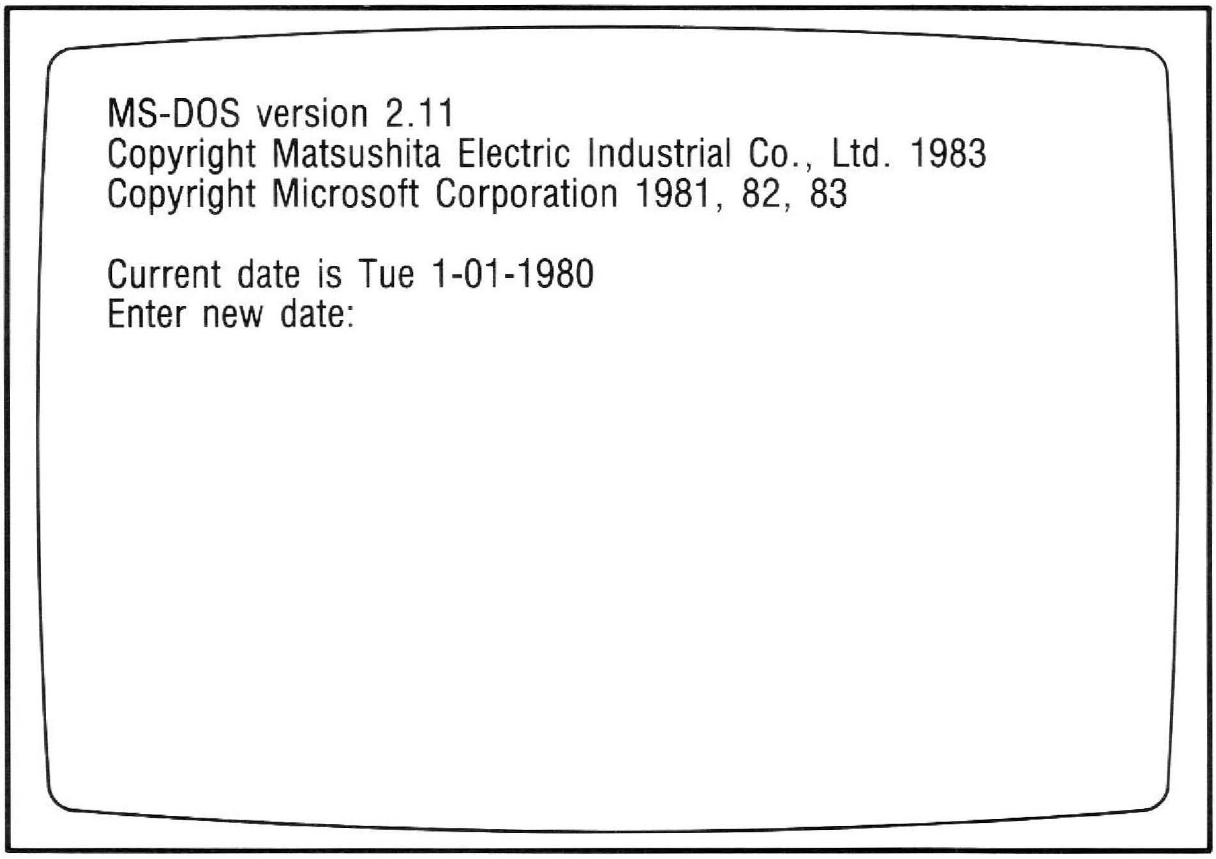


Fig. 4-20. THE OPENING SCREEN

When the system is up and running, this is the first screen you will see.

MS-DOS is the name of your operating system.

MS-DOS is a product of the Microsoft Corporation and is protected by copyright.

The current date (according to the computer!) is on the next line.

If this date is incorrect, the computer waits for a new date.

Prompts

The cursor at the end of the line tells you that the computer is waiting from a response from you.

Whenever the computer awaits input from you it displays a **prompt**. Different programs use different prompts. `A>` or `B>` is the prompt for DOS, `ok` is the prompt for BASIC. In this case, the cursor is the prompt.

If the Opening Screen Doesn't Appear

When the start-up tests indicate a problem, your screen will not look like this. As the computer shifts into a full diagnostic mode, your screen will fill with numbers. THIS IS VERY UNLIKELY TO OCCUR. If it does happen you have two choices.

You can turn OFF the machine and notify your dealer.

In all likelihood, you can diagnose the problem quite quickly and easily. See Appendix A for simple step-by-step procedures.

Entering the Date

```
Current date is Tue 1-01-1980
Enter new date:_
```

Using the number keys in the typewriter section of your keyboard, set the date on your Sr. Partner:

Enter the month as a one or two digit number, 1 to 12.

Digits must be separated by a HYPHEN (-) or a SLASH (/).

Enter the date as a one or two digit number, 1 to 31.

Enter the year as a two digit number from 83 to 99. You may enter four digits if you wish (1983). The computer will accept years up to 2099.

You must end the entry with <ENTER>.

If you hit the wrong key while entering the date, simply use <BACKSPACE> to erase the entry and type it again. You must correct the entry BEFORE YOU PRESS <ENTER>.

Now type in the date. For the example, we will assume it is May 1, 1984.

You may use any of these formats to enter the date:

```
5-1-84 <ENTER>
5/1/84 <ENTER>
5-01-84 <ENTER>
5/01/84 <ENTER>
05-01-84 <ENTER>
05/01/84 <ENTER>
05/1/84 <ENTER>
```

The computer does not "remember" the date. You must reset the date each time you turn on the computer.

If you do not wish to modify the date, simply press <ENTER>.

Entering the Time

```
Current time is 0:05:18.02
Enter new time:
```

Once you have entered the date, the computer requests the current time. The computer is very precise, time is reported in hours, minutes, seconds and hundreds of seconds.

To set the time on your Sr. Partner:

Enter the hour as a one or two digit number, 0 to 23.

Enter a COLON (:)

Enter the minutes as a one or two digit number, 0 to 59.

End the entry with <ENTER>.

This is all the information that the computer requires. If you wish you may enter seconds and hundreds of a second.

Enter a COLON (:).

Enter the seconds as a one or two digit number, 0 to 59.

ENTER a PERIOD (.).

Enter the hundreds of a second as a two-digit number 00 to 59.

End the entry with <ENTER>.

```
9:30:0.00 <ENTER>
09:30:0.00 <ENTER>
9:30:0 <ENTER>
9:30 <ENTER>
```

The computer does not "remember" the time. You must reset time each time you turn on the computer.

If you do not wish to modify the time, simply press <ENTER>.

The DOS Prompt A>

This is the real starting point of your computing experience. When the A> prompt appears, it tells you DOS is up and ready to operate.

The very first operation you will perform on your Sr. Partner is to make a backup of the System Disk. Since the computer cannot operate with the System Disk, and you have only the original copy, MAKE THE BACKUP SYSTEM DISK FIRST!!!


```
MS-DOS version 2.11
Copyright Matsushita Electric Industrial Co., Ltd. 1983
Copyright Microsoft Corporation 1981, 82, 83

Current date is Tue 1-01-1980
Enter new date: 5-1-84


Current time is 0:05:18.02
Enter new time: 9:30:0.00

A>
```

Fig. 4-21. THE DOS PROMPT

The Brightness Control

Before you begin the process of backing up the System Disk, be sure that you can read the display comfortably. You can vary the contrast on your screen by adjusting the brightness control.

The control is located along the bottom edge of the rear panel. Turn it now to find the best setting for your eyes and the lighting conditions in the room.

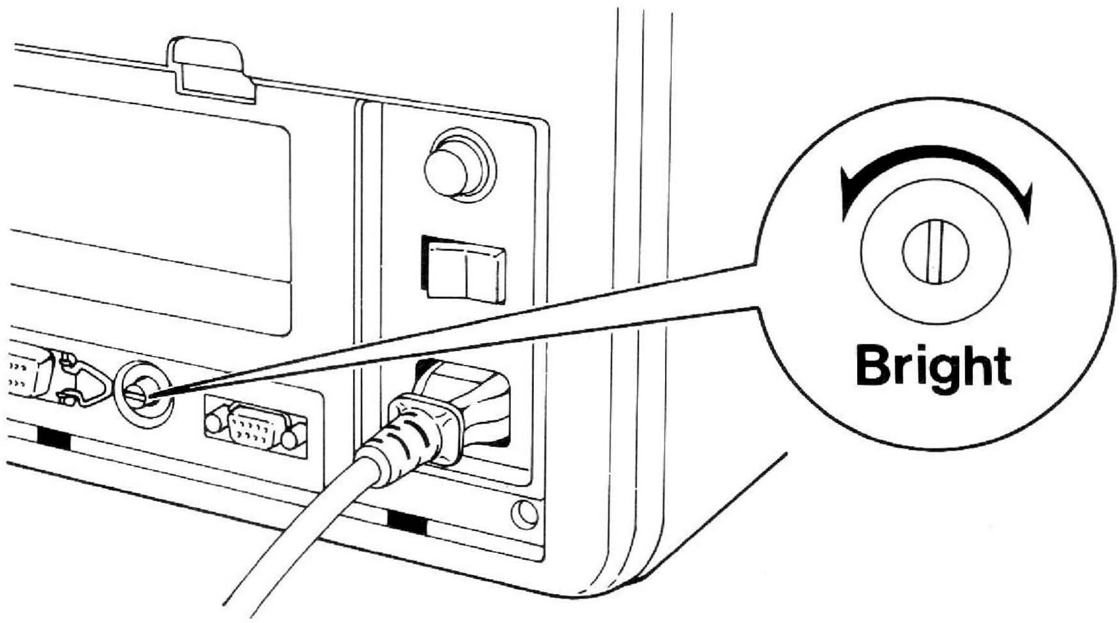


Fig. 4-22. THE LOCATION OF BRIGHTNESS CONTROL

BACKING UP THE SYSTEM

What is a Backup?

The very first operation you should perform on your new Sr. Partner is to make a copy of your System Disk. This provides insurance for you, in case the original disk becomes damaged. This copy is called a "backup". Making the copy is called "backing up" the system.

It is a good practice to use your "backup" System Disk in your everyday operations. Store the original System Disk in a safe location.

You will need two disks to perform this operation. The first, your System Disk, is the one you wish to backup. The disk you are "copying from" is called the original or SOURCE disk. Your source disk is already loaded in Drive A.

The second disk will become the backup. It is the disk you will be "copying to". This is the destination or TARGET disk. Have on hand a new, unformatted disk.

original system disk (copy from)=source disk

new, unformatted disk (copy to)=target disk

Your System Disk is write-protected. This means you cannot write to the disk. This protects you from erasing the stored information on the disk (see WRITE-PROTECT NOTCH earlier in the chapter.) If you get confused and attempt to write to your System Disk, this message will appear:

```
Write protect error writing drive A(B)
Abort, Retry, Ignore?
```

OPERATION

This manual contains two procedures for backing up the system. The first is for systems with two drives. It begins on page 4-43.

The second is for systems with one disk drive. It begins on page 4-46.

Device Designations

Before you begin working with the DOS, disk operating system, you should become familiar with drive designations.

You must tell DOS where a file or where to put the results of a procedure. You do this by using drive designations **A:** and **B:**.

A: or a:	Tells DOS to get this information **from** or write this information **to** the disk in Drive A.
B: or b:	Tells DOS to get this information **from** or write this information **to** the disk in Drive B (when you have two disk drives).

OPERATION

Default Drive

The DOS prompt A> means that DOS is currently getting and sending information from the disk in Drive A. DOS always assumes that the file you want is on the current disk. As long as you want to read or write from the current drive, you do not need to include the drive designator in commands.

When you have two drives, however, you may want to read from or write to the disk in Drive B. Then you must include B: in your instructions to the computer.

You can change the default drive. Suppose you wanted to work with a group of files on the disk in Drive B. You don't want to specify B: in every file name. You can change the default drive to B.

```
A>
```

This is the DOS prompt. It tells you that DOS in ready, but it also tells you that Drive A is the default drive.

YOU TYPE:

```
b:
```

and press <ENTER>

SCREEN DISPLAYS:

```
B>
```

B> is also a DOS prompt. It tells you that Drive B is the default drive.

To return to A as the default just type a: in response to the B> prompt.

NOTE: DOS always assumes it will find the file or write the results to the default drives unless you indicate a drive specifier in your commands.

The DISKCOPY Command

The DISKCOPY command transfers the entire contents of one disk to another disk. You will use DISKCOPY to make your Backup System Disk. In the future, use DISKCOPY any time to want to copy and entire disk.

Using DISKCOPY With Two Drivers

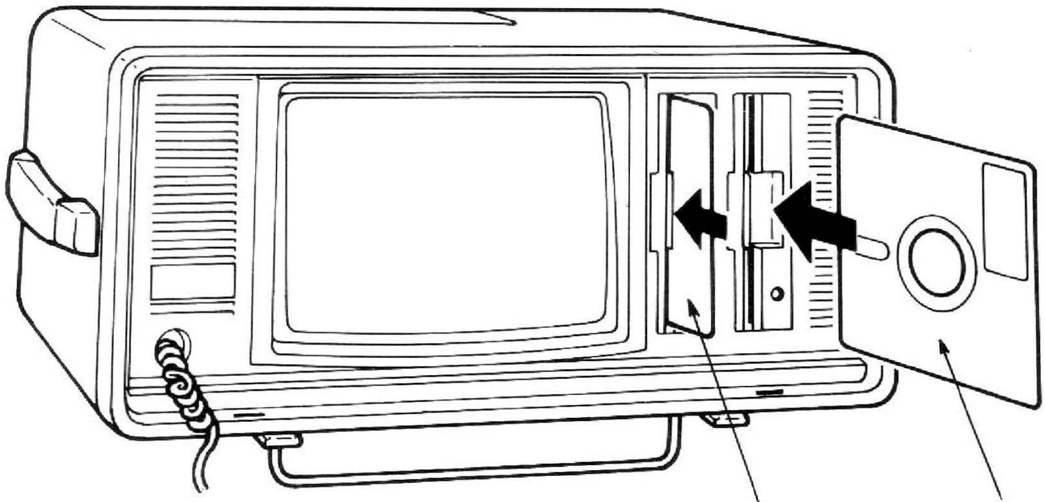


Fig. 4-23. USING DISKCOPY WITH TWO DRIVES

STEP 1 Be sure you have inserted your SOURCE disk (System Disk) into Drive A.

STEP 2 Insert a new, unformatted disk into Drive B. Hold the disk by its label (with the label facing left.)

STEP 3 SCREEN DISPLAYS:

A>

YOU TYPE:

diskcopy a: b:

STEP 4 Press <ENTER>

SCREEN DISPLAYS:

Insert source disk into drive A:
Insert target disk into drive B:
Press any key when ready

STEP 5 Since your disks are already inserted, press any key.

SCREEN DISPLAYS:

Copying 9 sectors per track, double sided

Formatting while copying

The copying is done automatically. While the information from Drive A is being transferred to Drive B you will hear the drives moving, and see the In Use Indicators come off and on.

In a short time the copying will be completed.

SCREEN DISPLAYS:

```
Copy complete

Copy another (Y/N)?
```

STEP 6 TYPE:

```
y
```

if you want to make another copy

TYPE:

```
n
```

to end the diskcopying session.

SCREEN DISPLAYS:

```
A>
```

to indicate that the diskcopying session is completed.

NOTE: Now that you have successfully completed the copying of your original system disk, return it to its jacket and store it carefully according to the directions for disk storage.

Label your copied disk "system disk-backup". Use this copy in your everyday operations. If your backup disk becomes damaged or inoperable make a new backup from your original System Disk.

Turn to page 4-51.

OPERATION

The DISKCOPY Command

The DISKCOPY commands transfers the entire contents of one disk to another disk. You will use DISKCOPY to make your Backup System Disk. In the future, use DISKCOPY any time to want to copy an entire disk.

Using DISKCOPY With One Drive

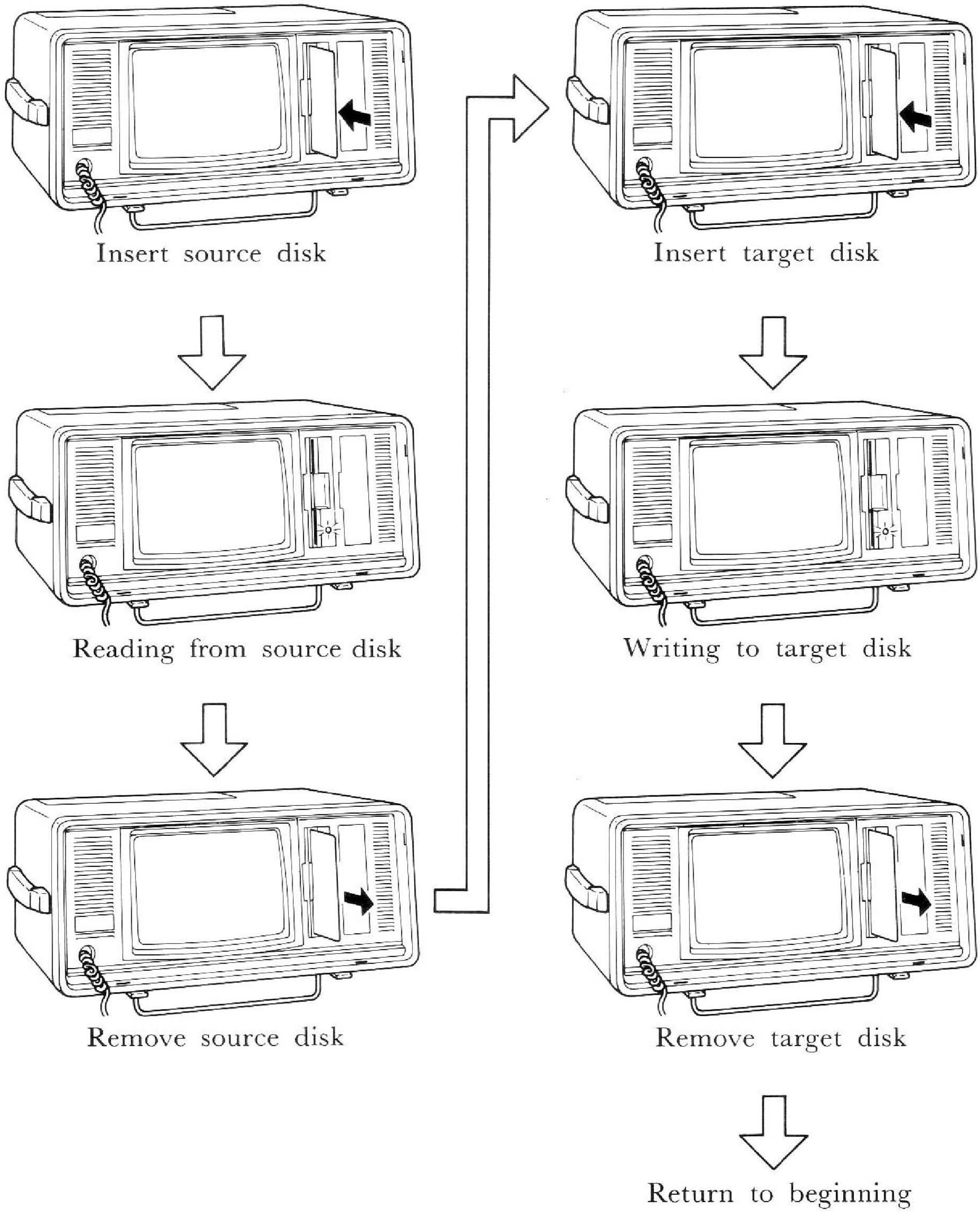


Fig. 4-24. USING DISKCOPY WITH ONE DRIVE

When you make copies with two drives, the information comes off the SOURCE disk, passes through the computer, and is recorded immediately on the TARGET disk.

The same process happens when you copy with a single drive. But in order to pass on the information from the SOURCE disk, the computer must momentarily hold it in memory, while you insert the TARGET disk. This means you may have to switch disks several times during the copying procedure.

The messages are easy to follow if you remember:

original disk (copy from)=source disk
backup disk (copy to)=target disk

OPERATION

STEP 1 Be sure you have inserted your SOURCE disk in Drive A.

STEP 2 Remove a new, unformatted disk from its package. This will be your TARGET disk.

STEP 3 SCREEN DISPLAYS:

```
A>
```

TYPE:

```
diskcopy
```

STEP 4 Press <ENTER>

SCREEN DISPLAYS:

```
Insert source disk into drive A:
Press any key when ready
```

STEP 5 Since your source disk is already in drive A, press any key.

You will see the In Use Indicator for Drive A come on as the original disk is being read into memory.

SCREEN DISPLAYS:

```
Copying 9 sectors per track, double sided
Insert target disk into drive A:
Press any key when ready
```


STEP 6 Remove your source Disk. Insert your new, unformatted target disk into the disk drive.

STEP 7 Press any key.

You will see the In Use Indicator come on as the information is written to the Target Disk.

SCREEN DISPLAYS:

```
Formatting while copying
Insert source disk into drive A:
Press any key when ready
```

The computer has formatted and copied one section of the SOURCE disk onto the TARGET disk. It now needs the next section to store into memory.

STEP 8 Remove the target disk from the drive and insert the source disk.

STEP 9 Press any key.

SCREEN DISPLAYS:

```
Insert target disk into drive A:
Press any key when ready
```

The computer has now written the second section to the backup disk.

OPERATION

STEP 10 You will need to keep "swapping" the source and target disk until the entire disk has been copied.

When the procedure is completed the computer sends this message:

SCREEN DISPLAYS:

```
Copy complete

Copy another (Y/N)?
```

STEP 11 TYPE:

```
y
```

if you want to make another copy

```
n
```

to end the session.

SCREEN DISPLAYS:

```
A>
```

to indicate that the diskcopying sessions is completed.

NOTE: Now that you have successfully completed the copying of your original system disk, return it to its jacket and store it carefully according to the directions for disk storage.

OPERATION

Label your copied disk "system disk-backup". Use this copy in your everyday operations. If your backup disk becomes damaged or inoperable make a new backup from your original System Disk.

GETTING ACQUAINTED WITH DOS

To help you get started using your new machine and to acquaint you with some of its capabilities, this chapter will introduce you to some frequently used commands in the DOS, disk operating system.

In this section you will learn how to use the following commands:

FORMAT

DIR

TYPE

COPY

RENAME

ERASE

MODE

Chapter 5 through 12 of this manual present a comprehensive discussion of your operating system and is a useful source of general and reference information about DOS.

Files

Like a enormous file cabinet, your computer stores all of its information in files. In order to find data the computer must be instructed exactly where to look. (Don't forget the computer, by itself, is dumb. All it can do is follow instructions!)

Some files come built into your Sr. Partner. These files help the computer to regulate itself and operate the various parts that make up your system. These internal files are stored in ROM.

OPERATION

Your System Disk also contains operating files for your Sr. Partner. Once they are loaded into memory they perform automatically upon demand.

The disks which operate your application programs also store information files.

When you enter information into the computer, this too must be arranged in files.

You will also use files to read in and store your data.

A File's Name

A file is simply a collection of related data stored on a single disk. You can have many different types of files. For example:

SCHED	Contains your upcoming appointments
ACCTREC	Contains your accounts receivable
BDAY	Contains a list of important birthdays.
FORMLET	Contains a form letter
GAMES	Contains educational programs

You may store several files on one disk. Sometimes the files may be related. Or a disk may contain all of the files which you access on a regular basis. It does not matter to the computer how you organize and store your files.

> **NOTE:** Each file on a disk must have a unique name. If you store information under an existing file name, the old information will be replaced by the most current entry. This is a good way to update files, but be careful not to inadvertently lose data by assigning it to an already existing filename.

Filenames and Extensions

A file's name is made up of two components. The first section is the **filename**, the second section is the **extension**.

A **filename** may contain as many as eight **characters**. The **extension** always begins with a **period** and may contain up to **three additional characters**.

Here are the files already named with file extensions:

SCHED.JUL

ACCREC.CUR

BDAY.BAS

FORMLET.84

GAMES.JOD

NOTE: Extensions are optional. However, if you do assign an extension to a file's name, or the file you want to use has a predefined extension, you must include the extension when referring to the file.

What to Name a File

DOS allows up to eight characters in the filename, a period plus up to three additional characters in the extensions. Within these restrictions you can name a file almost anything you want. But remember, the file's name is for your reference so be sure the name is convenient to type and easy to remember.

You can use any of these characters to name your file:

The letters of the alphabet (upper or lower case)

The numbers 0 through 9

These special characters-

$ # & @ ! % ( ) - ' _

What NOT to Name a File

DOS does have some restrictions on filenames. Here are some unacceptable file names:

Q and R	Filenames cannot contain spaces
"SPEECH"	Filenames cannot contain quotation marks
A,B,C	Filenames cannot contain commas
.NG	Filename is missing
###@@.***	This name would be acceptable but who knows what the file contains??

TOOMANYCHARACTERSINTHISNAME

For more information on filenames, see Chapter 5 "Using DOS".

OPERATION

Drive Specifiers

In additions to the filename and optional extensions, the computer must know one more piece of information to retrieve or store a file. It must know WHERE to find the file.

To indicate which disk the file resides on you must include a **drive specifier** in your DOS commands. The two specifiers are A: and B: (indicating the disk in Drive A or the disk in Drive B).

The drive specifier + filename + extension equal the **file specification**. Do not put any spaces between the three parts.

Here are the files already names with complete file specifications:

A:SCHED.JUL

A:ACCREC.CUR

B:BDAY.BAS

A:FORMLET.84

B:GAMES.JOD

Sometimes you don't need to type the drive specifier. If the file you are creating will reside in the default or current drive, or the file you want to read is in the default drive you don't need to indicate the specifier. So if the default drive is A (indicated by the A> prompt) you could type either:

A:SCHED.JUL

or

SCHED.JUL

Creating a File

Managing files with DOS is simply a matter of practice. DOS is really very easy to use and in no time at all you will be wondering how you managed to do any work in pre Sr. Partner days!

While you are learning however, you may make a few small mistakes. So, to prevent any of your valuable data from being damaged, we are going to create a practice file to use while you acquaint yourself with some fundamental DOS commands.

You use EDLIN to create a file. EDLIN is a special part of DOS which allows you to create, change and display files. Chapter 8 of this manual provides full instruction for using EDLIN but for now simply follow the steps below to create your practice file.

The name of our file will be **sample**. Although we could add an extension, lets just keep the name simple for now. **sample** will be located on your backup system disk.

> **NOTE:** Do not proceed with these exercises unless you have made a backup of your original system disk see "backing up the system" earlier in this chapter.

When instructions refer to your System Disk, they are referring to this backup version.

STEP 1 Insert your System Disk in Drive A. (It is assumed Sr. Partner is already on)

SCREEN DISPLAYS:

```
A>
```

STEP 2 YOU TYPE:

```
edlin sample
```

OPERATION

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
New File
*
```

The * is the prompt for EDLIN

STEP 4 YOU TYPE:

```
i
```

The i stands for Insert, it means you want to add information to the named file (in this case create the file).

STEP 5 Press <ENTER>

SCREEN DISPLAYS:

```
1:*
```

The number 1 stands for the line number. EDLIN references all input by line numbers. Each time you press <ENTER> a new line number will appear at the beginning of the next line.

STEP 6 YOU TYPE

```
This is a practice file.
```

STEP 7 Press <ENTER>

SCREEN DISPLAYS:

```
1:*This is a practice file.
2:*_
```

OPERATION

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
New File
*
```

The * is the prompt for EDLIN

STEP 4 YOU TYPE:

```
i
```

The i stands for Insert, it means you want to add information to the named file (in this case create the file).

STEP 5 Press <ENTER>

SCREEN DISPLAYS:

```
1:*
```

The number 1 stands for the line number. EDLIN references all input by line numbers. Each time you press <ENTER> a new line number will appear at the beginning of the next line.

STEP 6 YOU TYPE

```
This is a practice file.
```

STEP 7 Press <ENTER>

SCREEN DISPLAYS:

```
1:*This is a practice file.
2:*_
```

STEP 8 Continue typing in the text for the practice file.

YOU TYPE and PRESS:

We will call it sample. <ENTER>
Sample will help explain DOS. <ENTER>

NOTE: If you make a mistake while typing in the contents of the file, use <BACKSPACE> to erase the characters and then retype the entry. You must erase a line before you type <ENTER>.

OPERATION

Even if you do enter incorrect lines this is only a practice file. The actual contents don't really make any difference.

STEP 9: The screen should look like this:

SCREEN DISPLAYS:

```
A>edlin sample
New File
*i
      1:*This is a practice file.
      2:*We will call it sample.
      3:*Sample will help explain DOS.
      4:*_
```

STEP 10 Now indicate the end of the file.

YOU PRESS:

<Ctrl> <z> (hold down <Ctrl> while pressing <z>, then let go off both keys)

SCREEN DISPLAYS:

```
4:*^Z _
```

STEP 11 Press <ENTER>

SCREEN DISPLAYS:

    *

The file is completed and we are back to the EDLIN prompt.

STEP 12 To check the contents of your file ask for a **listing** of the file.

YOU TYPE:

    l (lowercase letter L)

and press <ENTER>

The screen will type out your sample file.

STEP 13 Having created the file, we need to exit from EDLIN and return to DOS.

YOU TYPE:

    e

and press <ENTER>

This command will save the file to the default drive. The In Use Indicator will light up and you will hear the drive turning. The sample file is being written to the disk in drive A.

SCREEN DISPLAYS:

    A>

to indicate DOS is now ready.

Giving DOS Commands

Now you are ready to begin using DOS. Actually, you already have used DOS to backup your system. DISKCOPY is a DOS command. Here are a few guidelines to remember when entering DOS commands:

Enter Command after the DOS prompt (A>) is displayed.

Type in the command and any other parts the command requires (drive specifier, extensions).

You may use upper or lower case characters.

Use a blank (<SPACEBAR>) to separate the parts of the command from each other.

If you make a mistake while entering a command, use <BACKSPACE> to erase the characters and enter the command again. Correct a command before pressing <ENTER>.

Press <ENTER> when you have finished typing.

How Many Drives Do You Have?

Some commands follow a slightly different procedures depending on the number of drives in your system. This was demonstrated in the procedures for backing up the system in the last section. To avoid confusion as to command procedures, we will divide this section into two parts.

For two drive systems the commands begin on the next page.

For single drive systems the commands begin on page 4-83.

THIS SECTION CONTAINS COMMANDS FOR TWO DRIVE SYSTEMS

The FORMAT Command

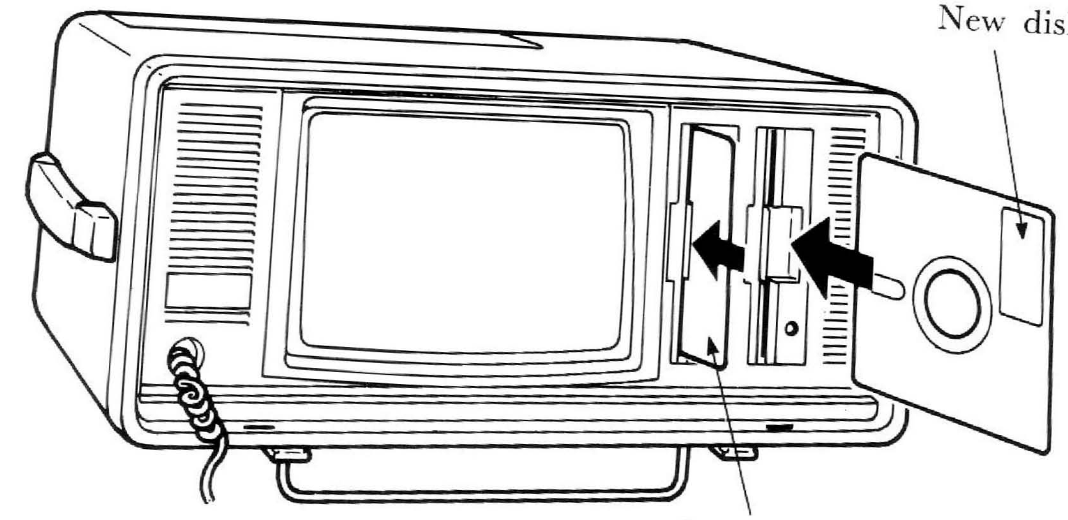
When you purchase disks they are blank. Specific computers store information on disks in different formats. Before you can use a new disk, you must specify the format of your YOUR SPECIFIC MACHINE. To get the disk ready to receive information, use the FORMAT Command.

You only format a disk once, the very first time you use it. Since FORMAT treats the disk as a blank it will erase any information already on a disk.

You can use FORMAT to prepare a disk for new information. Just be sure you NO LONGER NEED the data on the disk.

> **NOTE:** You do not need to format a disk before using the DISKCOPY command. DISKCOPY will perform the format operation first if necessary, then copy the data.

Using the FORMAT Command with Two Disk Drives



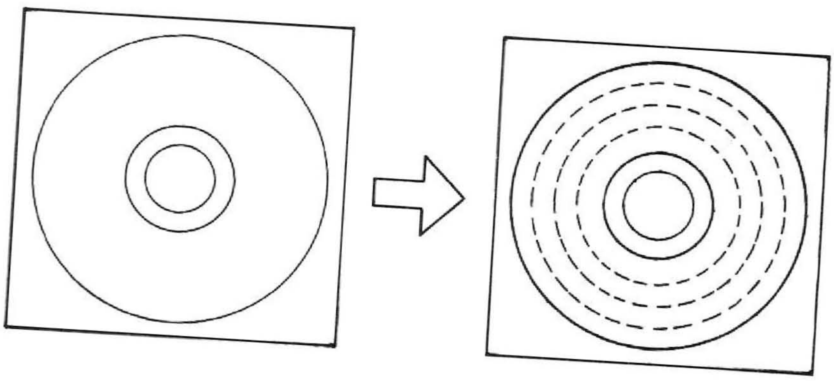


Fig. 4-25. INSERTION OF SYSTEM DISK IN DRIVE A AND BLANK DISK IN DRIVE B

OPERATION

STEP 1 Insert your System Disk in Drive A. It is a good idea to check and insure the disk has a write protect tab on it.

STEP 2 Insert your new unformatted disk (Target Disk) in Drive B.

STEP 3 SCREEN DISPLAYS:

```
A>
```

STEP 4 YOU TYPE:

```
format b:
```

NOTE: Be sure to include the drive specifier B: Otherwise DOS will format the disk in the default drive (Drive A) and you will erase your System Disk.

STEP 5 Press <ENTER>

SCREEN DISPLAYS:

```
Insert new disk for drive B:
and strike any key when ready
```

STEP 6 Since you have already inserted a new disk in drive B, press any key.

SCREEN DISPLAYS:

```
Formatting...
```

The In Use Indicator for Drive B will light up. You will hear the drive operating as it formats the disk.

OPERATION

The computer tells you when it has completed the operation.

SCREEN DISPLAYS:

```
Formatting... Format complete

    xxxxxx bytes total disk space
    xxxxxx bytes available on disk

Format another (Y/N)?
```

STEP 7 YOU TYPE:

```
y
```

if you want to format another disk.

```
n
```

to end the formatting session.

SCREEN DISPLAYS:

```
A>
```

to indicate the system is ready for another operation.

NOTE: They are several options available within the FORMAT command. See FORMAT command in Chapter 6 of this manual.

OPERATION

The DIR Command

How does the operating system keep track of all your files? Well, just like any other well-organized information system, the computer has a **directory**. Each disk contains the directory for its files.

You do not need to update the directory as you add, delete or rename files. The computer does it automatically, so the directory always indicates the current status of the disk.

Along with the names of the files, the directory also contains information on the sizes of the files and the total amount of space utilized on the disk.

To access this information, you use the DIR command.

View the Directory of the Current Disk

STEP 1 Insert your System Disk in Drive A.

STEP 2 YOU TYPE:

```
dir
```

or

```
dir a:
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
Volume in drive A has no label

Directory of A:\

COMMAND    COM    16229    12-21-83    4:00p
ANSI        SYS    1715    12-21-83    4:00p
CHKDSK     COM     6468    12-21-83    4:00p
DEBUG      COM    12146    12-21-83    4:00p
DISKCOMP   COM     2464    12-21-83    4:00p
DISKCOPY   COM     2318    12-21-83    4:00p
EDLIN      COM     8110    12-21-83    4:00p
FORMAT     COM     6475    12-21-83    4:00p
GRAPHICS   COM      679    12-21-83    4:00p
MODE       COM     2306    12-21-83    4:00p
MORE       COM     4364    12-21-83    4:00p
PRINT      COM     3808    12-21-83    4:00p
RECOVER    COM     2295    12-21-83    4:00p
SYS        COM     1588    12-21-83    4:00p
TREE       COM     1374    12-21-83    4:00p
EXE2BIN     EXE    1649    12-21-83    4:00p
   •
   •
   •
   •
SAMPLE                1     5-01-84    9:53a
            XXXXXX          XXXXX     XXXXX
A>
```

The directory lists several pieces of information about each file.

The file names are listed in the first column (extensions are included when present—COM is an extension or a Command file).

NOTE: It is quite possible that there are some changes between this list of Directory in the Reference Guide and that one on the screen.

OPERATION

The second column indicates the number of bytes occupied by the file.

The third column lists the date that the file was last accessed.

The last column gives the time of last use.

The total number of files, and the amount of free space still on the disk are given at the bottom.

Notice the last file listed. It is the file we created to use in these DOS exercises. Be sure you are using the System Disk which contains the sample file.

When your directory is very large, it may not all fit on the screen at one time. Then when you type dir, the display goes flying by, rolling off the top to be replaced by new files on the bottom. To view the directory in sections:

YOU PRESS:

<Ctrl> <s> or <Ctrl> <Num Lock>

This will freeze the directory on the screen. To resume scrolling (adding information from the bottom as it goes off the top) press any key.

View the DIR of the Disk in Drive B

DIR displays the names of the files on the disk in the default drive (the default drive is always A unless otherwise specified.)

To view the directory of the disk in drive B:

YOU TYPE:

```
dir b:
```

Then press <ENTER>.

Even after you request the directory for Drive B, the A> prompt returns because A is still the default drive.

You may also use the DIR command to verify that a specific file is resident on the disk. To find out the file size, date and time of a file:

YOU TYPE:

```
dir sample
```

and press <ENTER>

The screen will display the information for the sample file.

If you wanted to find out the file existed on the disk in Drive B:

YOU TYPE:

```
dir b: sample
```

and press <ENTER>

OPERATION

The COPY Command

The **COPY** command makes a copy of an individual file (DISKCOPY makes a copy of an entire disk.)

Use COPY when you:

need two copies of the file

need to backup a changed file on a disk

need to make changes in a file but want a backup of the original

want to rearrange a file (like cutting and pasting print copy)

There are several ways to copy files:

copy a file to another disk using the same name

copy a file to another disk using a new name

copy a file to the same disk using a new name (you cannot have two files with the same name on the same disk)

copy a file to the same disk using a name already in use on the disk (this replaces the information stored under that filename)

Using the COPY Command with Two Drives

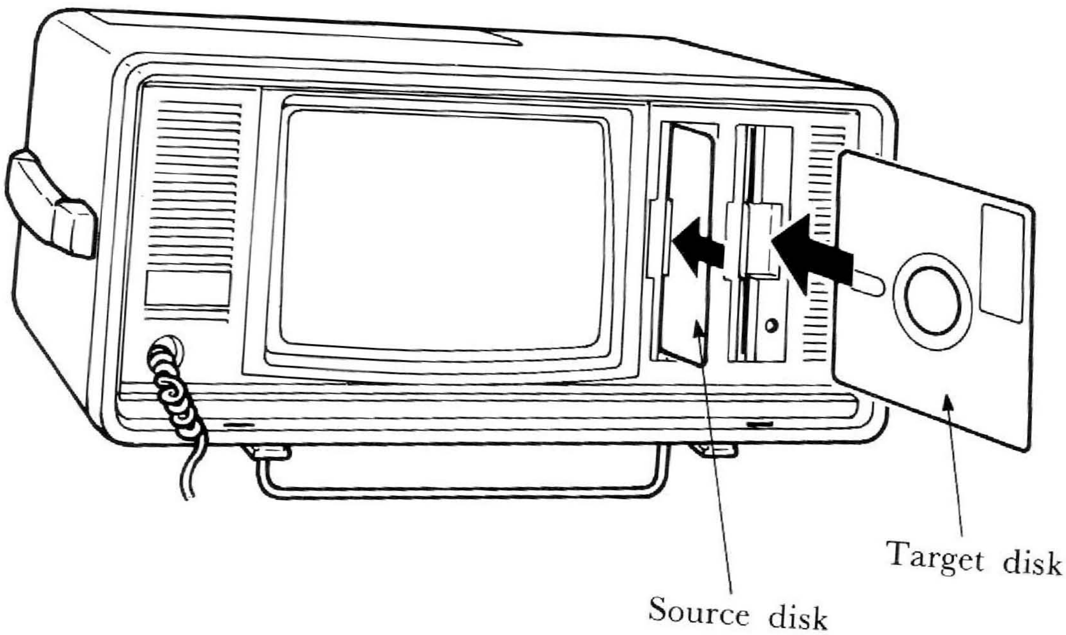


Fig. 4-26. USING THE COPY COMMAND WITH TWO DRIVES

OPERATION

COPY the File to Another Disk (Same Name)

STEP 1 Insert the source disk in Drive A and the target disk in Drive B.

SCREEN DISPLAYS:

```
A>
```

STEP 2 YOU TYPE:

```
copy sample b:
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
1 File(s) copied
```

STEP 4 (Optional)

You can verify that the file has been copied by using the DIR command.

YOU TYPE:

```
dir b:
```

and press <ENTER>

SCREEN DISPLAYS:

```
Volume in drive B has no label
Directory of B:\

SAMPLE 106   1-01-80   12:22a
1 File(s)      361472 bytes free
```

OPERATION

COPY the File to Another Disk (Change the Name)

You can give the file a new name when you copy it to a different disk. The file contents will still be the same.

In STEP 2 YOU TYPE:

copy sample b:example

and press <ENTER>

NOTE: If you tried this exercise, you now have two copies of the file on drive B—SAMPLE and EXAMPLE. Both files contain the contents of the original sample file on the disk in Drive A.

OPERATION

COPY the File to the Same Disk

You must change the name of your file if you want to make a copy of it on the same disk. Each file on a disk must have a unique name, OTHERWISE THE MOST CURRENT FILE WILL REPLACE THE EXISTING FILE. In our example we will change the name of our file to practice.

STEP 1 Insert the source disk in Drive A.

SCREEN DISPLAYS:

A>

STEP 2 YOU TYPE:

copy sample practice

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

1 File(s) copied

STEP 4 (Optional)

You may check to see that both files are on the disk using the DIR command.

The TYPE Command

The **TYPE** command lets you display the contents of a file. You can also print the contents at the same time. When you are using TYPE you are only "looking" at the file. You cannot access a file to make changes using the TYPE command.

In order to use TYPE you must know the exact name of the file including any file specifications. To be sure the file is on the disk you are using, or to check for the exact file name, use the DIR command.

OPERATION

TYPE the Contents of a File on the Current Disk

STEP 1 Insert the System Disk in Drive A (original sample file).

SCREEN DISPLAYS:

```
A>
```

STEP 2 YOU TYPE:

```
type sample
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS

```
This is a practice file.
We will call it sample.
Sample helps explain DOS.
```

NOTE: If you want to print the contents of a file, press <Ctrl> <PrtSc> before typing "type sample". When printing finishes, press <Ctrl> <PrtSc> again, then the printer is released.

TYPE the Contents of a File on Another Disk

If you want to see a file displayed which is on another disk you must add a drive specifier to the command.

STEP 1 Insert the disk containing the file in Drive B (the copied disk).

SCREEN DISPLAYS:

```
A>
```

STEP 2 YOU TYPE:

```
type b:sample
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
This is a practice file.
We will call it sample.
Sample helps explain DOS.
```

The RENAME Command

The **RENAME** command allows you to change the name of a file. You may change both the filename and the extension or only one of the components of the name.

RENAME is useful when you have two files whose names are very similar and you keep confusing them. Or may be at the time you named the file its contents seemed perfectly clear, but now the name is confusing. Most of the time you RENAME a file just because a different name would be more useful to you.

RENAME a File on the Current Disk

STEP 1 Insert the disk containing the file you want to rename into Drive A.
(Our System Disk contains both the **sample** file and the copied **practice** file)

SCREEN DISPLAYS:

```
A>
```

STEP 2 We are going to rename our file called **practice** to **exercise**.

YOU TYPE:

```
rename practice exercise
```

The "old name" comes first, then a space, and the "new name".

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
A>
```

STEP 4 The computer does not verify the rename procedure. To check on the rename use the DIR command.

YOU TYPE:

```
dir
```

and press <ENTER>

SCREEN DISPLAYS:

(the directory for A.)

Note that **practice** no longer appears. It has been replaced by **exercise**.

OPERATION

RENAME a File on Another Disk

STEP 1 Insert the disk containing the file to be renamed in Drive B. (This is the disk we have been coping to.)

SCREEN DISPLAYS:

```
A>
```

STEP 2 YOU TYPE:

```
rename b:example exercise
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
A>
```

STEP 4 Use the DIR command to verify the rename (remember to type dir b:).

The ERASE Command

Like everything that accumulates, you can occasionally find yourself with a lot of files—many of which are outdated, or have outlived their usefulness. **ERASE** is a convenient way to eliminate these files, leaving you usable space on your disks.

HOWEVER, ONCE YOU ERASE A FILE IT IS GONE FOREVER. USE THIS COMMAND WITH CARE.

OPERATION

Be especially careful when you are typing in the name of the file to be erased. This is crucial if you have files with similar names.

One note of hope. If you make backups regularly (and this is very strongly recommended) you may be able to retrieve a file which was accidently deleted from your backup.

NOTE: ERASE and DEL (delete) are the same command.

ERASE a File on the Current Disk

STEP 1 Insert the disk containing the file to be erased in Drive A.

(First time users insert the backup System Disk).

SCREEN DISPLAYS:

```
A>
```

STEP 2 Since we are now finished with our practice file, we will erase it from the system disk.

YOU TYPE:

```
erase sample
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
A>
```

STEP 4 The computer does not verify the ERASE procedure. To verify the deletion use the dir command.

If you wish you may also ERASE the **exercise** file from the disk.

OPERATION

ERASE a File From Another Disk

STEP 1 Insert the disk containing the file to be deleted in Drive B.

(First time users insert the disks containing the copied files.)

SCREEN DISPLAYS

```
A>
```

STEP 2 YOU TYPE:

```
erase b:sample
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS

```
A>
```

STEP 4 If you want to verify the deletion use the dir command (type dir b:)

If you wish you may also ERASE the **exercise** file from the disk in Drive B.

The MODE Command

The **MODE** command does not work on a file. This command is used to adjust the display on your screen. Use MODE when the first two or three characters on a line are not appearing on your screen.

Your System Disk must be in Drive A to use the MODE command.

STEP 1 Insert your System Disk in Drive A.

SCREEN DISPLAYS:

A> (you may not be able to see the prompt on the screen)

STEP 2 YOU TYPE:

mode ,r,t

(be sure there is a space and a comma before the r)

This form of the command shifts the display to the right.

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

01234567890123456789012...90123456789

Do you see the leftmost 0 ? (Y/N)

STEP 4 YOU TYPE:

y

if the screen looks okay

n

to shift the display to the right.

NOTE: When the LAST two or three characters on a line are not appearing on your screen, use

mode ,l,t

OPERATION

Using DOS

This has been an introduction to the many ways in which DOS can help you manage your files. Now that you are familiar with some of the DOS commands, you are ready to put DOS to work for you!

Your complete DOS reference manual begins on page 5-1.

THIS SECTION CONTAINS COMMANDS FOR SINGLE DRIVE SYSTEMS

The FORMAT Command

When you purchase disk they are blank. Specific computers store information on disks in different formats. Before you can use a new disk, you must specify the format of your YOUR SPECIFIC MACHINE. To get the disk ready to receive information, use the FORMAT Command.

You only format a disk once, the very first time you use it. Since FORMAT treats the disk as a blank it will erase any information already on a disk.

You can use FORMAT to prepare a disk for new information. Just be sure you NO LONGER NEED the data on the disk.

> **NOTE:** You do not need to format a disk before using DISKCOPY. DISKCOPY will perform the format operation first if necessary, then copy the data.

Using FORMAT with a Single Drive

Until you become accustomed to using your single drive system, the instructions from the computer may seem confusing.

In a single drive system, the computer call your drive "Drive A" even though there is no drive B. Because it must read instructions and data, temporarily store them in memory, and then perform the correct procedure on the second disk, you must "swap" two disks in Drive A.

To be sure you don't become confused, label your disks clearly.

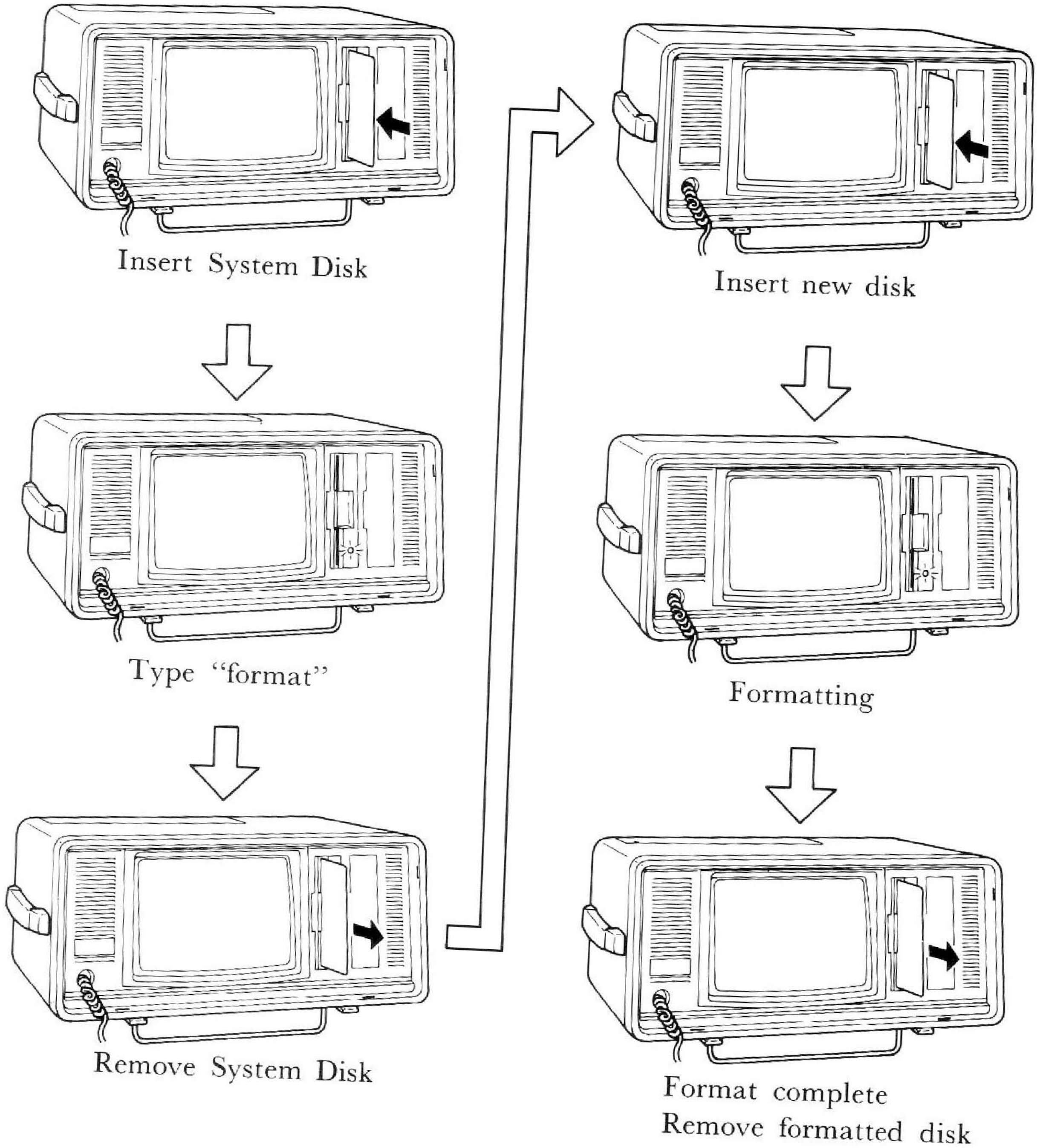


Fig. 4-27. USING FORMAT WITH A SINGLE DRIVE

STEP 1 Insert your System Disk in Drive A. Have on hand a new, unformatted disk.

SCREEN DISPLAYS:

```
A>
```

OPERATION

STEP 2 YOU TYPE:

format

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
Insert new disk for drive A:
and strike any key when ready
```

STEP 4 When the In Use Indicator goes off, remove your System Disk from Drive A. Insert your new unformatted disk.

OPERATION

STEP 5 Press any key.

SCREEN DISPLAYS:

```
Formatting...
```

The In Use Indicator for Drive A will light up. You will hear the drive operating as it formats the new disk.

The computer tells you when it has completed the operation.

SCREEN DISPLAYS:

```
Formatting... Format complete

        xxxxxx bytes total disk space
        xxxxxx bytes available on disk

Format another (Y/N)?
```

STEP 6 YOU TYPE:

```
y
```

if you want to format another disk.

```
n
```

to end the formatting session.

SCREEN DISPLAYS:

```
A>
```

to indicate the system is ready for another operation.

NOTE: There are several options available when using the FORMAT command. For a complete description of this command, see FORMAT in Chapter 6 "Command Reference".

OPERATION

The DIR Command

How does the operating system keep track of all your files? Well, just like any other well-organized information system, the computer has a **directory**. Each disk contains the directory for its files.

You do not need to update the directory as you add, delete or rename files. The computer does it automatically, so the directory always indicates the current status of the disk.

Along with the names of the files, the directory also contains information on the sizes of the files, and the total amount of space utilized on the disk.

OPERATION

To access this information, you use the DIR command.

View the Directory of the Current Disk

STEP 1 Insert your System Disk in Drive A.

STEP 2 YOU TYPE:

```
dir
```

or

```
dir a:
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
Volume in drive A has no label

Directory of A:\

COMMAND    COM    16229    12-21-83    4:00p
ANSI       SYS     1715    12-21-83    4:00p
CHKDSK     COM     6468    12-21-83    4:00p
DEBUG      COM    12146    12-21-83    4:00p
DISKCOMP   COM     2464    12-21-83    4:00p
DISKCOPY   COM     2318    12-21-83    4:00p
EDLIN      COM     8110    12-21-83    4:00p
FORMAT     COM     6475    12-21-83    4:00p
GRAPHICS   COM      679    12-21-83    4:00p
MODE       COM     2306    12-21-83    4:00p
MORE       COM     4364    12-21-83    4:00p
PRINT      COM     3808    12-21-83    4:00p
RECOVER    COM     2295    12-21-83    4:00p
SYS        COM     1588    12-21-83    4:00p
TREE       COM     1374    12-21-83    4:00p
EXE2BIN    EXE     1649    12-21-83    4:00p
   •
   •
   •
   •
SAMPLE                1     5-01-84    9:53a
```

The directory lists several pieces of information about each file.

The file names are listed in the first column (extensions are included when present—COM is an extension or a Command file).

The second column indicates the number of bytes occupied by the file.

NOTE: It is quite possible that there are some changes between this list of Directory in the Reference Guide and that one on the screen.

OPERATION

The third column lists the date that the file was last accessed. If you do not enter a new date when you turn on this computer, this date will not change even if you use the file.

The last column gives the time of last use.

The total number of files, and the amount of free space still on the disk are given at the bottom.

Notice the last file listed. It is the file we created to use in these DOS exercises. Be sure you are using the System Disk which contains the sample file.

OPERATION

When the directory is very large, it may not all fit on the screen at one time. Then when you type dir, the display goes flying by, rolling off the top to be replaced by new files on the bottom. To view the directory in sections:

YOU PRESS:

<Ctrl> <s> or <Ctrl> <Num Lock>

This will freeze the directory on the screen. To resume scrolling (adding information from the bottom as it goes off the top) press any key.

You may also use the DIR command to verify that a specific file is resident on the disk. To find out the file size, date and time of a file:

YOU TYPE:

dir sample

and press <ENTER>

The screen will display the information for the sample file.

The COPY Command

The **COPY** command makes a copy of an individual file (DISK-COPY makes a copy of an entire disk.)

Use COPY when you:

need two copies of the file

need to backup a changed file on a disk

need to make changes in a file but want a backup of the original

want to rearrange a file (like cutting and pasting print copy)

There are several ways to copy files:

copy a file to another disk using the same name

copy a file to another disk using a new name

copy a file to the same disk using a new name (you cannot have two files with the same name on the same disk)

copy a file to the same disk using a name already in use on the disk (this replaces the information stored under that filename)

Remember when you use a single drive you must "swap" the source and target disks. The messages for the COPY command sound as if there are two drives (Drive A and Drive B). To make it easier for yourself, imagine your disks as drives. The source disk is "Drive A", the target disk is "Drive B".

Using the COPY Command with a Single Drive

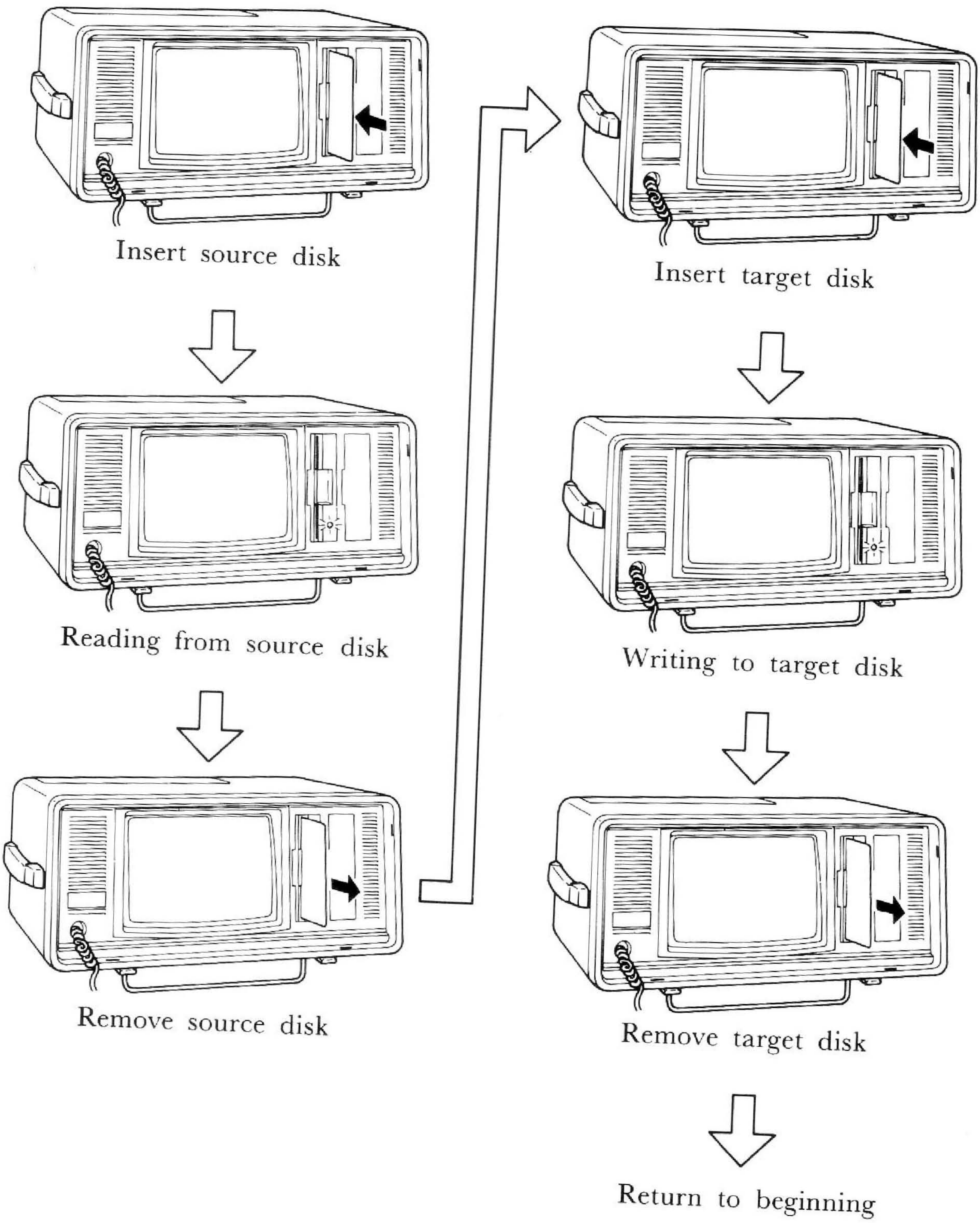


Fig. 4-28. USING THE COPY COMMAND WITH A SINGLE DRIVE

COPY the File to Another Disk (Same Name)

STEP 1 Insert the source disk in Drive A. Have on hand a new, formatted disk to be used as your target disk.

SCREEN DISPLAYS:

```
A>
```

STEP 2 YOU TYPE:

```
copy sample b:
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
Insert disk for drive B: and strike
any key when ready.
```

STEP 4 Remove your SOURCE Disk and insert the new disk in Drive A.

STEP 5 Press any key to tell DOS you have swapped the disks.

STEP 6 DOS can copy only so much information at one time. If your file is large you may see this message.

SCREEN DISPLAYS:

```
Insert disk for drive A: and strike
any key when ready.
```

OPERATION

STEP 7 DOS needs to read another section of the file into memory.

Remove the TARGET Disk from the drive and insert your SOURCE Disk.

STEP 8 Press any key to tell DOS you have swapped the disks.

STEP 9 After the additional information is read into memory, this message appears.

SCREEN DISPLAYS:

```
Insert disk for drive B: and strike
any key when ready
```

STEP 10 Remove the SOURCE disk and insert the TARGET disk to receive the next section of the file.

STEP 11 Press any key to indicate the disks have been swapped.

Continue alternating disks until this message is displayed.

SCREEN DISPLAYS:

```
        1 File(s) copied

A>
```

The A> prompt indicates that DOS is ready for another operation.

STEP 12 (Optional)

You can verify that the file has been copied by using the DIR command.

Be sure the disk which contains the copied file (TARGET disk) is in Drive A.

COPY the File to Another Disk (Change the Name)

OPERATION

You can give the file a new name when you copy it to a different disk. The file contents will still be the same.

In STEP 2 YOU TYPE:

copy sample b:example

and press <ENTER>

NOTE: If you tried this exercise, you now have two copies of the file on the target disk—SAMPLE and EXAMPLE. Both files contain the contents of the original SAMPLE file on the source disk.

COPY the File to the Same Disk

You must change the name of your file if you want to make a copy of it on the same disk. Each file on a disk must have a unique name, OTHERWISE THE MOST CURRENT FILE WILL REPLACE THE EXISTING FILE. In our example we will change the name of our file to practice.

STEP 1 Insert the SOURCE Disk in Drive A.

SCREEN DISPLAYS:

```
A>
```

STEP 2 YOU TYPE:

```
copy sample practice
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
1 File(s) copied
```

STEP 4 (Optional)

You may check to see that both files are on the disk using the DIR command.

OPERATION

The TYPE Command

The **TYPE** command lets you display the contents of a file. You can also print the contents at the same time. When you are using TYPE you are only "looking" at the file. You cannot access a file to make changes using the TYPE command.

In order to use TYPE you must know the exact name of the file including any file specifications. To be sure the file is on the disk you are using, or to check for the exact file name, use the DIR command.

OPERATION

TYPE the Contents of a File

STEP 1 Insert the disk containing the file to be displayed in Drive A.

SCREEN DISPLAYS:

```
A>
```

STEP 2 If you want to print the file while it is displayed, press <Ctrl> <PrtSc> (Hold down <Ctrl> while you press <PrtSc>. Then release both keys.)

STEP 3 YOU TYPE:

```
type sample
```

STEP 4 Press <ENTER>

SCREEN DISPLAYS

```
This is a practice file.
We will call it sample.
Sample helps explain DOS.
```

The contents of the file are printed at the same time. After printing finishes, press <Ctrl> <PrtSc> again, then the printer is deactivated.

The RENAME Command

The **RENAME** command allows you to change the name of a file. You may change both the filename and the extension or only one of the components of the name.

RENAME is useful when you have two files whose names are very similar and you keep confusing them. Or maybe at the time you named the file its contents seemed perfectly clear, but now the name is confusing. Most of the time you RENAME a file just because a different name would be more useful to you.

OPERATION

RENAME a File on the Current Disk

STEP 1 Insert the disk containing the file you want to rename into Drive A.
(Our System Disk contains both the **sample** file and the copied **practice** file)

SCREEN DISPLAYS:

A>

STEP 2 We are going to rename our file called **practice** to **exercise**.

YOU TYPE:

rename practice exercise

The "old name" comes first, then a space, and the "new" name.

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

A>

STEP 4 DOS does not verify the rename procedure. To check on the rename use the DIR command.

YOU TYPE:

```
dir
```

and press <ENTER>

SCREEN DISPLAYS:

(the directory for A.)

Note that **practice** no longer appears. It has been replaced by **exercise**.

OPERATION

The ERASE Command

Like everything that accumulates, you can occasionally find yourself with a lot of files—many of which are outdated, or have outlived their usefulness. **ERASE** is a convenient way to eliminate these files, leaving you usable space on your disks.

HOWEVER, ONCE YOU ERASE A FILE IT IS GONE FOREVER. USE THIS COMMAND WITH CARE.

Be especially careful when you are typing in the name of the file to be erased. This is crucial if you have files with similar names.

OPERATION

One note of hope. If you make backups regularly (and this is very strongly recommended) you will have a copy of any file which is accidently deleted.

NOTE: ERASE and DEL (delete) are the same command.

ERASE a File on the Current Disk

STEP 1 Insert the disk containing the file to be erased in Drive A.

SCREEN DISPLAYS:

```
A>
```

STEP 2 Since we are now finished with our practice file, we will erase it from the System Disk.

YOU TYPE:

```
erase sample
```

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

```
A>
```

STEP 4 DOS does not verify the ERASE procedure. To verify the deletion use the DIR command.

If you wish you may also ERASE the **exercise** file from the disk.

OPERATION

The MODE Command

The MODE command does not work on a file. This command is used to adjust the display on your screen. Use MODE when the first two or three characters on a line are not appearing on your screen.

Your System Disk must be in Drive A to use the MODE command.

STEP 1 Insert your System Disk in Drive A.

SCREEN DISPLAYS:

A> (you may not be able to see the prompt on the screen)

STEP 2 YOU TYPE:

mode ,r,t

(be sure there is a space and a comma before the r)

This form of the command shifts the display to the right.

STEP 3 Press <ENTER>

SCREEN DISPLAYS:

012345678901234567890 12...90123456789

Do you see the leftmost 0 ? (Y/N)

OPERATION

STEP 4 YOU TYPE:

y

if the screen looks okay

n

to shift the display to the right.

NOTE: When the LAST two or three characters on a line are not appearing on your screen, use

mode ,l,t

Using DOS

This has been an introduction to the many ways in which DOS can help you manage your files. Now that you are familiar with some of the DOS commands, you are ready to put DOS to work for you!

Your complete DOS reference manual begins on page 5-1.

MEMO

DEMONSTRATION SOFTWARE

Your new Sr. Partner can help you in many different ways. In the previous section you learned how DOS can help you in your everyday operations.

In this section you will get a preview of some of the capabilities and applications programs which make your Sr. Partner a very powerful tool.

The demonstration programs are contained in a file called **DEMO.BAT**. This file in contained on your System Disk.

Using DEMO.BAT

Turn ON the main power switch.

Insert your backup System Disk in Drive A.

After you have entered the date and time the A> prompt will appear.

YOU TYPE:

demo

and press <ENTER>.

Now watch the screen!

For a moment, your screen will look like this:

Sr.Partner™

```
1 == Graph
2 == Spread Sheet
3 == Word Processor
4 == Specifications
5 == Automatical DEMO
Q == QUIT

Please select number
```

If you want to start the DEMO 1-4, press <1> for Graph, <2> for Spread Sheet, <3> for Word Processor or <4> for Specifications.

If you want to return to the system mode, press <Q> or <q> key.

If you want to start the DEMO 1 and continue DEMO 2, 3, 4, press <5>.

1 = introduces Graph

2 = introduces Spread Sheet

3 = introduces Word Processor

4 = introduces Specifications

5 = introduces Auto matical DEMO

If you want to return to the menu, press <F5> during the demonstration.

BASIC

Your Sr. Partner uses the BASIC programming language.

Included in your Sr. Partner package is a BASIC Reference Guide. This manual provides complete programming instruction.

BASIC is contained on your System Disk. Once you have loaded BASIC into the computer's memory, it is ready to use.

OPERATION

> **NOTE:** Do not attempt to use BASIC until you have made a backup of your System Disk. See "Backing Up the System" on page 4-39.

To get into BASIC:

STEP 1 Insert your System Disk in Drive A.

SCREEN DISPLAYS:

```
A>
```

STEP 2 YOU TYPE:

```
basica
```

or

```
basic
```

STEP 3 and press <ENTER> After messages are displayed, the screen displays

```
Ok
```

Ok is the BASIC prompt. You are now ready to program.

To return to DOS

YOU TYPE:

```
system
```

and press <ENTER>

SCREEN DISPLAYS:

```
A>
```

to indicate DOS is ready for the next operation.

OPERATION

TURNING OFF THE SYSTEM

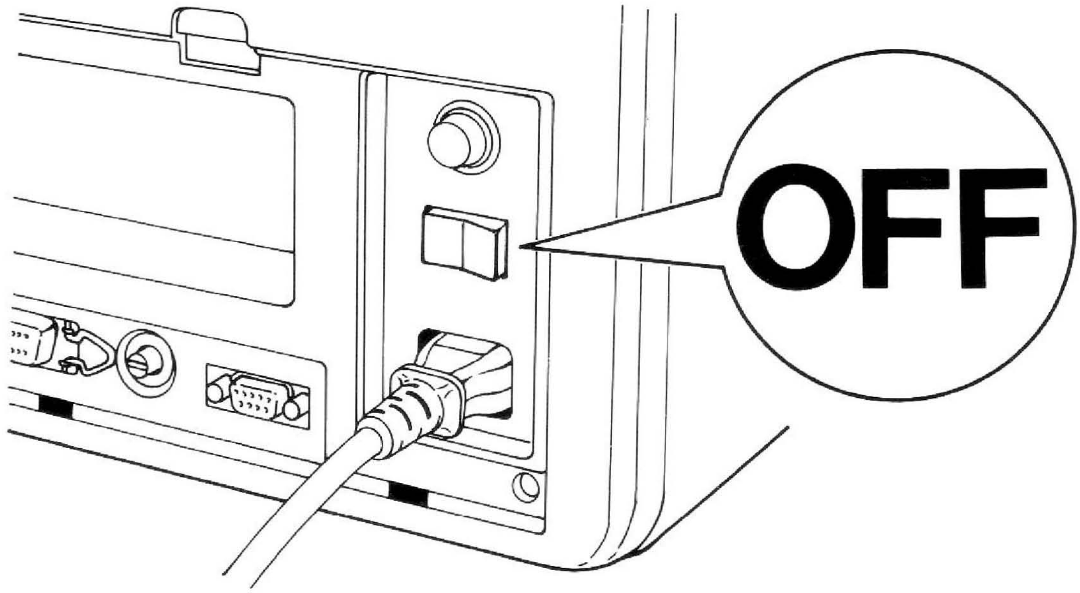


Fig. 4-30. THE MAIN POWER SWITCH IN OFF POSITION

Before you turn the computer off be sure to save any of the files you have been working on. Saving a file means transferring it out of RAM (where you have been working) to a disk for storage. RAM CONTENTS ARE LOST WHEN YOU TURN OFF THE COMPUTER.

It is also a good idea to make a backup of the file when you are finished working on it. Backups can be your salvation if the power accidently fails and you lose a file, or if a file is accidently erased.

Never turn the computer off when an In Use Indicator is lit. You can damage the recording head of your disk drive and you will almost certainly destroy the data on the disk.

If you wish to stop a printing procedure, press <Ctrl> <Num Lock>. You can press any key to resume printing. The power to the printer is controlled by the main power switch.

When you turn off the computer in BASIC, it will come on again in DOS.

It is a good idea to remove disks before turning off the computer.

In general, you can turn off the computer at almost any time. Just remember to SAVE YOUR FILES BEFORE YOU TURN OFF THE SYSTEM.

TRANSPORTING YOUR SR. PARTNER

The compact design of the Sr. Partner makes it a convenient traveling companion. Because all of the components fit together easily, you can pack up and move your computer quickly and efficiently.

Your computer is a delicate instrument, however, so take time to prepare the Sr. Partner for travel.

STEP 1 Be sure the main power switch is in the OFF position. Unplug the computer from the wall outlet. Disconnect the other end of the power cord from the system unit.

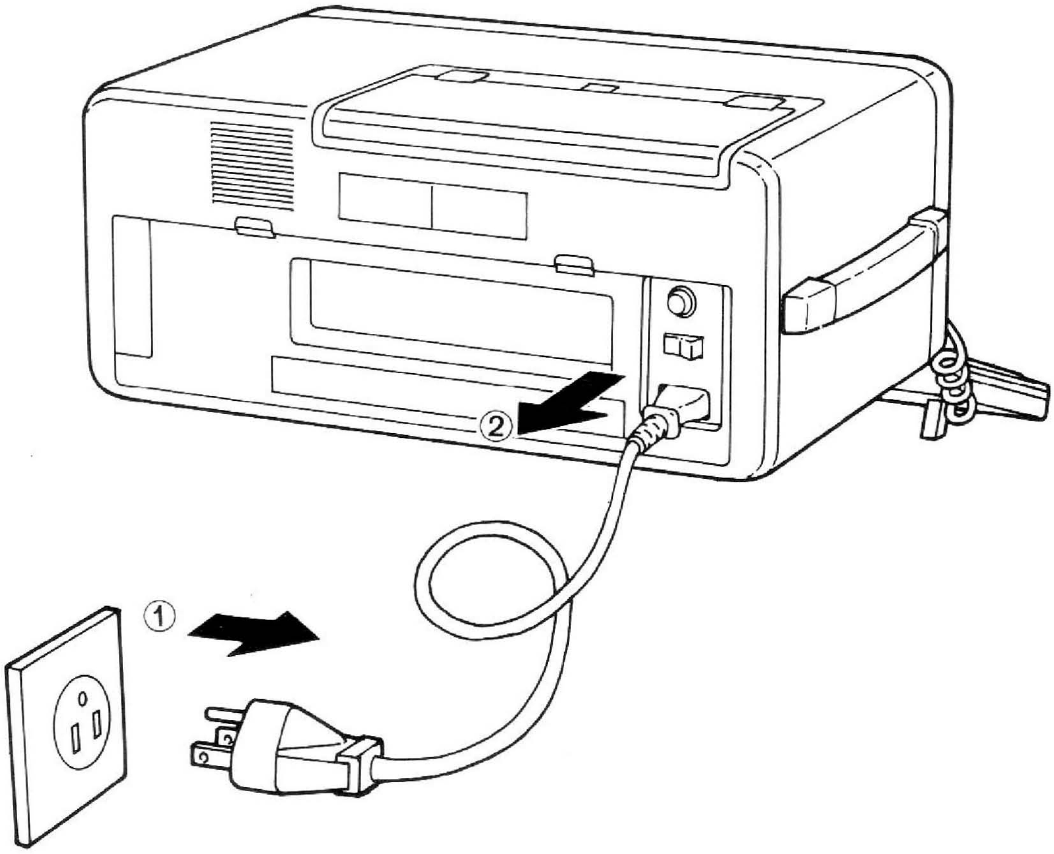


Fig. 4-31. DISCONNECTING POWER CORD

STEP 2 Store the power cord in the compartment in the rear panel. Attach the outer protective covering of the rear panel and lock it into place.

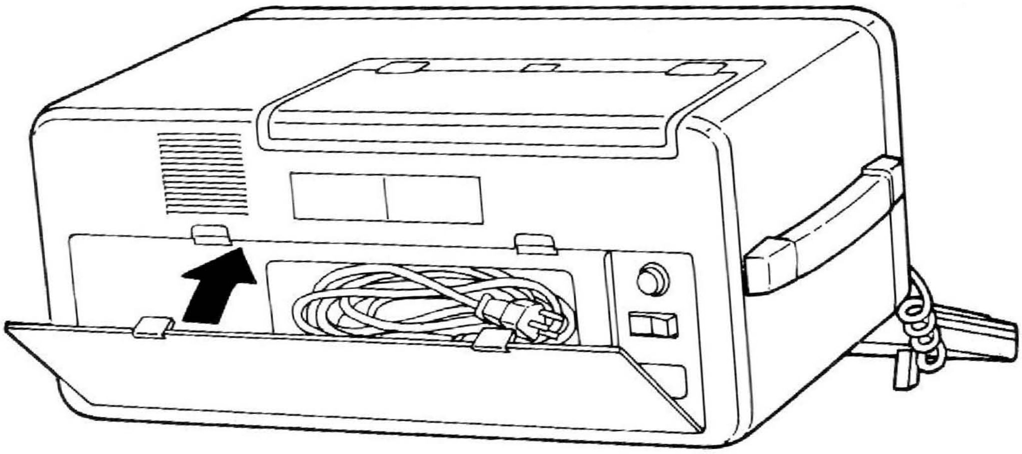


Fig. 4-32. POWER CORD STORAGE

OPERATION

STEP 3 Be sure there are no disks in the drive(s). Disk(s) can be damaged during transport. Insert the head protection sheet(s) that were in the drive(s) when you unpackaged your Sr. Partner.

Close and lock the drive doors.

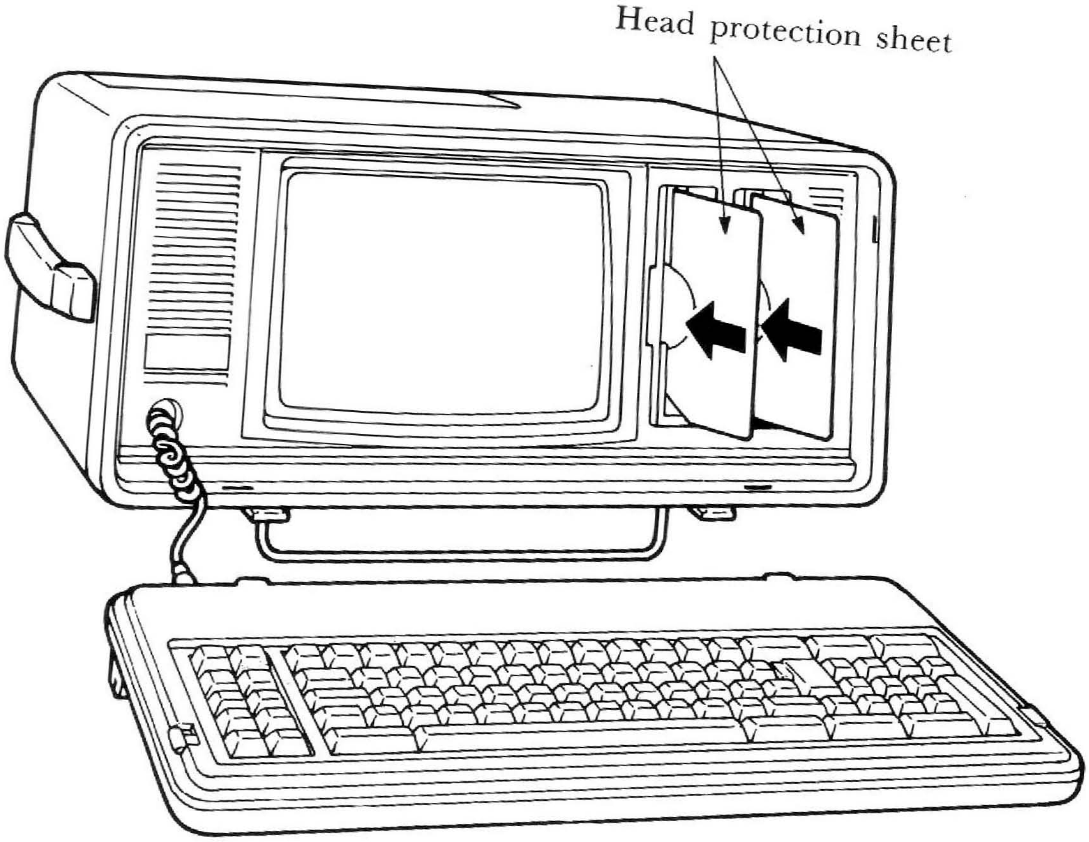


Fig. 4-38. PREPARING THE SR. PARTNER FOR TRANSPORT

OPERATION

STEP 4 Turn a wheel surrounding the connector off and unplug the coiled cord connecting the keyboard to the system unit.

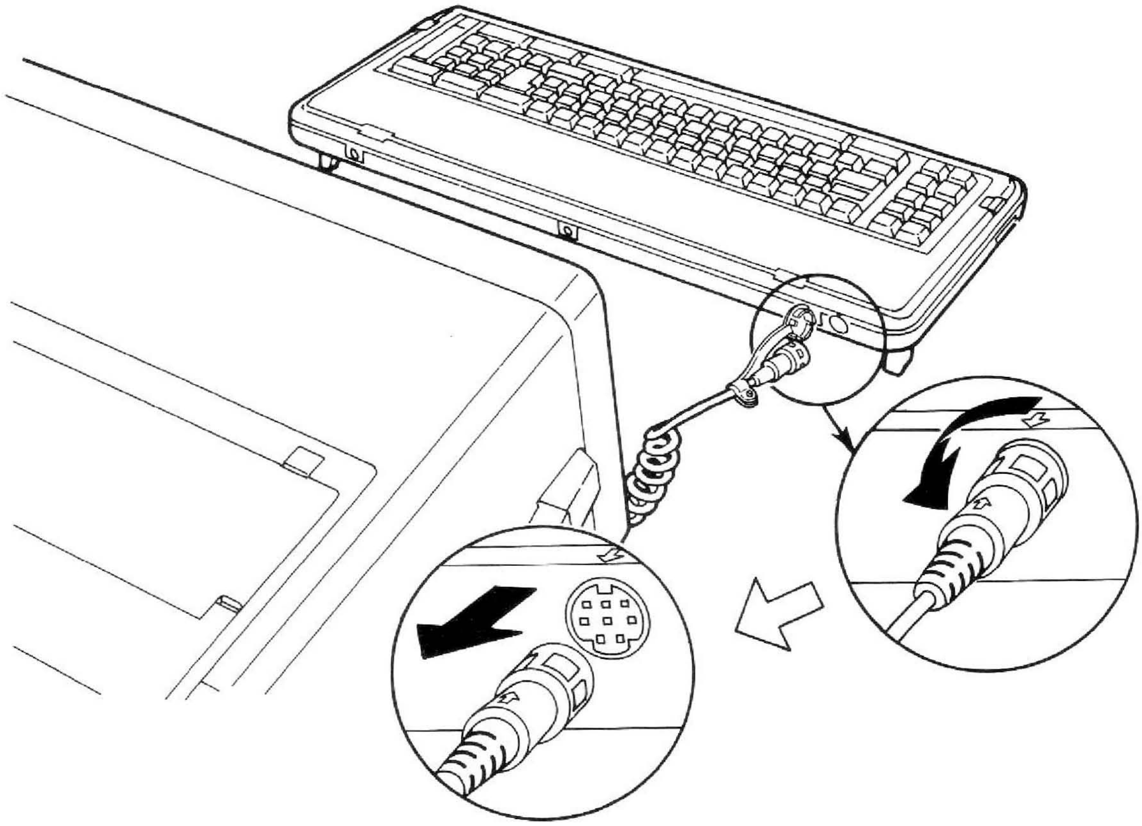


Fig. 4-34. UNATTACHING KEYBOARD

STEP 5 Slide the coiled cord into the system unit until the stop is reached. Replace the cap on the connector at the keyboard

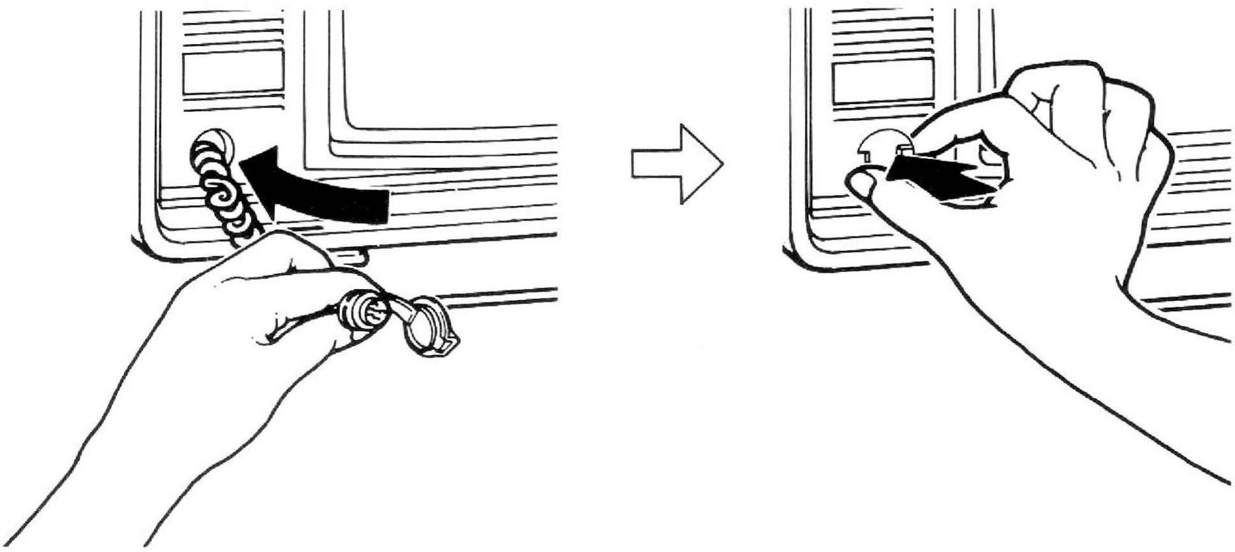


Fig. 4-35. REPLACING KEYBOARD CABLE

OPERATION

STEP 6 Attach the keyboard to the system unit. Fit the plastic tabs into the slots along the bottom edge of the system unit. Fold the keyboard up and lock it into place.

STEP 7 Fold the keyboard legs into place and use the tabs to lock them into position.

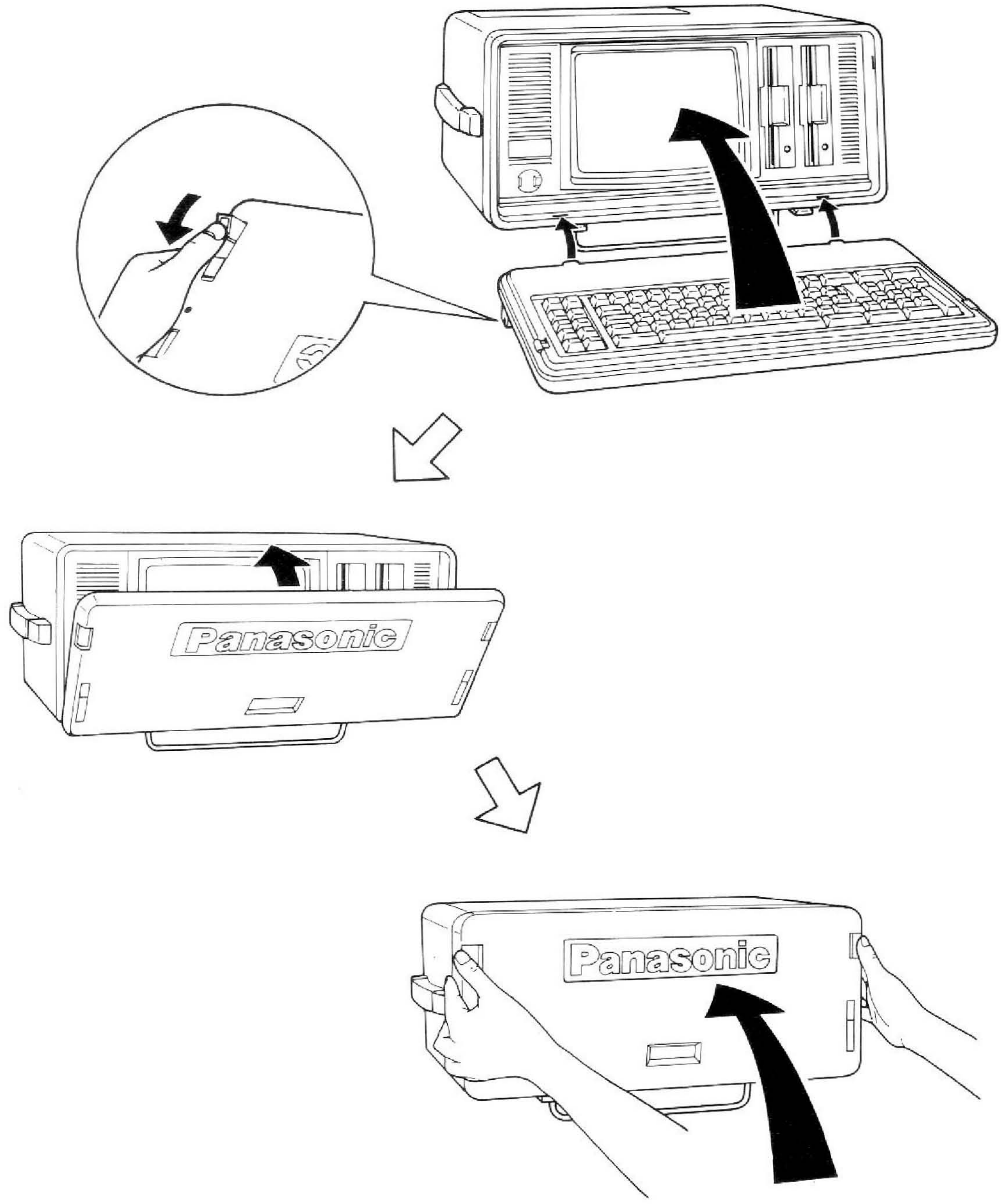


Fig. 4-36. HOW TO ATTACH THE KEY BOARD TO THE SYSTEM UNIT

STEP 8 Lift up the computer and set it on its "feet". Fold the metal rod over the keyboard unit.

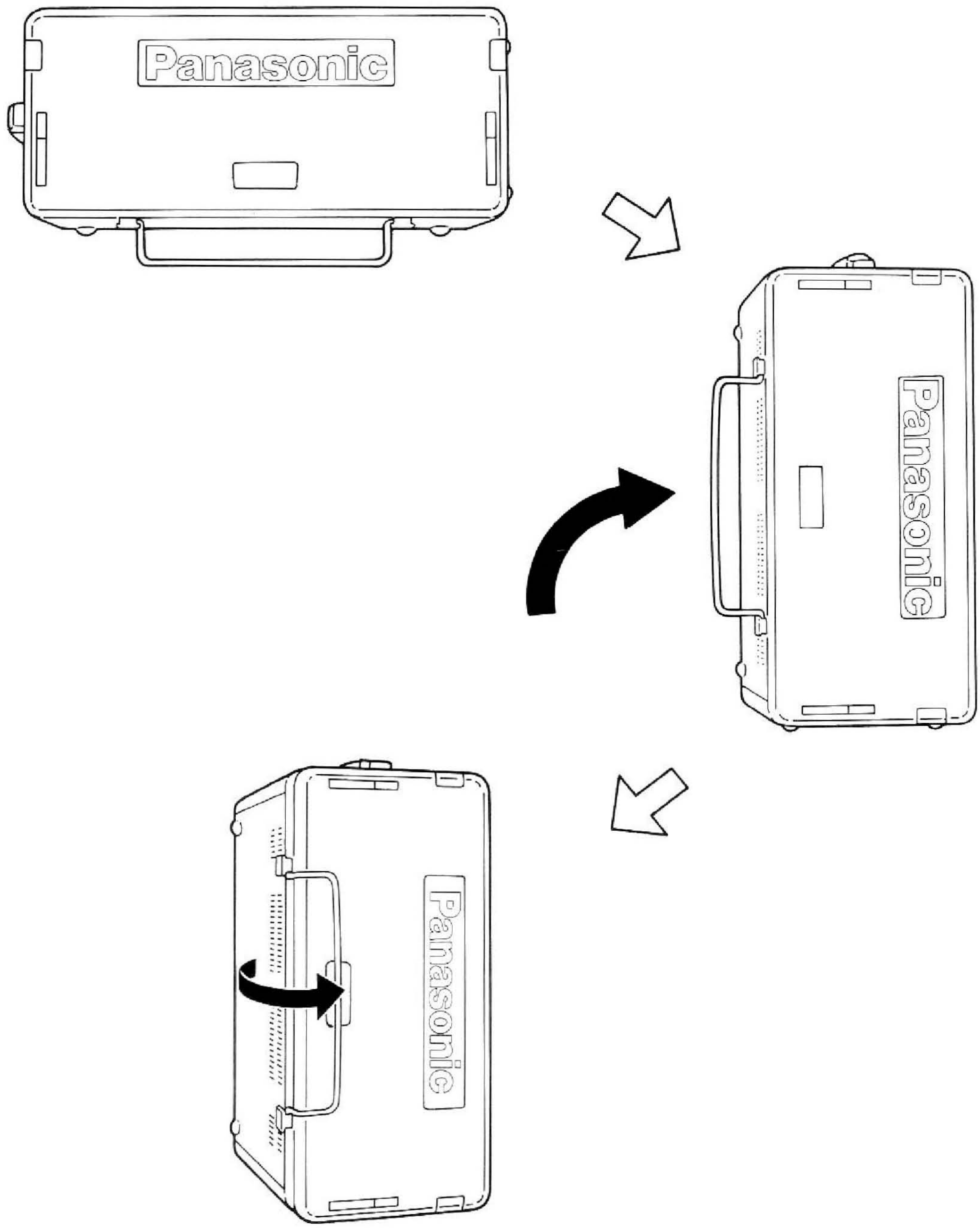


Fig. 4-37. FOLDING METAL ROD

Your Sr. Partner is new ready to travel! But treat your partner with respect. Avoid rough handling. After all, you want to be sure your computer will be ready and willing to get right to work when you reach your destination!

Detaching the Keyboard

The two ribbed rectangles at the right and left upper edge of the case are the locks for the keyboard unit. Using your thumbs, push these tabs toward the center to release the keyboard.

It may be necessary to lift the keyboard slightly to clear the indentations which hinge the keyboard at the bottom. Remove the keyboard and place it on a firm surface.

*** **WARNING** ***

> Never turn on the power or attempt to operate the computer while the keyboard is attached to the system unit. The keyboard blocks the ventilation slots necessary to cool the computer during operation.

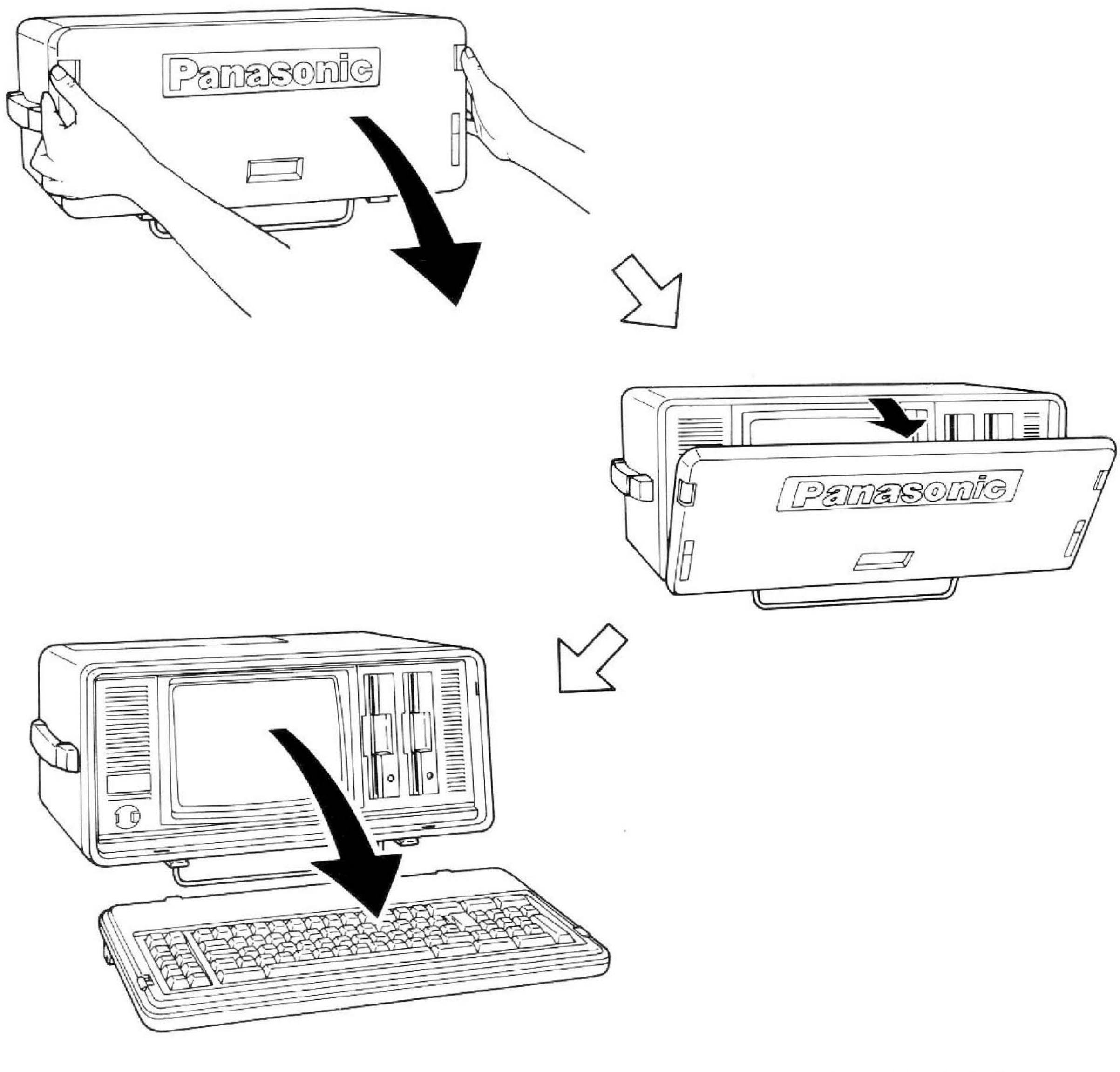


Fig. 4-38. HOW TO REMOVE THE KEY BOARD

MEMO

CHAPTER 5
USING DOS

USING DOS

INTRODUCTION

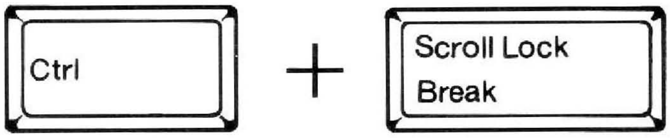
DOS is the disk operating system provided for you to use with your Sr. Partner. DOS is a group of programs that allow you to communicate with your computer. DOS programs provide a way to organize and use the information on your disks.

THE KEYBOARD

Your computer keyboard has some special keys to use with DOS. These keys enable you to perform DOS functions by just pressing keys on the keyboard.

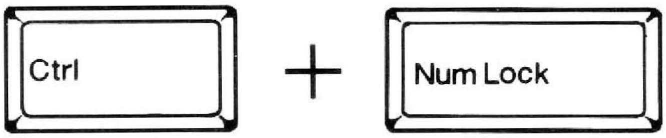
When more than one key is needed for a particular function hold down the first key and then press the second key.

Cancel the current operation.



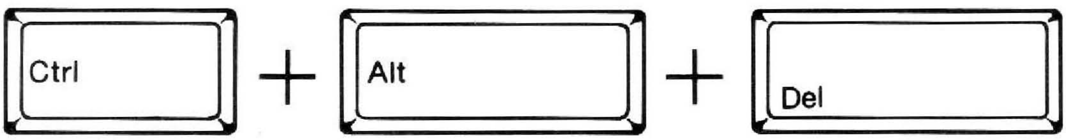
Stops the execution of the current command. DOS displays the prompt and you may enter another command.

Suspend the screen display.



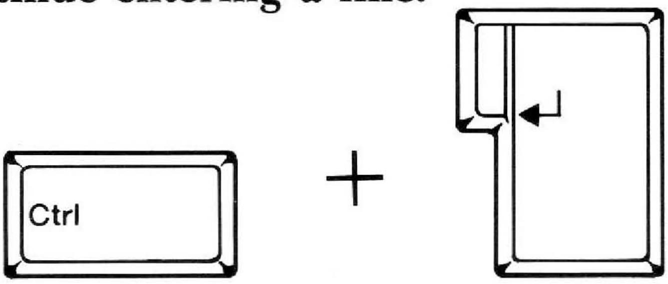
The information displayed on the screen is temporarily halted. When you are ready to continue receiving information, press any key.

Start DOS again.



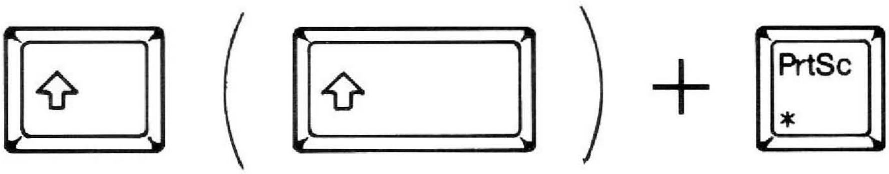
Be sure you have the DOS disk in drive A. Press and hold down the <Ctrl> and <Alt> keys and then press the <Del> key. This function will restart (known as reboot) DOS.

Continue entering a line.



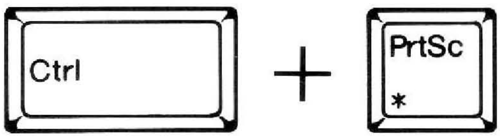
Moves the cursor to the next displayed line on the screen to continue entering the line you are typing.

Print what is on the screen.



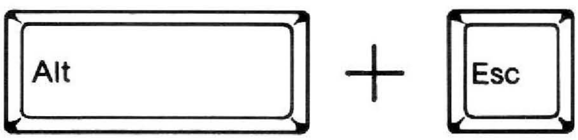
Produces a printed version (known as a hard copy) of what is on the screen.

Print what you are typing.



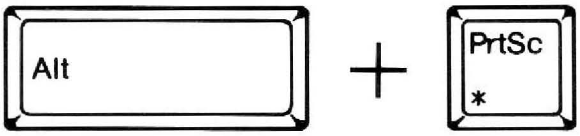
Produces a printed version (known as a hard copy) of each line as you type it. This echo feature can be turned off by pressing <Ctrl> <PrtSc> again.

Reset the internal printer



If the printer jams or fails, or when you have changed the linefeed dip switch of the internal printer (It is located under a roll paper.), press <Alt> <Esc>.

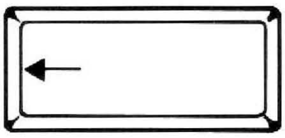
Switch printer.



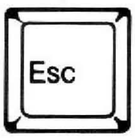
The internal printer is switched to an external parallel printer or vice versa.

Correct mistakes.

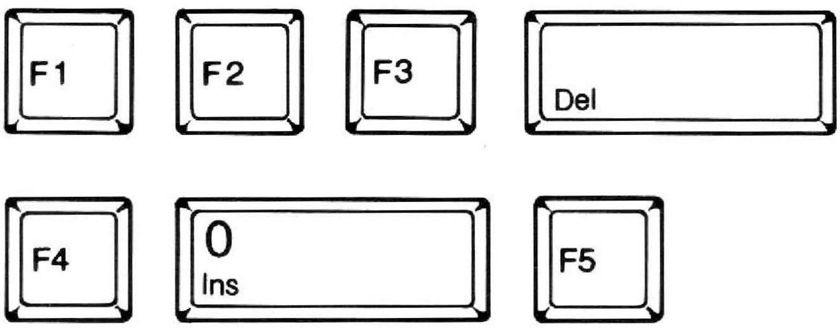
There are several ways to correct mistakes before you press <ENTER>.
The backspace key (located on the top row next to NumLock) deletes characters as it moves the cursor to the left.



The Esc key cancels the entire line. A backslash (\) is displayed on the screen and the cursor moves one line down. You may then type the command again.



See Chapter 8 EDLIN for information on the editing keys.



FILES

Naming Files

A filename can have a maximum of eight characters plus an optional extension. The extension consists of a period (.) followed by a maximum of three characters. The extension immediately follows the filename.
For example:

```
MYFILE.TST
```

If your filename has an extension always include it.

The valid characters in file names and extensions are:

A–Z	$	&	#	@	–
0–9	!	%	(	)	_
{	}	`	´		

Any other characters are invalid and the file name will be cut short (truncated).

Reserved Device Names

Reserved device names are used only for specific system input/output devices and cannot be used in any other context.

Reserved Name:	Device:
CON:	Keyboard input and screen output. To end CON: when using the keyboard as an input device, press F6 and then press <ENTER>.
AUX: or **COM1:**	Built-in RS232C (Serial) interface.

PRN: Internal printer

LPT1: External printer connected to the parallel port.

NUL: Dummy device for testing various applications. You can use NUL: when you do not want to create a file, but the syntax of the command requires an input or output file name.

Wildcard Characters

USING DOS

The characters * and ? may be used with filenames and their extensions to provide enhanced flexibility when issuing DOS commands.

The ? indicates that any valid character may occupy that position. For example:

```
DIR AB?.XYZ
```

Lists all directory entries beginning with AB, having any character next, with an extension of XYZ.

The DIR command might list the following files:

```
ABC.XYZ
AB1.XYZ
AB2.XYZ
```

The * indicates that any valid character may occupy that position and all remaining positions. For example:

```
DIR AB*.XYZ
```

Lists all directory entries beginning with AB and having any other characters with an extension of XYZ.

The DIR command might list the following files:

```
ABC.XYZ
ABCDE.XYZ
AB1234.XYZ
```

DIRECTORIES

Directory Structure

Directories provide a way to organize the files on each disk. The directory contains:

Names of the files.
Sizes of the files.
Dates the files were created.
Dates the files were updated.
Locations of the files on the disk.

USING DOS

You can organize your files into convenient categories by setting up separate directories.

Any one directory may contain entries for files and entries for other directories (called sub-directories). This method of file organization looks like an inverted tree and is called a hierarchical directory structure. The levels are defined as follows:

ROOT	The first level in the directory structure. This directory is automatically created when you format a disk and put files in it. It can contain 112 entries, either file names or sub-directory names.

SUB-DIRECTORIES A subsequent level in the directory structure. Sub-directories themselves can include the names of additional sub-directories. The sub-directories are actually files and are therefore not restricted in size, but are limited only by the available space on the disk. The sub-directory names are in the same format as file names. Files or directories with the same name can exist as long as each is in a different directory.

It is possible to "travel" around your directory structure. For example, it is possible to find any file in the system by starting at the root and traveling down through the sub-directories. You may also start where you are within the file system and travel up towards the root.

Figure 5-1 illustrates a typical hierarchical directory structure.

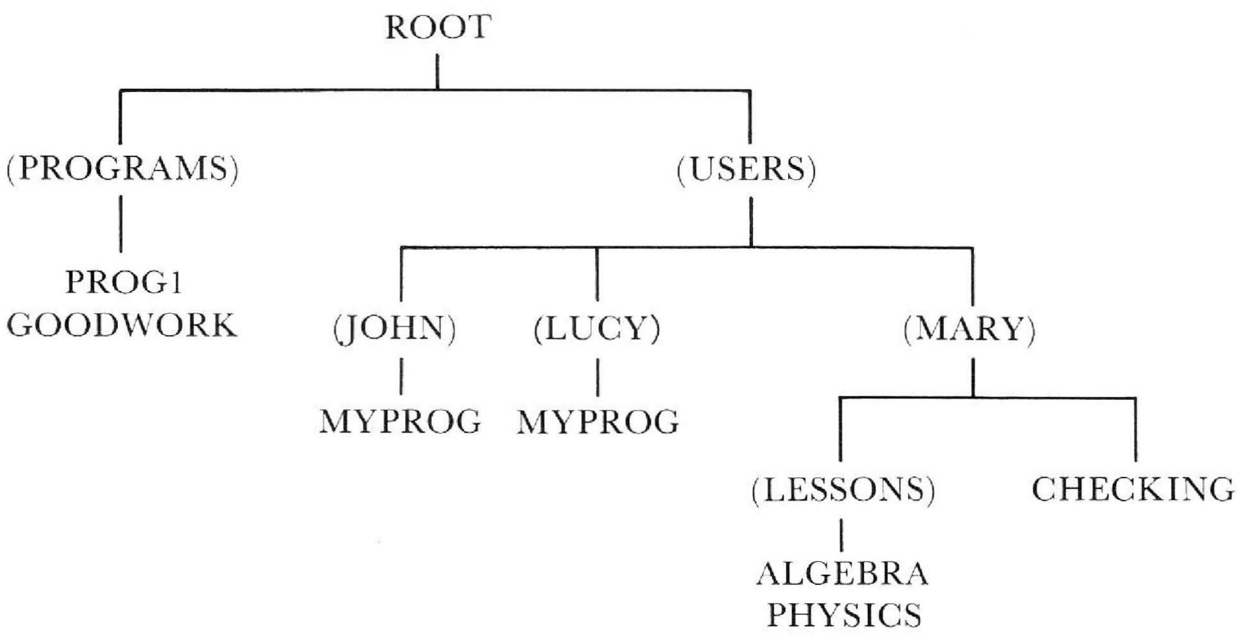


Fig. 5-1. A SAMPLE HIERARCHICAL DIRECTORY STRUCTURE
(Sub-directory names are in parenthesis)

In this example, two sub-directories of ROOT have been created. These include:

A directory of programs, named PROGRAMS.

A USERS directory containing separate sub-directories for the users of the system.

John, Lucy and Mary each have their own directories which are sub-directories of the USERS directory. John and Lucy have files in their directories, each named MYPROG. Notice that John's file is unrelated to Lucy's. Mary has a file named CHECKING as well as a sub-directory named LESSONS.

You can get a list of the files in Mary's LESSONS directory by typing:

```
DIR \USERS\MARY\LESSONS
```

Note that the backward slash mark (\) is used to separate directories from other directories and files.

To find out what files Mary has in her directory, you could type:

```
DIR \USERS\MARY
```

Current Directory

DOS will supply a default directory if you do not specify which directory a file is in. This default is the current directory. DOS will automatically use the root directory as the current directory until you change it with the CHDIR command. To find out what the current directory is or change the current directory use the CHDIR command.

For example, if your current directory is \USER\LUCY, when you type:

CHDIR<ENTER>

you will see:

A:\USER\LUCY

This is your current drive designation plus the current directory.

Each directory contains two special entries. They are special shorthand notations for the current directory and the parent directory (one level up) of the current directory.

• Notation to indicate the name of the current directory in all hierarchical directory listings. DOS automatically creates this entry when a directory is created.

• • The current directory's parent directory. If you type:

DIR • •

DOS will list the files in the parent directory of your working directory.

If you type:

DIR • •\ • •

DOS will list the files in the parent's PARENT directory.

See Directory Commands in this chapter for more information on CHDIR.

Paths To Files

To tell DOS where the files are located in the directory structure use a pathname to the file.

To find or create a file DOS requires the following information:

The drive.
The name of the file.
The name of the directory.

DOS searches the current directory automatically. If the file is not in the current directory you must tell DOS the path leading to the correct directory.

USING DOS

A pathname (denoted by *path*) is:

A sequence of directory names separated by backslashes (\) and a filename.

The syntax of *path* is:

[*directory*]\[*directory...*]\[*filename*]

If a pathname begins with a backslash, DOS searches for the file beginning at the root directory. Otherwise, DOS begins at the user's current directory, and searches downward from there.
For example:

If your current directory is MARY, and you want to find the file PHYSICS you can use either:

```
\USERS\MARY\LESSONS\PHYSICS
```

or

```
LESSONS\PHYSICS
```

In the first case the full path from the root was specified by the leading backslash. In the second case the path from the current directory was given.

Setting a Path

The PATH command allows you to specify additional paths for DOS to search if it does not find a command in the current directory.
For example:

> If you are in a working directory named \COM\PROG, and all DOS external commands are in \COM, you must tell DOS to choose the \COM path to find the DISKCOPY command. The command
>
> PATH \COM
>
> tells DOS to search in your working directory and the \COM directory for all commands. You only specify this path once to DOS during your terminal session. DOS will now search in \COM for the external commands.

USING DOS

To check what the current path is, type the word PATH and the current value of PATH will be displayed.

For more information on the DOS command PATH, refer to Chapter 6 Command Reference.

Table of Directory Commands

COMMAND	PURPOSE	SYNTAX
CHDIR	Displays working directory; changes directories	CHDIR [[*d:*] *path*]
MKDIR	Makes a new directory	MKDIR [*d:*] *path*
RMDIR	Removes a directory	RMDIR [*d:*] *path*
TREE	Displays directory structure	TREE [*d:*]

Directory Commands

Creating a Directory

To create a sub-directory in your current directory, use the MKDIR (Make Directory) command.

For example, to create a new directory named MOREDIR under your current directory, simply type:

```
MKDIR  MOREDIR
```

After this command has been executed, a new directory will exist in your tree structure under your current directory. To make directories anywhere in the tree structure specify MKDIR and then a pathname. DOS automatically creates the . and .. entries in the new directory.

USING DOS

Deleting a Directory

To delete a directory in the tree structure, use the RMDIR (Remove Directory) command.

For example, to remove the directory MOREDIR from the current directory, type:

```
RMDIR  MOREDIR
```

The directory MOREDIR must be empty except for the . and .. entries before it can be removed. This prevents you from accidentally deleting files and directories. To remove the \USERS\JOHN directory make sure that it has only the . and .. entries, then type:

```
RMDIR  \USERS\JOHN
```

To remove all the files in a directory (except for the . and .. entries), type DEL and then the pathname of the directory. For example, to delete all files in the \USERS\JOHN directory, type:

```
DEL  \USERS\JOHN
```

You cannot delete the . and .. entries. They are created by DOS as part of the hierarchical directory structure.

Setting the Current Directory

To change from your current directory to another directory use the CHDIR (Change Directory) command and supply a pathname. For example:

```
CHDIR \USERS
```

changes the current directory from \USERS\JOHN to \USERS. The command CHDIR . . will always put you in the parent directory of your working directory.

Displaying the Directory Tree

To see a report describing the directory structure of a disk, use the command TREE. The report includes all the directory paths and, if the /F parameter is used, the names of all files in each sub-directory. For example:

```
TREE B:/F
```

shows all the directory paths on drive B and the names of the files in each sub-directory.

MEMO

CHAPTER 6
COMMAND REFERENCE

DOS COMMANDS

INTRODUCTION

This chapter explains the ways to use the Disk Operating System (DOS) through DOS commands. These commands provide file handling capabilities and program execution, among other things.

The DOS commands:

Manipulate files—compare, copy, display, erase, and rename.

Manipulate disks—format, compare, and copy.

Run programs—execute DOS system programs (i.e. EDLIN or DEBUG) as well as your programs.

List directories.

Enter date or time or remarks.

Set mode of a printer, screen formats and the serial interface.

Route printer output—direct printer output to the serial interface.

Convert EXE files to COM type files.

Transfer DOS to another disk.

Request the system to wait.

TYPES OF DOS COMMANDS

Internal Commands:

These commands are built into the command processor. They execute immediately.

External Commands:

These commands are not built into the command processor but reside on disks as program files. In order to execute them they must be read from the disk which must be in the disk drive in order for DOS to read it.

Your Own Commands:

DOS COMMANDS

Any filename with the extension of .COM or .EXE is considered to be an external command. Therefore, you can create your own commands and add them to the system by using these extensions. Programs such as JUMP.COM or HOP.EXE are treated as external commands. When entering an external command, do not include the filename extension.

SYNTAX NOTATION

The following notations show how the DOS commands should be formatted:

Any words shown in CAPITAL letters are keywords and must be entered exactly as illustrated. DOS will automatically convert all keywords to uppercase. Thus, you may enter the keywords in any combination of upper and lower cases.

Commas (,), equal signs (=), question marks (?), slashes (/), or colons (:) must be entered exactly where shown. The only punctuation not needed are the square brackets ([ ]).

Any items shown in lowercase *italic* letters indicate where to supply variable information. When you see *filename*, replace *filename* with the name of your file.

An elipsis (...) indicates the repetition of an item as many times as needed.

Square brackets ([ ]) indicate optional information. To include the optional information, type in just the information inside the brackets—not the brackets themselves.

PARAMETERS

Parameters are variable or constant items used to specify limits in DOS command statements. There are both optional and required parameters. In some cases DOS will provide a default value for a parameter not included. (A discussion of the default parameters appears in the detailed descriptions of each DOS command). The following four parameters, whether entered or defaulted, are necessary for every DOS command:

d: Drive. This letter indicates when you should specify a drive. A letter followed by a colon tells DOS which disk drive to use. If you omit this parameter, the default drive is assumed. For example:

A: represents the first drive on the system.
B: represents the second drive.

DOS COMMANDS

filename A 1–8 character filename, composed of valid characters. The valid characters are:

A–Z, 0–9,
$ & # @ - ! % ( ) _ { } ` ´

Any other character is considered to be a delimiter and will truncate the filename.

.ext Filename extension is optional and is made up of a period and 1–3 characters. As with filenames, only valid characters are allowed and filename extensions must immediately follow the filename.

filespec [*d:*] *filename* [*.ext*]

The wildcard characters (* and ?) are valid in a filename or extension.

path [*directory*] [\ *directory*[...]]
See page 5–12.

Reserved Device Names

DOS reserves the following names as system devices:

Reserved Name	Device
CON:	Console, keyboard or screen. When **CON**: is used as an input device, you may use the F6 function key and then press <ENTER> to generate an end-of-file marker which terminates **CON:** as an input device.
AUX: or **COM1:**	Built-in serial (RS232C) port.
NUL:	Nonexistent or dummy device. This is used when testing. When **NUL:** is used as an input device an immediate end-of-file mark is generated. When **NUL:** is used as an output device, the write operations are simulated but no actual data is written.
LPT1: or **PRN:**	Parallel printer, used as an output device only.

NOTE: The reserved device names can substitute for a filename. If a filename extension or drive specifier is entered with these device names, it will be ignored. The colon at the end of the reserved device name is optional.

Wildcard Characters

There are two special characters which allow for greater flexibility with DOS commands. The "?" and the "*" allow for special replacement characters or default characters to appear in a filename or its extension.

The "?" Character:

A "?" in a filename or in a filename extension means that any character can fill that position. For example:

```
DIR   AB?DE.FGH
```

This command would list all the directory entries on the default drive with the filenames that have the first two characters AB, any other character, the last two characters DE and have an extension of FGH. The position occupied by the "?" can contain any valid character. If the above command were entered, here is an example of what might be listed:

```
ABCDE.FGH
AB3DE.FGH
AB#DE.FGH
```

The "*" Character:

An "*" in a filename or in a filename extension means that any character can fill that position and all the remaining positions of either the filename (if placed in the filename) or the filename extension (if placed in the extension). If placed in the filename, the substitution stops at the period of the extension name. For example, the command:

```
DIR   AB*.EFG
```

would list all directory entries on the default drive with filenames that begin with AB, followed by any other valid character or characters and have a filename extension of EFG. The filename may be anywhere from 2–8 characters long. If the above command were entered, here is an example of what might be listed:

```
ABC.EFG
ABCDE.EFG
ABXYZ123.EFG
```

Examples:

To list all the filenames on your default drive that begin with FINANCE, enter:

```
DIR  FINANCE.???
```
or
```
DIR  FINANCE.*
```

To list all the filenames on Drive A that end with the filename extension of DAT, enter:

```
DIR  A:????????.DAT
```
or
```
DIR  A:*.DAT
```

To list all the filenames on Drive A that begin with the letters FIN and have filename extensions that begin with the letter C, enter:

```
DIR  A:FIN?????.C??
```
or
```
DIR  A:FIN*.C*
```

TABLE OF COMMANDS

Command	Purpose	Syntax
ASSIGN	Specify the disk drive	ASSIGN [*x*=*y*[...]]
BACKUP	Back up files from a hard disk to floppy disk.	BACKUP *d:*[*path*] [*filename* [*.ext*]] *d:* [/S][/M][/A] [/D: *mm-dd-yy*]
BREAK	Check for a control break.	BREAK [ON|OFF]
CHDIR	Change or display current directory	CHDIR [[*d:*] *path*] or CD [[*d:*] *path*]
CHKDSK	Report on disk status.	CHKDSK [*d:*] [*filename*] [/F] [/V]
CLS	Clear display screen.	CLS
COPY	Copy files	COPY [/A] [/B] [*d:*] *filename* [*.ext*] [/A] [/B] [*d:*] [*filename*[*.ext*[[[/A] [/B] [/V] or COPY [/A] [/B] [*d:*] [*path*] *filename*[.ext] [/A] [/B] [+[*d:* [[*path*]*filename*[*.ext*[[/A] [/B]...] [*d:*] [*path*] [*filename*[*.ext*]] [/A] [/B] [/V]
CTTY	Change input and output devices.	CTTY *device-name*
DATE	Set and display date.	DATE [*mm-dd-yy*]

DOS COMMANDS

Command	Purpose	Syntax
DEL	See "Erase command"	————
DIR	List of directory entries.	DIR [*d:*] [*path*] [*filename*[*.ext*]] [/P][/W]
DISK-COMP		DISKCOMP [*d:*] [*d:*] [/1] [/S]
DISK-COPY	Copy disk contents.	DISKCOPY [*d:*] [*d:*] [/1]
ERASE	Delete files.	ERASE [*d:*] [*path*] [*filename*[*.ext*]] or DEL [*d:*] [*path*] [*filename*[*.ext*]]
EXE2BIN	Convert .EXE files to .COM files.	EXE2BIN [*d:*] [*path*] *filename*[*.ext*] [*d:*] [*path*] [*filename*[*.ext*]]
FC	Compare file contents	FC [*d:*] [*path*] [*filename*[*.ext*]] [*d:*] [*path*] [*filename*[*.ext*]]
FDISK	Process the partition on the hard disk.	FDISK
FIND	Output lines containing specified string.	FIND [/V] [/C] [/N]*string* [[*d:*] [*path*] *filename* [*.ext*]...]
FORMAT	Initialize the disk.	FORMAT [*d:*] [/S] [/V] [/B] [/1] [/8]
GRAPHICS	Print graphic screen.	GRAPHICS

DOS COMMANDS

Command	Purpose	Syntax
MKDIR	Create sub-directory.	MKDIR [*d:*]*path* or MD [*d:*]*path*
MODE	Set display mode.	MODE LPT#*:*[*n*] [,[*m*] [,P]] or MODE *n* or MODE [*n*],*m*[,T] or MODE COM*n:* *baud*[,*parity*[,*databits* [,*stopbits*[,P]]]] or MODE LPT#*:*=COM*n*
MORE	Send one screen of data and then pause.	MORE
PATH	Set directory path.	PATH [*d:*]*path*[[; [*d:*]*path*]...]
PRINT	Set up print queue.	PRINT [[*d:*] [*filename*[.*ext*]] [/T] [/C] [/P]...]
PROMPT	Set new system prompt.	PROMPT [*prompt-text*]
RECOVER	Recover files from damaged disk.	RECOVER [*d:*] [*path*]*filename*[.*ext*] or RECOVER *d:*

DOS COMMANDS

Command	Purpose	Syntax
RENAME	Rename a file.	REN[AME] [*d:*] [*path*]*filename*[*.ext*] *filename*[.ext]
RMDIR	Remove sub-directory.	RMDIR [*d:*]*path* or RD [*d:*]*path*
RESTORE	Restore files from floppy disk to a hard disk.	RESTORE *d:*[*d:*] [*path*] [*filename*][*.ext*][/S][/P]
SET	Insert environment strings.	SET [*name*=[*parameter*]]
SHIP	Protect data on the hard disk	SHIP
SORT	Sort data.	SORT [/R] [/+*n*]
SYS	Transfer system files.	SYS *d:*
TIME	Set and display time.	TIME [*hh:mm:ss.xx*]
TREE	Display directory paths and list files in sub-directory.	TREE [*d:*][/F]
TYPE	Display file contents.	TYPE [*d:*] [*path*] *filename*[*.ext*]
VER	Display DOS version number.	VER
VERIFY	Verify successful write to disk.	VERIFY [ON|OFF]
VOL	Display volume information.	VOL [*d:*]

DOS COMMANDS

COMMAND DESCRIPTIONS

This section provides a detailed description of the syntax, purpose, type, and comments for each DOS command. Some information is common to all DOS commands. The following notations apply to all commands:

Most commands are followed by one or more parameters.

Parameters and commands may be entered in either upper or lower case (DOS will convert all characters to upper case).

Commands and parameters must be delimited by either a space, comma, semicolon, equal sign or the tab key. You need not use the same delimiter within one command. For example, you could enter:

```
COPy oldfile.txt;newfile.txt
```

A drive and filename must be separated by colon (:) and a filename and .ext must be by period (.).

A command takes effect only after you press the <ENTER> key.

Drives may be referred to as source and target drives. A source drive is the drive from which you will get information and a target drive is the drive to which you will send information.

The prompt produced by the command processor is the default drive designation letter, plus >. For example: A>.

The DOS editing keys and control keys described in Chapter 5 may be used while entering commands.

Filename extensions are optional when you create or rename a file; however, if a file has a filename extension, you must use it when referring to that particular file.

To end a command while it is executing, press <Ctrl><Break>. <Ctrl><Break> is only recognized while the system is reading from the keyboard or writing to the screen; therefore, the command may not end immediately after you press <Ctrl><Break>.

Pressing the <ENTER> key begins the command. Until you press this key you may change whatever you typed onto the screen and nothing will have happened.

Device names and wildcard characters are not permitted in a command name. You may, however, use them in command parameters.

To suspend the display of output on the screen, you can press the <Ctrl><NumLock> key. To restart the display, press any other key.

BREAK

Internal

Syntax: **BREAK** [ON|OFF]

Purpose: Checks for a control break whenever a program requests that DOS perform any functions.

Comments: Usually DOS performs a check for a <Ctrl><Break> being entered at the keyboard during screen, keyboard, printer, or auxiliary device operations. This command specifies when DOS should check for a <Ctrl> <Break>.

```
BREAK ON
```

DOS checks for a <Ctrl><Break> any time a program requests any DOS function.

```
BREAK OFF
```

DOS checks for a <Ctrl><Break> only during screen, keyboard, printer, or Asynchronous Communication Adapter operations.

To check the current state of <Ctrl><Break> checking enter BREAK with no parameter.

CHDIR (Change Directory)

Internal

Syntax: **CHDIR** [[*d:*]*path*]

or

CD [[*d:*]*path*]

Purpose: Changes the DOS current directory or displays the current directory path of a drive.

Comments: If you do not indicate a specific drive, DOS assumes the default drive. If you use CHDIR or CD with no parameters, or with only a *d:* parameter, DOS shows the current directory path of the named or default drive.

Examples:

```
CHDIR \
```

Changes the current directory of the default drive to its root directory.

```
CD B:\SUB1\SUB2
```

Changes Drive B's current directory to the path "root→SUB1→SUB2".

```
CD B:SUB3
```

Changes Drive B's directory to the current directory path plus SUB3. If the previous example had been used, the resultant path would be:

root→SUB1→SUB2→SUB3

DOS COMMANDS

CHKDSK

External

Syntax: **CHKDSK** [*d:*][*filename*][/F][/V]

Purpose: Produces a disk and memory status report.

Comments: CHKDSK first shows any error messages with the status report following. For a listing of error messages see Appendix B.

CHKDSK does not automatically correct errors. In order to correct an error, you must use the /F (fix) parameter. Without the /F parameter, CHKDSK will not actually make the corrections on the disk.

CHKDSK will show the number of non-contiguous areas occupied by the file or files if you include a filename, but CHKDSK will look only in the current directory for these files.

CHKDSK will provide more detail about the errors if you use the /V parameter.

CHKDSK always assumes that the disk to check is in the named drive. On a single disk-drive system, it is important that the specified drive is not the default drive, unless you wish to check the DOS disk itself.

It is a good idea to run CHKDSK occasionally for each disk to be sure of the integrity of your file structures.

Examples:

A sample status report:

```
Volume DISK12   Created DEC 16, 1984   11:00

    189710 bytes total disk space
      8944 bytes in 2 hidden files
       512 bytes in 1 directories
     26014 bytes in 5 user files
    134244 bytes available on disk

    196608 bytes total memory
    170736 bytes free
```

CLS (Clear Screen)

Internal

Syntax: **CLS**

Purpose: Clears the display screen.

Comments: The screen is cleared unless foreground and background colors have been selected (See Chapter 11). If colors were selected, they will remain unchanged.

COPY

Internal

Syntax: **COPY** [/A] [/B] [*d*:] [*path*]*filename*[*.ext*] [/A] [/B] [*d*:] [*path*] [*filename*[*.ext*]] [/A] [/B] [/V]

or

COPY [/A] [/B] [*d:*] [*path*]*filename*[*.ext*] [/A] [/B] [+[*d:*] [*path*]*filename*[*.ext*] [/A] [/B]...] [*d*:] [*path*] [*filename*[*.ext*]] [/A] [/B] [/V]

Purpose: Copies files to another disk and gives the copy a different name if desired.

COPY will also copy files to the same disk. The copies must have different names unless a different directory is indicated.

You may combine files (Concatenate) while performing the copy.

DOS COMMANDS

COPY may also be used to transfer data between any of your system devices.

Comments: The source file is the first file named, and the target file is the second. If the second parameter has no filename, files are copied without a name change.

The wildcard characters ? and * may be used in the filename and in the extension of both the duplicate and the original files. If you use an * or ? in the source filespec, the name of the files will be shown as the files are being copied.

/A — If you use /A with a source files specification, the file is treated as a text (ASCII) file. COPY copies the content of the file until it finds the first EOF (End of File) character, Ctrl-Z (1AH). The rest of the file is not copied. If you use /A with a target file specification, COPY adds Ctrl-Z character as the last character of the file. /A is the default when concatonation (Option 3) is specified.

/B — If you use /B with a source file specification, COPY copies the entire file. If you use /B with a target file specification, COPY doesn't add Ctrl-Z character as the last character. /B is the default when concatination is not specified.

> **NOTE:** /A or /B takes effect on the file preceding it and on all remaining files until another /A or /B is found.

/V — Verify Option:

It indicates to DOS to verify that the sectors are written on the target disk correctly. This option provides the ability to verify that critical data has been properly recorded even though errors are rare. The COPY command will run more slowly because of the overhead of verification.

If you have specified the VERIFY ON command previously, /V is unnecessary.

There are three format options for the COPY command:

Option 1—Source and Destination Files Have The Same Names

The copied file will have the same filename and extension as the source file. For example:

COPY [*d:*][*path*]*filename* [.ext]

or

COPY [*d:*][*path*]*filename* [*.ext*] *d:*[*path*]

The first example shows the file is copied to the current directory of the default drive. The second example specifies the target drive and/or directory.

The copied file will have the same filename as the source file because we did not indicate a name for the second file. The source drive and the target drive cannot be the same unless different directories were either implied or specified; otherwise, the copy is not allowed. For example:

```
COPY *.* B:
```

If the default drive is A, this command copies all the files from the default Drive A to Drive B. There are no changes in the filenames or in the extensions. The filenames are shown while the files are copied.

If the files are in different sub-directories they can be on the same disk. For example:

```
COPY B:\PROG1 B:\SUB1
```

This command copies the file PROG1 from the root directory on Drive B to the directory path SUB1 on the same drive. Both the copy and the original files have the same filenames. This example assumes that directory SUB1 exists on Drive B. If the SUB1 directory is not there then the file PROG1 will be copied into a file named SUB1 in the root directory of Drive B. If the second parameter names a directory that is there, the file (or files) will be copied into that directory with the same filename. If the second parameter names a directory that is not there, it will be considered a filename.

Option 2—Source and Destination Have Different Names

The copied file will have a different name from the file that is copied. For example:

COPY [*path*]*filename* [*.ext*] *filename*[*.ext*]

or

COPY [*path*]*filename*[*.ext*] *d:*[*path*]*filename* [*.ext*]

The first example shows the first filename is copied and renamed to the second filename. Because a drive was not specified, the default drive was used. The second example shows the file is copied and renamed also, but a target drive was named. The source drive and the target drive may be the same because the name of the file was changed. The current directory may be different or the same. For example:

```
COPY  PROG1.XYZ  B:*.ABC
```

The file PROG1.XYZ from the disk in default Drive A is copied to Drive B, renaming the copy PROG1.ABC. We used the current directory of each drive.

Reserved device names may be used for the copy operation. For example:

```
COPY CON: fileA
COPY CON: AUX:
COPY CON: LPT1:
COPY AUX: LPT1:
COPY AUX: CON:
COPY filel  CON:
COPY file2  AUX:
COPY file3  LPT2:
```

Any variation can use NUL:. The colon is optional when indicating a reserved device name.

DOS COMMANDS

To use COPY to place what you type into a file:

```
A>COPY CON: file1
Type whatever and then press Enter.
Continue to type and then press Enter.
•
•
•
Type your last line and press Enter.
Press F6 and then press Enter.
```

The COPY operation will end and save the data entered. In this example, the data is saved in a file named "file1".

<F6> has been assigned the value Ctrl-Z. If you have altered the meaning of F6, substitute the key you have assigned to <Ctrl><Z> for <F6>.

Option 3—Combine Files (Concatenate)

You may combine files while copying. The system combines two or more files into one file by appending the other files to the first file.

The result file directory will contain the current date and time.

The number of files copied indicated in the message is the number of result files created.

To concatenate files, name any number of source files and separate them by plus (+) signs in the COPY command. This is the format to use:

> COPY [/A] [/B] [*d:*] [*path*]*filename*[*.ext*]
> [/A] [/B] [+[*d:*] [*path*]*filename*[*.ext*] [/A] [/B]...]
> [*d:*] [*path*]*filename*[*.ext*] [/A] [/B] [/V]

DOS COMMANDS

For example:

```
COPY  A:MY1.XYZ+B:Y2.XYZ+MY3.TXT COMBO.TXT
```

This command creates a file called COMBO.TXT on your default drive. The combination of MY1.XYZ, B:Y2.XYZ, and B:MY3.TXT is placed into COMBO.TXT.

If you do not specifically name a result filename, the additional source files are appended at the end of the first file, putting the result in the first file. For example:

```
COPY  ABC.TXT+XYZ.TXT
```

COPY appends XYZ.TXT to the end of ABC.TXT and puts the result in ABC.TXT.

The filenames of the files to be combined and the result file can use the wildcard characters ? and *.

For example:

COPY *.TXT RESULT.PRN

All files having *.TXT are concatenated into one file called RESULT.PRN.

COPY *.TXT+*.LST RESULT.PRN

All files having *.TXT and *.LST are combined into one file called RESULT.PRN.

COPY *.TXT+*.DAT *.PRN

Each file with *.TXT is combined with the .DAT files of the same name. The result file has the same name with the extension .PRN. File FILE1.TXT is combined with FILE1.DAT to form FILE1.PRN. FILE2.TXT is combined with FILE2.DAT to form FILE2.PRN. In this instance since multiple files are to be created, only one file from each of the source filespecs is used to create a destination file.

It is possible to enter a COPY command to combine files where one of the source files has the same name as the target file but it is not easy to detect this. For example:

COPY *.TXT BIG.TXT

If BIG.TXT was already there, there would be an error which would not be detected, until it was time for BIG.TXT to be appended. BIG.TXT by this time may already be altered.

In this case, COPY does the following:

Each input file name is compared with the target filename. If the names are the same that one input file is ommitted. The message "Content of destination lost before copy" is shown and copying goes on normally. This permits summing files, with a command like:

COPY BIG.TXT+*.TXT

This command adds all *.TXT files, except for BIG.TXT itself, to BIG.TXT. The error message is not displayed because this is a real physical append to BIG.TXT.

Special Cases of Concatenation:

```
COPY   B:MYFILE.ASM+
```

Copies the file MYFILE.ASM to the default drive and supplies a new date and time.

```
COPY   B:MYFILE.ASM+,,   B:
```

Changes the date and time, leaving the file in place. The two commas are needed to denote the end of the source filename, because COPY usually expects seeing another filename after the plus (+) sign.

If wildcard characters (? or *) are used, then all of the matching files will be added together into the first filename that matches. For example:

```
COPY   B:*.*+,,   B:
```

Does not update the dates and times of all files on Drive B. All of Drive B's files will be added into a single file which will replace the first file found on Drive B.

> **NOTE:** COPY is considered successful if at least one, but not necessarily all, of the name source files is found. If none of the source files are found, you will receive this message:
>
> ```
> 0 File(s) copied
> ```

CTTY (Change Console)

Internal

Syntax: **CTTY** *device-name*

Purpose: Changes the standard input and output devices.

Comments: CTTY defines the device to be used as the primary console.

AUX, COM1, or COM2:
Defines that device as the primary console.

CON:
Resets the standard input and output device to the primary console.

Example: The following command causes DOS to use the AUX device for its screen and keyboard operations:

```
CTTY AUX
```

The following command moves input and output back to the original device (standard screen and keyboard):

```
CTTY CON
```

You must specify the device which has capability of both input and output operations.

DOS COMMANDS

DATE

Internal

Syntax: **DATE** [*mm-dd-yy*]

Purpose: Sets and displays the date.

Comments: The date is recorded in the directory for any files you create or alter. If you enter a valid date with the DATE command, the date is accepted, and the system prompt appears. Otherwise, the DATE command issues the following prompt:

```
Current date is day mm-dd-yy
Enter new date:_
```

Enter a new date in the form mm-dd-yy or mm/dd/yy,
where:

mm is a number from 1–12
dd is a number from 1–31
yy is a number from 80–99
(the 19 is assumed)
or a number from 1980–2099

The date is entered from the keyboard or from a batch file. The valid separators within the date are hyphens (−) and slashes (/). The system does not prompt you for the date if you use an AUTOEXEC.BAT file. You may include the DATE command in that file.

NOTES: To leave the date unchanged, press <ENTER>.

DOS COMMANDS

DEL

See "ERASE Command" in this chapter.

DIR (Directory)

Internal

Syntax: **DIR**[*d:*][*path*] [*filename*[*.ext*]][/P][/W]

Purpose: Lists either all the directory entries, or only those for specified files.

Comments: The displayed information includes the volume identification and the amount of free space on the disk.

The line displayed for each file includes the file size in decimal bytes and the date and time information was last written to the file.

The notation <DIR> in the file size field indicated entries that name other directories.

Directory entries for system files are not listed, even if they are present.

Parameters:

/P The display will pause when the screen is filled. Press any key to continue with the directory listing.

/W Creates a wide display of the directory so that 5 filenames and directory names are shown on each line. (Only 80-column displays should use this parameter.)

The wildcard characters ? and * may be used in the filename and extension parameters.

The DIR command has two format options. (the /P and /W parameters may be used with either option).

Option 1—List All Files

This option lists all the files in a directory. For example:

DIR [*path*]

or

DIR *d:*[*path*]

The first example lists all directory entries on the default drive. The second example lists all directory entries on the specified drive. If a path is specified, the listing is of files in the specified directory. The current directory is listed if no path is specified.

The directory listing might look like this:

```
A>DIR

Volume in drive A is DISKTMP

Directory of A:\
TMP1    A        1099    10-05-83   9:13p
TMP2    A       20123    12-13-83   7:25a
TRYIT            4093     7-05-82   5:34p
PROGS      <DIR>          9-17-83   8:47p
FILEA   A        2288    11-12-83   2:32p
         5 File(s)      131215 bytes free
```

If the directory being listed is not the root directory it will include two unusual entries. The first entry contains a period in place of a filename. The second contains two periods in place of filename. The list of files shown above would follow these two entries. These two entries indicate that the directory being listed is a sub-directory, rather than the root directory.

Option 2—List Selected Files

This option lists selected files from a directory. For example:

DIR *filename.ext*

or

DIR *d:filename.ext*

If either the filename or the extension is omitted, an * is assumed.

Both examples list all the files that have the specified filename and extension. The first example uses the default drive and the second example uses the specified drive.

Using the previous example, if you enter:

```
dir tmp2.a
```

the screen might look like this:

```
A>dir tmp2.a

Volume in drive A is DISKTMP
Directory of A:\

TMP2 A          20123      12-13-83    7:25a
            1 File(s)      131215 bytes free
```

To see the entry for a file that has no extension, enter the filename followed by a period. In this case, the .ext does not default to *. For example:

```
dir file1.
```

This command displays the entry for FILE1, but not for FILE1.A.

If you wish to display all the files in directory PROGS on the above drive, you can enter:

```
dir progs
```

The screen will look like this:

```
A>dir progs

Volume in drive A is DISKTMP
Directory of A:\progs

.                  <DIR>            9-17-83
..                 <DIR>            9-17-83
PROG12        COM            2863  10-23-83  8:21a

               3 File(s)  140912 bytes free
```

All files in directory PROGS have been listed. The two special entries found in all sub-directories have been included. The single period entry is the directory being listed (PROGS), and the double period is this director's parent directory. In this case, the parent is the root directory. If your current directory is PROGS and you want to view the files in the parent directory, enter:

```
dir ..
```

The following screen is displayed:

```
A>dir ..

Volume in drive A is DISKTMP
Directory of A:\

TMP1     A       1099    10-05-83   9:13p
TMP2     A      20123    12-13-83   7:25a
TRYIT            4093     7-05-82   5:34p
PROGS           <DIR>     9-17-83   8:47p
FILEA    A       2288    11-12-83   2:32p

               5 File(s)     133,874 bytes free
```

DISKCOMP (Compare Disk)

External

Syntax: **DISKCOMP** [*d:*][*d:*][/1][/8]

Purpose: Compares the contents of one disk to the contents of another disk. After a DISKCOPY operation you may run DISKCOMP to ensure that the two disks are identical.

> **NOTES:** This command compares entire disks; the FC command compares files.

Parameters:

/1 Compares only the first side of the disks, even if the disks and drives are dual-sided.

/8 Compares only 8 sectors per track, even if the first disk contains 9 sectors per track.

DOS COMMANDS

DISKCOMP compares track by track. (0–39) If the tracks are different,

```
Compare error(s) on Track xx, Side y
```

XX is the track number (0–39) and y is the side (0 or 1) where a mismatch has occured. When the comparison ot the entire disks terminates, the following prompt appears:

```
Compare more disks (Y/N)?_
```

If you answer Y, the next comparison is performed on the same drives that you originally specified. You do receive prompts to insert the proper disks.

If you answer N, you end the command.

NOTES:

If you omit both parameters:

a single drive comparison is performed on the default drive.

If you omit the second parameter:

the default drive is used as the secondary drive. If the default drive is the first parameter, this results in a single-drive comparison.

A single drive comparison:

all prompts are for drive A, regardless of any drive specifiers entered and are displayed to insert the disks at the appropreate time. DISKCOMP waits for you to press any key before it continues.

If you try to compare a disk created by the COPY command with the disk you copied from DISKCOMP usually display a mismatch message. The COPY command produces a copy that contains the same information, but it places the information at different locations on the target disk from those locations used on the source disk. In this case, use the COMP command to compare individual files on the disks.

If a disk error occurs while DISKCOMP is reading the disk a message indicates where (track and side) the error occurred. DISKCOMP continues to compare the rest of the disk. The remainder of the data cannot be read correctly from the indicated track and side so you can expect to receive a "Compare error message".

The number of sides and sectors per track to be compared is based on the disk that is to be read first (the first drive parameter entered).

If the first disk or drive can be read on only one side, or if the /1 parameter is used only the first side is read from both disks.

If the first disk contains 9 sectors per track DISKCOMP will compare 9 sectors per track unless you used the /8 parameter.

If the first disk is dual-sided, and /1 is not specified a two-sided comparison is performed. An error message is produced if either the second drive or the disk is a single-sided disk.

DISKCOPY (Copy Disk)

External

Syntax: **DISKCOPY** [*d:*][*d:*][/1]

Purpose: Copies the contents of one disk to another disk. The target disk is formatted if necessary.

Comments: The source drive is the first parameter. The target drive is the second parameter. You can specify the same drives or different drives. If the drives are the same, a single-drive copy is performed. You are prompted to insert the disks at the appropriate times. DISKCOPY waits for you to press any key before continuing.

/1 The parameter causes DISKCOPY to copy only the first side of the disk, regardless of the disk or drive type.

After copying, DISKCOPY prompts:

```
Copy another (Y/N)?_
```

If you answer Y, the next copy is done on the same drives that you originally specified. You are prompted to insert the proper disks.

If you answer N, the command ends.

> **NOTES:** If the target disk has not been formatted the same as the source disk, DISK-COPY will format the target disk during the copy.

If you omit both drive parameters, a single DISKCOPY drive copy operation is performed on the default drive.

If you omit the second parameter, the default drive is used as the target drive.

If you omit the second parameter and you specify the default drive as the source drive, a single-drive copy operation is performed.

On a single-drive system, all prompts will be for Drive A, regardless of any drive letter you may enter.

Fragmented disks:

are disks that have had a lot of file creation and deletion activity. Disk space is not allocated sequentially. The first free sector found is the next sector allocated, regardless of its location on the disk.

A fragmented disk causes degraded performance due to delays involved in finding, reading, or writing a file.

It is recommended that you use the COPY command, instead of DISKCOPY, to eliminate the fragmentation. For example:

```
COPY A:*.* B:
```

This copies all the file from the disk in Drive A to the disk in Drive B.

After a successful DISKCOPY you may run DISKCOMP to ensure that the disks are identical.

If disk errors are encountered on either disk, DISKCOPY indicates the drive, track, and side in error and proceeds with the copy. The copy (target disk) may or may not be usable, depending on whether the affected disk location was to contain valid data.

DISKCOPY automatically determines the number of sides and sectors per track to copy, based on the source disk. If only the first side of the source disk can be read, then only the first side can be copied. If the source disk is dual-sided, both sides can be copied (unless you override it with the /1 parameter).

If the source disk has ever been formatted with nine sectors per track, then all nine sectors on each track will be copied.

ERASE

Internal

Syntax: **ERASE** [*d:*][*path*][*filename*[*.ext*]]

or

DEL [*d:*][*path*][*filename*[*.ext*]]

Purpose: Deletes a file or files. If no path is entered, the file is deleted from the current directory.

Comments: The shortened from, DEL, is a valid abbreviation. The wildcard characters ? and * are valid in the filename and in the extension. Wildcard characters should be used with caution because multiple files can be erased with a single command.

To erase all files in the current directory, enter:

ERASE [*d*]*.*

To erase all files in a specific directory, enter:

ERASE [*d:*]*path*

ERASE assumes a filename of *.* if no filename is given.

> **NOTES:** The system files cannot be erased. If you use the filespec *.* to erase all of the files in a directory or on a disk, DOS issues the following message to verify that you actually want to erase all files:
>
> ```
> Are you sure (Y/N)?
> ```

Enter Y if you do want to erase all of the files. Otherwise, enter N. Then press <ENTER>.

The two special entries in each subdirectory (• and • •) cannot be erased.

Examples: The file MYFILE.123 will be erased from the current directory of Drive A.

ERASE A:MYFILE.123

EXE2BIN

External

Syntax: **EXE2BIN** [*d:*][*path*] *filename*[*.ext*] [*d:*][*path*][*filename*[*.ext*]]

Purpose: Converts .EXE files to a form that is compatible with .COM programs. This results in a saving of disk space and faster program loading.

Comments: The first file named is the input file. The default extension is .EXE. The input file is converted to .COM file format (memory image of the program) and stored in the output file, [*d:*]*filename*[*.ext*]. The drive of the input file is used if you do not specify a drive. The input filename is used if you do not specify an output filename. The new file is given an extension of .BIN if you do not specify a filename extension in the output filename. The current directory is used if you do not specify a path.

The input must be in valid .EXE format as produced by the **linker**. The resident, or actual code and data, part of the file must be less than 64K. There must be no STACK segment.

Two kinds of conversions are possible, depending the specified initial CS:IP.

If CS:IP is not specified in the program (the .EXE file contains 0:0), a pure binary conversion is assumed. If segment fixups are necessary (instructions requiring segment relocation are in the program), you are prompted for the fixup value. This value is the absolute segment at which the program is to be loaded.

In this case, the DOS command processor will not be capable of properly loading the program. The program is usable only when loaded at the absolute memory address specified by a user application.

If CS:IP is specified as 0000:100H, it is assumed that the file is to be run as a COM file. The location pointer is set at 100H by the assembler statement ORG and the first 100H bytes of the file are deleted. No segment fixups are allowed because COM files must be segment relocatable. Once the conversion is complete, you may rename the resultant file to a .COM extension. Then, the command processor is capable of loading and executing the program in the same manner as the .COM programs supplied on your DOS disk.

If CS:IP does not meet one of these criteria, or if it meets the COM file criterion but has segment fixups, the following error message is displayed:

```
File cannot be converted
```

This message is also displayed if the file is not a valid .EXE file,

To produce standard COM files with the assembler:

Use the assembler statement ORG to set the location pointer of the file at 100H.

Specify the first location as the start address. This is done by using the END statement. For example:

```
ORG 100H
START:
•
•
•
END START
```

The program must not use references that are defined only in other programs.

DOS COMMANDS

External

Syntax: **FC** [/#] [/B] [/W] [/C] [*d:*] [*path*] [*filename*[*.ext*]] [*d:*] [*path*] [*filename*[*.ext*]]

Purpose: Compares the contents of the first file (or set of files) to the contents of the second file (or set of files).

> **NOTE:** FC compares two sets of files, DISKCOMP compares two entire disks.

Comments: The files that you compare may be on the same drive or on different drives, and they may be in the same directory or different directories.

It is possible to compare all files in one directory with all corresponding files in another directory. For example:

```
FC      A: PROGS1   A: PROGS2
```

There are four switches that you can use with FC:

/B Forces a binary comparison of both files. The two files are compared byte-to-byte, with no attempt to re-synchronize after a mismatch. The mismatches are printed as follows:

```
--ADDRS----FI----F2-
xxxxxxxx    yy    zz
```

(where xxxxxxxx is the relative address of the pair of bytes from the beginning of the file). Addresses start at 00000000; yy and zz are the mismatched bytes from file 1 and file 2, respectively. If one of the files contains less data than the other, then a message is printed out. For example, if file 1 ends before file 2, then FC displays:

```
*** Data left in F2 ***
```


/# # stands for a number from 1 to 9. This switch specifies the number of lines required to match for the files to be considered as matching again after a difference has been found. If this switch is not specified, it defaults to 3. This switch is used only in source comparisons.

/W Causes FC to compress whites (tabs and spaces) during the comparison. Thus, multiple contiguous whites in any line will be considered as a single white space. Note that although FC compresses whites, it does not ignore them. The two exceptions are beginning and ending whites in a line, which are ignored. For example (note that an underscore represents a white)

__More__data_to_be_found__

will match with

More_data_to_be_found

and with

____More____data_to_be____found____

but will not match with

__Moredata_to_be_found

This switch is used only in source comparisons.

/C Causes the matching process to ignore the case of letters. All letters in the files are considered upper-case letters. For example,

Much_MORE_data_IS_NOT_FOUND

will match

much_more_data_is_not_found

FC reports the differences between the two files you specify by displaying the first filename, followed by the lines that differ between the files, followed by the first line to match in both files. FC then displays the name of the second file followed by the lines that are differents, followed by the first line that matches. The default for the number of lines to match between the files is 3. (If you want to change this default, specify the number of lines with the /# switch.) For example,

```
------<filename 1>
<difference>
<1st line to match file 2 in file 1>

------<filename 2>
<difference>
<1st line to match file 1 in file 2>
--------
```

FC will continue to list each difference.

If there are too many differences (involving too many lines), the program will simply report that the files are different and stop.

If no matches are found after the first diference is found, FC will display

```
*** Files are different***
```

and will return to the prompt (for example, A>).

DOS COMMANDS

FIND Filter

External

Syntax: **FIND**
[/V] [/C] [/N]*string*[[*d:*] [*path*]*filename*[*.ext.*]...]

Purpose: This filter sends to the standard output device all lines from the filenames specified in the command line that contain the specified string.

Comments: The following options are available:

/V All lines not containing the string are displayed.

/C A count of the number of matching occurrences of the string in each file are displayed. The matching lines are not displayed.

/N The relative line number of each matching line is displayed ahead of the line from the file.

The string should be enclosed in double quotes. Two quotes in succession are taken as a single quote.

Wildcard characters are not allowed in the filenames or extensions.

Example: FIND "My Story" story1.txt story2.txt story3.txt

This command outputs all lines from story1.txt, story2.txt and story3.txt (in that order) that contain the string "My Story".

Since FIND creates a temporary file on the disk, the disk must not be write-protected. The temporary file is deleted when the operation finishes.

DOS COMMANDS

FORMAT

External

Syntax: **FORMAT** [*d:*] [/S][/V][/B] [/1] [/8]

Purpose: Sets up the disk in the named or default drive to a format usable by DOS. Studies the entire disk for any bad tracks. Readies the disk to accept DOS files by setting up the directory, File Allocation Table, and system loader.

Comments: All new disks must be formatted (using either the FORMAT or DISKCOPY command) before they can be used. Formatting destroys all previously existing data on the disk.

The following options are available:

/S The operating system files are copied from the default drive to the new disk.

/1 The disk is formatted for single-sided use, regardless of the drive type.

/8 The disk is formatted for use at eight sectors per track. The default is nine sectors per track usage if you do not specify /8. FORMAT always creates nine physical sectors on each disk track, but it instructs DOS to use only eight sectors per track if you use the /8 parameter.

/V A volume label is created. The label is written on the disk. The volume label is for your use in keeping track of your disks. The /V parameter cannot be used with the /8 parameter. FORMAT will prompt you to enter a volume label (volume identification). The label can have from one to eleven characters. All characters acceptable in filenames are also acceptable in the volume label. The volume label, however, does not contain a period between the eighth and ninth characters.

/B Creates an eight or nine sector-per-track disk with space allocated for the system files. It does not place the system modules or the command processor on the disk.

The /S parameters cannot be used with the /B parameter.

NOTES: Any defective tracks are marked as reserved to prevent the tracks from being allocated to a file.

Directory entries for system files are marked as hidden files. They do not appear in the directory.

FORMAT produces a status report. The report includes:

Total disk space

Space marked as defective

Space currently allocated to files
(when the parameter /S is used)

Space available for your files

A disk formatted for single-sided use can be used in a dual-sided drive. A disk formatted for dual-sided use will not be usable in a single-sided drive.

The following command formats the disk in Drive B, the operating system files are copied and a volume label is written.

```
FORMAT  B:/S/V
```

The system displays the following message:

```
Insert new disk for drive B:
and strike any key when ready
```

After you insert the appropriate disk and press any key, the system displays this message:

```
Formatting..
```

When the formatting is complete, the system displays this message:

```
Formatting...Format complete
System transferred

Volume label (11 characters, ENTER for none)? DATADISK


   xxxxxx bytes total disk space
   xxxxxx bytes used by system
   xxxxxx bytes available on disk

Format another (Y/N)?
```

Enter Y to format another disk.

Enter N to end the FORMAT program.

GRAPHICS (Screen Print)

External

Syntax: **GRAPHICS**

Purpose: Prints the contents of a graphics display screen.

Comments: To print what is on the screen, press the <Shift> <PrtSc> keys. If the screen is in text mode, the text will print in less than 1 minute. If the screen is in graphics mode, the following things occur each time the <PrtSc> key is pressed:

The screen contents will be printed in up to four shades of gray in the 320×200 color graphics mode.

The screen will print sideways on the paper in the 640×200 color graphics mode. The lower left corner of the screen will print on the lower right corner of the paper.

The code INT 5 will call the screen print from a program.

MKDIR (Make Directory)

Internal

Syntax: **MKDIR** [*d:*]*path*

or

MD [*d:*]*path*

Purpose: Generates a sub-directory on the indicated disk.

Comments: The default drive is assumed if no drive is named.

Example: This command generates an entry in the root directory for a new sub-directory named SUB1:

```
MD \SUB1
```

To add another directory level, use either of the following two examples:

If the current directory is the root directory:

```
MD \SUB1\SUB2
```

This command adds an entry for sub-directory SUB2.

If the current directory is SUB1:

```
MD SUB2
```

Both examples will do the same thing. The directory SUB1 will have an entry for sub-directory SUB2.

The first \ tells DOS to search directories in the root directory. With no leading \ in the last example, DOS begins searching in the current directory. Each directory may hold names of other directories.

The only limit to creating as many sub-directories as wanted is available disk space. Be sure that the maximum length of any single path from the root directory to the level you want is 63 characters, including all backslashes.

MODE

External

Syntax: **MODE** LPT#:[*n*][,[*m*][,P]]

or

MODE *n*

or

MODE [*n*],*m*[,T]

or

MODE COM*n:baud*[*,parity*[*,databits* [*,stopbits*[,P]]]]

or

MODE LPT#:=COM*n*

DOS COMMANDS

Purpose: Determines the mode of operation for a monitor or printer. Routes printer output to an serial (RS232C) interface.

Comments: If an *n* or *m* parameter is invalid or missing, the mode operation will not be altered for that parameter. There are four format options:

Option 1—Use with the printer.

MODE LPT#:[*n*][,*m*][,P]]

where:

#	is 1, 2, or 3 (the printer number)
n	is 80 or 132 (characters per line)
m	is 6 or 8 (lines per inch vertical spacing)
P	specifies continuous retry on time-out errors

For example:

MODE LPT1:132,8

The printer is set to 80 to 132 characters per line and 8 lines per inch of vertical spacing. 80 characters per line and 6 lines per inch are the default options for the printer.

<Ctrl><Break> will stop the retry loop. In order to stop the retry of time-out errors if you have entered parameter P, use MODE Option 1 without naming P.

Option 2—Use to switch the built-in display circuit to initiate the display modc of it.

MODE *n*

or

MODE [*n*],*m*,[T]

where:

n is 40, 80, BW40, BW80, CO40, CO80, or MONO

40	places the width of the display at 40 characters per line
80	places the width of the display at 80 characters per line
BW40 (CO40)	The display mode is set to Black and White with 40 characters per line (color is deactivated).
BW80 (CO80)	The display mode is set to Black and White with 80 characters per line (color is deactivated).
MONO	alters the active display adapter to the Monochrome Display Adapter. This will always have a display width of 80 characters per line.

m is L or R used to shift the display left or right.

T asks for the use of a test pattern to align the display.

Shifting the display one character for 40 column displays or two characters for 80 column displays in either direction will enhance readability. If you enter T in the MODE command, a prompt will ask if the screen is aligned correctly. Entering Y ends the command. Entering N repeats the shift and displays the prompt again. For example:

```
MODE 80,L,T
```

This places the display mode of operation at 80 characters per line and shifts the display two characters to the left. Since the test pattern is shown again you have the chance to shift the display again without having to enter the command again.

Option 3—Use with the serial (RS232C) interface. (Asynchronous Communication Interface)

MODE COM*n*:*baud*[,*parity* [,*databits*[,*stopbits*[,P]]]]

where:

n	Set to either 1 or 2 (1: Internal serial interface (Serial Port) 2: Optional Asynchronous Communication Interface)
baud	110, 150, 300, 600, 1200, 2400, 4800, or 9600
parity	Either E (even), O (odd), or N (none). The default is E.
databits	Either 7 or 8. The default is 7.
stopbits	Either 1 or 2. If the baud is 110, the default is 2. If the baud is not 110, the default is 1.

DOS COMMANDS

To indicate baud you only need to enter the first two characters. Other characters are ignored.

These are called the protocol parameters. They initialize the serial interface. You must enter at least the baud rate. To specify defaults, all other parameters may be omitted by entering only commas. For example,

MODE COM1:30,O,8,1,P

This command places the baud rate at 300, odd parity, eight databits, and one stopbit. To get the defaults for the above, enter:

MODE COM1:30,,,,P

The parity will default to even, the databits to seven, and the stopbits to one.

In order to set the serial interface for use as a serial interface printer, and to have the time out errors retried, enter option P. Pressing Ctrl-Break stops the retry loop. Reinitializing the serial interface without entering the P will stop the continuous retry of time out errors.

Option 4—Use to redirect parallel printer output to an serial (RS232C) interface (Asynchronous Communication Interface)

MODE LPT#:=COM*n*

where:

\# Either 1, 2, or 3 (printer number)

n Either 1 or 2 (1: Internal serial interface, (Serial Port), 2: optional Asynchronous Communication Interface)

The output originally sent to printer LPT# is sent to the *n* instead.

> **NOTES:** To redirect parallel printer output to a serial device, first initialize the serial interface with Option 3 (see above). Include parameter P if that serial device is a printer.
>
> To stop the redirection for the printer designated by the #, enter:
>
> MODE LPT#:[*n*][,*m*]

DOS COMMANDS

MORE Filter

External

Syntax: **MORE**

Purpose: This filter reads data from the standard input device, and sends one screen-full of data to the standard output device. The transmission pauses with the message —More— on the screen.

Comments: The next screen-full of data is written to the standard output device when any character key is pressed. This process continues until all input data is read.

Example: The command line displays the contents of the TEST.ASM one screen-full at a time. When the screen is full, the message —More— appears on the bottom line. To see the next screen-full you can press any key:

```
MORE<TEST.ASM
```

Since MORE creates a temporary file on the disk, the disk must not be write-protected. The temporary file is deleted when the operation finishes.

PATH (Set Search Directory)

Internal

Syntax: **PATH** [*d:*] *path*[[;[*d:*]*path*]...]

Purpose: Sets a path to be searched for commands or batch files that were not found by a search of the current directory.

Comments: The list of drives and path names must be separated by semicolons. If a command cannot be found in the current directory, DOS will search the named directories in the order you entered them, but the current directory is not changed.

PATH with no parameters displays the names that were indicated in a previous PATH command. PATH with only a semicolon (PATH;) resets the search path to null. The default when DOS begins is a null search path. DOS will only search the current directory for commands and batch files.

Examples: Assume the program PROGA.COM resides in directory ACCTDIR on Drive B, and that the default drive is Drive A:

```
PATH \SUB1;\SUB1\SUB2;B:\ACCTDIR
```

This instructs DOS to look in the current directory of the drive specified, followed by A: SUB1, then A: SUB1 SUB2 then B: ACCTDIR until it finds the command you have entered. If the command entered is not found in any of the directories specified in PATH, the message "Bad command or filename" is displayed.

If you enter the command:

PROGA

DOS searches four directories. DOS finds the program PROGA in B: ACCTDIR.

NOTES: Incorrect information in the paths, such as invalid drive specifications or imbedded delimiters, will not be detected until the specified paths are actually searched.

If a path is specified that no longer exists, DOS ignores that path and goes on to the next.

PRINT

External

Syntax: **PRINT** [[*d:*][*filename*[*.ext*]][/T][/C][[/P]...]

Purpose: Prints a list of data files on the printer while you are doing other jobs on the computer.

Comments: You may enter multiple filenames on the command line, each filename with appropriate parameters. Up to 10 filenames can be listed for printing at one time. Wildcard characters * and ? are allowed in the filename and extension. Only files in the current directory can be listed for printing. You can change the current directory without affecting the printing of the files already in the print queue.

The files are queued for printing in the order entered. After each file is printed, the printer paper is advanced to the next page.

Parameters:

/T Terminate mode. All listed files are canceled from the print queue (files waiting to be printed). If a file is currently being printed:

The printing stops.
A cancellation message is printed.
The paper is advanced to the next page.

/C Cancel mode. You may select which file or files to cancel. The preceding filename and all following filenames entered on the command line are canceled from the print queue until a /P is entered on the command line, or the <ENTER> key is pressed.

/P Print mode. The preceding filename and all following filenames are added to the print queue until a /C is entered on the command line, or the <ENTER> key is pressed.

If no parameters (/) are specified following a filename and the <ENTER> key is pressed, the files listed on the command line are listed for printing (/P is assumed).

If PRINT is entered with no filenames, PRINT displays the names of the files currently in the print queue.

The first time the PRINT command is executed after you start your system, the following message is displayed:

```
Name of list device [PRN]:
```

You may specify the output list device, LPT1, LPT2, LPT3, PRN, COM1, COM2, AUX, etc. If you press <ENTER> the default (PRN) will be selected.

> **NOTE:** Be sure the device you name is attached to your system. Specifying a nonexistent device causes unpredictable system behavior.

If a disk error occurs while the system attempts to read the file to be printed, PRINT will:

Cancel the file currently printing.

Print a disk error message.

Advance the printer paper to the next page.

Print the remaining files in the print queue.

If the /T or /C parameters are used to cancel a file or files currently being printed, PRINT will:

Print a file cancellation message.

Advance the printer paper to the next page.

Resume printing with the first file remaining in the print queue.

NOTES: The disk containing the files being printed must remain in the specified drive until all printing is complete. Files in the print queue must not be altered or erased until after they have been printed.

The printer cannot be used for any other purpose while PRINT has data to print.

Examples: To use the PRINT command for the first time, enter:

```
PRINT a:myfile.tst
```

DOS responds with:

```
Name of list device [PRN]:
```

Press the <ENTER> key to send output to the printer.

DOS adds the file MYFILE.TST from Drive A to the print queue and outputs its content to the device "PRN" printer.

To empty the print queue, enter:

    PRINT /T

To remove all TEST* files from the print queue that have the same drive letter as the default drive, enter:

    PRINT test.*/C

To remove all TEST * files from the printer queue Drive A from the print queue, enter:

    PRINT a:test1.tst/C a:test2.tst a:test3.tst

To add the files TEST1.TST and TEST2.TST to the print queue, and remove TEST3.TST from the print queue.

    PRINT test1.tst test2.tst test3.tst/C

To remove file TEST1.TST from the print queue, and add the files TEST2.TST and TEST3.TST to the print queue, enter:

    PRINT test1.tst/C test2.tst/P test3.tst

PROMPT (Set System Prompt)

Internal

Syntax: **PROMPT** [*prompt-text*]

Purpose: Changes the DOS prompt.

Comments: All text on the PROMPT command line becomes the new DOS prompt. If no text is specified, the default DOS prompt is assumed. Strings with special meaning may be imbedded in text in the form $\$c$.

c is one of the following:

$	The "$" character.
t	The time.
d	The date.
p	The current directory of the default drive.
v	The version number.
n	The default drive.
g	The ">" character.
l	The "<" character.

DOS COMMANDS

b	The "¦" character.
q	The "=" character.
h	A backspace
e	The ESCape character.
–	A CR LF sequence.

Example: This command would set the normal DOS prompt:

PROMPT ng

This command would set ABC as the system prompt:

PROMPT ABC

DOS COMMANDS

This command would set up a two-line prompt:

PROMPT Time=t_Date=$d

The prompt would be:

Time=(current time)

Date=(current date)

If you precede any of the DOS command delimeters (such as semicolon, blank, etc.) with a character plus a null string, you can create a prompt that begins with one of the delimiters. In the following example, the semi-colon will be treated as the first character of the prompt, rather than as a delimiter between the word PROMPT and its parameter. For example:

PROMPT $A;ABC

The $A is treated as a null character, because A is not one of the defined characters in the above list. All of the characters that follow the null character will become the new DOS prompt.

RECOVER

External

Syntax: **RECOVER** [*d:*] [*path*]*filename*[*.ext*]

or

RECOVER *d:*

Purpose: Recovers files from a defective disk. The disk may have a defective sector or the directory may be damaged.

Comments: If a sector on a disk is bad, you can recover the file specified *filename* containing that sector. The part of the file that is in the bad sector is not recovered.

In the second format, all files on the specified disk are recovered. It is assumed that the directory is damaged.

If the filename or extension include the wildcard characters * and ?, only the first matched file is recovered.

Examples: If the disk file to be recovered is PROG1 enter:

```
RECOVER A:PROG1
```

The disk file PROG1 on Drive A is read sector by sector, skipping the bad sectors. The filename is not changed. When the bad sectors are found, the sectors are marked and DOS will no longer allocate your data to that sector.

To recover the contents of an entire disk from Drive A, enter:

```
RECOVER A:
```

The disk file allocation table on Drive A is scanned for chains of allocation units. A new root directory is created for every chain of allocation units. The directory is in named as follows:

```
FILExxxx.REC
```

Here, xxxx is a sequential number (0001~). It points to one of the recovered files.

The second format of the RECOVER should be used only if the directory of the disk has become unusable. RECOVER assumes that the entire directory is bad, and recovers all files, including files for which there may still have been valid directory entries.

RENAME (or REN)

Internal

Syntax.: **REN**[**AME**][*d:*][*path*]*filename*[*.ext*]*filename* [*.ext*]

Purpose: Renames a file.

Comments: **RENAME** will alter the name of the file indicated in the first parameter to the name specified in the second parameter. If the second parameter includes drive information it will be ignored.

The abbreviated REN may be used for the RENAME command. The wildcard characters ? and * are valid in the parameters. You may use a path only with the first filename. After changing its name the file will stay in the same directory.

Example: To rename the file AUTO on Drive B to CAR, enter:

```
RENAME  B:AUTO  CAR
```

To rename the file AUTO on Drive B to AUTO.NEW, enter:

```
REN  B:AUTO *.NEW
```

To rename the file PROG.COM in directory SUB2 on Drive B to filename PROG1. COM, enter:

```
REN  B:\SUB2\PROG.COM  PROG1.COM
```

RMDIR (Remove Directory)

Internal

Syntax: **RMDIR** [*d:*] *path*

or

RD [*d:*] *path*

Purpose: Deletes a sub-directory from the named disk.

Comments: First, empty the directory except for the "." and ".." entries. The removed directory will be the last directory named in the path.

Examples: The following command deletes the entry for SUB3 from directory SUB2:

```
RD  B:\SUB2\SUB3
```

NOTE: You cannot remove the root directory and the current directory.

DOS COMMANDS

SET (Set Environment)

Internal

Syntax: **SET** [*name*=[*parameter*]]

Purpose: Inserts strings into the command processor's environment. All the strings in the environment are available to all commands and applications.

Comments: This command is meaningful only if you want to set values that will be used by programs you have written. An application program can check all values that have been set with the SET command by issuing SET with no options. For example, SET TTY=VT52 sets your TTY value to VT52 until you change it with another SET command.

The SET command can also be used in batch processing. In this way, you can define your replaceable prameters with names instead of numbers. If your batch file contains the statement "LINK %FILE%", you can set the name that DOS will use for that variable with the SET command. The command SET FILE=DOMORE replaces the %FILE% parameter with the filename DOMORE. Therefore, you do not need to edit each batch file to change the replaceable parameter names. Note that when you use text (instead of numbers) as replaceable parameters, the name must be ended by a percent sign.

Examples: The following command adds the string STR=abc to the other strings in the environment. The lowercase str is converted to uppercase. Using SET you can enter keywords and parameters that are not meaningful to DOS, but can be found and interpreted by applications that examine the environment.

```
SET str=abc
```

The following command adds the string ABCD=\ACCTS2 to the environment. An application program can search the environment for the name ABCD, and use the supplied parameter as the directory name to use for its files.

```
SET ABCD=\ACCTS2
```

The following command removes ABCD = ACCTS2 from the environment:

SET ABCD=

NOTES: You do not need the SET command to add either PROMPT or PATH commands to the environment. DOS add these two commands to the environment when you enter them.

When DOS starts up it places the first string in the environment. This string is a COMSPEC=parameter which describes the path that DOS uses to reload the command processor if necessary.

The message out of environment space appears if you issue a SET command that would cause the combined environment strings to exceed 127 bytes. (If you have not loaded a resident program DOS expands the environment string area to hold additional strings.)

SORT

External

Syntax: **SORT** [/R] [/+*n*]

Purpose: Reads data from the standard input device, sorts the data, and then writes the data to the standard output device.

Comments: The sort uses the ASCII collating sequence. The blanks are not included for Tab characters.

/R Reverses the sort. (Make "Z" come before "A".)

/+*n* An integer that starts the sort with column *n*. The default is column 1. The maximum size for a file that can be sorted is 63K.

Examples:

```
SORT/R <PRESORT.DAT >SORT.DAT
```

The command line reads the file PRESORT.DAT, reverses the sort, then writes the output to the file SORT.DAT.

```
A>DIR  SORT/+14
```

The output of the directory command is piped to the SORT filter. The SORT filter sorts with column 14, which is the column the file size starts in. A directory sorted by file size is created. The output is to the console.

Since SORT creates a temporary file on the disk, the disk must not be write-protected. The temporary file is deleted when the operation finishes.

SYS (System)

External

Syntax: **SYS** *d:*

Purpose: Transfers the DOS system files from the disk in the default drive to the disk in the drive specified by *d:*

Comments: The directory of the disk in drive *d:* must be totally empty, or the disk must have been formatted by a FORMAT *d:*/S or FORMAT *d:*/B command to contain directory entries for the DOS files.

> **NOTE:** SYS is used to transfer DOS files to application program disks which are sold without the DOS files. The space required for these files has already been allocated and the SYS command transfers the files to the disk.

TIME

Internal

Syntax: **TIME** [*hh:mm:ss.xx*]

Purpose: Sets and displays the time. The time is recorded in the directory entry for any files you create or alter.

Comments: Any valid time is accepted, and the system prompt appears. If an invalid entry is made, the TIME command issues the following prompt:

```
Current time is hh:mm:ss.xx
Enter new time:_
```

You now can enter the current time, where:

hh is a number from 0–23 (hours)

mm is a number from 0–59 (minutes)

ss is a number from 0–59 (seconds)

xx is a number from 0–99 (hundredths of a second)

The date is entered from the keyboard or from a batch file.

NOTES: To leave the time unchanged, press <ENTER>.

If you enter incomplete information, the remaining fields are set to zero.

As long as the digits are within the defined ranges the time will be considered valid.

The valid separators within the time are the colon (:) separating the hours, minutes, and seconds, and the period (.) separating the seconds and the hundredths of a second.

An "Invalid time" message is displayed if you enter an invalid time or delimiter.

TREE (Display Directory)

External

Syntax: **TREE** [*d:*][/F]

Purpose: Displays the directory paths found on the specified drive. Lists the files in each sub-directory if you enter the optional /F.

Comments: The default drive is assumed if no drive is entered.

Displays the full path name, along with the names of any directories defined within it.

Option:

/F The names of all files in each sub-directory also will be displayed.

DOS COMMANDS

Examples:

```
TREE B:/F>SEE.DAT
```

All directories on Drive B are displayed. The output will be written to the file SEE.DAT in Drive B, and will contain the names of all sub-directories and files at each directory.

TYPE

Internal

Syntax: **TYPE** [*d:*][*path*]*filename*[*.ext*]

Purpose: Displays the contents of the file on the screen.

Comments: You may examine the contents of a file but cannot modify the file. To modify the file use EDLIN.

The only formatting done is that tab characters are expanded to an eight-character boundary; that is, columns 8, 16, 24, etc.

> **NOTES:** You must specify a filename.
>
> You can redirect the output to a file or the printer. Press <Ctrl><PrtSc> if you want to output the contents of the file.
>
> Text files are legible. Other files, such as object program files, may appear unreadable due to nonalphabetic or nonnumeric characters.
>
> Wildcard characters * and ? are NOT allowed in the filename or extension. The message "File not found" will appear if wildcard characters are entered.

Example:

```
TYPE  B:prog1.tst
```

The file PROG1.TST on the disk in Drive B is displayed on the screen.

VER (Version)

Internal

Syntax: **VER**

Purpose: Displays the version number of DOS.

Comments: The DOS version is a major version number (single digit), followed by a period, followed by a minor revision level (two digit).

DOS COMMANDS

VERIFY

Internal

Syntax: **VERIFY** [ON|OFF]

Purpose: Turns the verify switch on or off when writing to disk.

Comments: **VERIFY ON** stays on until it is turned off. When ON, DOS will perform a verify operation after every disk-write operation. The system will run more slowly when programs write data to the disk because the verification requires extra time.

DOS shows the current state (off or on) of the verify option when you enter VERIFY with no parameters.

Examples:

```
VERIFY ON
```

Turns the verify feature on.

```
A>VERIFY
VERIFY is on
A>
```

Show the current status of the VERIFY command.

VOL (Volume)

Internal

Syntax: **VOL** [*d:*]

Purpose: Shows the disk volume identification of the named drive.

Comments: The default drive is assumed if a drive is not indicated.

Example:

```
A>VOL

     Volume in drive A is DISKABC
A>
```

DOS COMMANDS

CHAPTER 7

BATCH PROCESSING

BATCH

INTRODUCTION

Often you have to type the same sequence of commands over and over to perform a commonly used task. A special file (batch file) can be created to execute the entire sequence simply by typing the name of the batch file. The commands in batch files are processed as if they were typed at a terminal. A batch file must be named with the .BAT extension, and is executed by typing the filename without its extension.

Batch processing is useful for executing several DOS commands with one batch command. For example, a batch file (DOWNEW-.BAT) to format and check a new disk might look like this:

```
1: REM   This file checks new disks
2: REM   It is named DONEW.BAT
3: PAUSE   Insert new disk in drive B
4: FORMAT B:
5: DIR B:
6: CHKDSK B:
```

BATCH

To execute this .BAT file, simply type the filename without the BAT extension:

```
DONEW
```

The result is the same as if each of the lines in the .BAT file was entered at the terminal as an individual command.

Helpful Information for Using Batch Files

1. The filename extension (.BAT) should not be entered to execute a Batch file.

2. You may set up the last command in a batch file to be the name of another batch file. This means that you can call one batch file from another after the first has completed.

3. All commands in the batch file are run.

4. Pressing <Ctrl><C> while in batch mode displays the following prompt:

   ```
   Terminate batch job (Y/N)?
   ```

   Pressing Y will ignore the remainder of the commands in the batch file. The system prompt will appear.

   Pressing N will stop only the current command and batch processing will continue with the next command in the file.

5. If a disk containing a batch file that is being run is removed, DOS will prompt you to insert the disk again before the next command may be processed.

THE AUTOEXEC.BAT FILE

The **AUTOEXEC.BAT** file is a batch file that is automatically executed each time you start the system. An AUTOEXEC.BAT file allows you to automatically execute programs when you start DOS. When you start or restart DOS, the command processor searches the root directory for a file named AUTOEXEC.BAT.

If DOS finds the AUTOEXEC.BAT file, the file is automatically executed and the date and time prompts are bypassed.

If DOS does not find an AUTOEXEC.BAT file when you first load the DOS disk, then the date and time prompts will be issued.

> **NOTE:** If you use an AUTOEXEC.BAT file, DOS will not prompt you for a current date and time unless you include the DATE and TIME commands in the AUTOEXEC.BAT file. You should include these two commands in your AUTO-EXEC.BAT file, since DOS uses this information to keep your directory current.

CREATING A .BAT FILE

To create a batch file, use EDLIN (the Line Editor) or the COPY command.

If, for example, you want to automatically load BASIC and run a program called TEST each time you start DOS, you could create an AUTOEXEC.BAT file as follows.

First, enter:

```
COPY CON: AUTOEXEC.BAT
```

This command tells DOS to copy the information from the console (keyboard) into the AUTOEXEC.BAT file. The AUTOEXEC.BAT file must be created in the root directory of your DOS disk.

Now enter:

```
BASIC TEST
```

This is the first statement in your AUTOEXEC.BAT file. Whenever DOS is started it will load BASIC and run the MENU program.

Now press <Ctrl> <Z> and then the <ENTER> key to indicate the completion of the AUTOEXEC.BAT file.

The TEST program now will automatically be run whenever you start DOS.

You can enter any DOS command or series of commands in the AUTOEXEC.BAT file. To run your own BASIC program, enter the name of your program instead of TEST in the second line of the example.

Creating a .BAT File with Replaceable Parameters

You may want to create an application program and run it with different sets of data. This data may be stored in various DOS files.

You can create a batch (.BAT) file with dummy (replaceable) parameters. These parameters, named %0–%9, can be replaced by values supplied when the batch file executes. For example:

```
COPY CON: SAMPLE.BAT
COPY %1.ABC %2.ABC
TYPE %2.TXT
TYPE %0.BAT
```

Press <Ctrl><Z> and then press <ENTER>.

DOS responds with this message:

```
1 File(s) copied
A>_
```


The file SAMPLE.BAT, which consists of three commands, is now on the disk in the default drive.

When you execute the file, the dummy parameters %0, %1 and %2 are replaced sequentially by the parameters you supply. The dummy parameter %0 is reserved for the drive designator, if specified, and the filename of the batch file (for example, SAMPLE).

> **NOTES:** Up to 10 dummy parameters (%0–%9) can be specified. See the SHIFT command in this chapter if you wish to specify more than 10 parameters.
>
> To use the percent sign as part of a filename within a batch file, you must type it twice. For example, to refer to the file FLE%.EXE, you must type it as FLE%%.EXE in the batch file.

EXECUTING A .BAT FILE

To execute the batch file SAMPLE.BAT and to specify the parameters that will replace the dummy parameters, enter the batch filename (without its extension) followed by the parameters you want DOS to substitute for %1, %2, etc.

The file SAMPLE.BAT consists of 3 lines:

```
COPY %1.ABC %2.ABC
TYPE %2.TXT
TYPE %0.BAT
```

To execute the SAMPLE batch file, type:

```
SAMPLE  A:APROG  B:BPROG
```

As the file executes, SAMPLE is substituted for %0, A:APROG is substituted for %1, and B:BPROG is substituted for %2.

The result is the same as if you had typed each of the commands in SAMPLE with their parameters, as follows:

```
COPY  A:APROG.ABC  B:BPROG.ABC
TYPE  B:B PROG.TXT
TYPE  SAMPLE.BAT
```

BATCH

Notice that the dummy parameter %0 is reserved. It is automatically replaced by the drive designator (if specified) and the filename of the batch file.

TABLE OF BATCH COMMANDS

Command	Purpose	Syntax
ECHO	Allows or inhibits screen display of DOS commands.	ECHO [ON|OFF| *message*]
FOR	Iterative execution of DOS commands.	FOR %%*variable* IN (*set*) DO *command*
GOTO	Transfer control through use of labels.	GOTO *label*
IF	Conditional processing of DOS commands.	IF [NOT] *condition* *command*
PAUSE	Suspends execution of DOS commands.	PAUSE [*remark*]
REM	Displays comments during batch file execution.	REM [*remark*]
SHIFT	Provides access to more than 10 variables.	SHIFT

BATCH

ECHO

Syntax: **ECHO** [ON|OFF|*message*]

Purpose: To allow or inhibit the screen display of the DOS commands executed from a batch file. It does not interfere with messages produced during the execution of the commands.

Comments: ECHO is ON after you have powered-on or reset the system and shows all the commands on the screen as they run. ECHO OFF stops the showing of commands on the screen, including the REM command.

The ECHO message option will show messages on the screen whether or not the current state is ON or OFF. Specific messages will display even when ECHO has been turned off.

The current ECHO state of ON or OFF is displayed if ECHO is issued with no parameters.

BATCH

Example: If the batch file contains the following:

```
echo off
rem **** the echo is now off
dir b:/w
echo on
dir b:/w
```

When the batch file executes, the following will display:

```
A > echo off

Volume on drive B has no label
Directory of B:\

filename1.ext   filename2.ext
      2 file(s) xxxxx bytes free

A> dir b:/w

Volume in drive B has no label
Directory of B:\

filename1.ext   filename2.ext

      2 file(s) xxxxx bytes free
```

The result of using "echo off" in this example is that "echo off" is displayed, the commands "rem" and "dir b:/w" are not displayed, but the output of the dir command is still displayed.

FOR

Syntax: **FOR** %%*variable* IN (*set*) DO *command*

Purpose: To allow iterative execution of DOS commands.

Comments: Each member of the *set* is sequentially substituted in the command which follows DO. Wildcard * or ? characters are valid for members of the set and all matching filenames on the disk will be substituted in the command. Path names are not accepted with filenames in the *set*.

Only one FOR command can be specified on a command line.

Example: If you enter the command:

```
FOR %%f IN (myprog.asm acct.bas acct2.bas)DO dir %%f
```

The result is:

```
dir myprog.asm
dir acct.bas
dir acct2.bas
```

GOTO

Syntax: **GOTO** *label*

Purpose: Transfers control to the line following a label. A label is inserted in a batch file as a colon (:) followed by the label name.

Comments: The GOTO label causes the commands on the line immediately after *label* to process. If *label* is not named, the current batch file ends with the following message:

```
Label not found.
```

To set up a label in a batch file use a colon (:) followed by a character string. The first eight characters of the string are considered the significant characters. The first eight characters must make the label different from all the other labels.

Example: The following batch file sets up an infinite loop of messages on the screen. Your screen will display "rem this is a loop.. "and" **GOTO MYLABEL**".

```
:MYLABEL
rem this is a loop..
GOTO MYLABEL
```

Labels within a batch file are not shown while the batch file is executing. In the above example, the line :MYLABEL would not be shown. Therefore, unreferenced labels may be used to place comments within your batch file. They are not displayed when the file is run.

IF

Syntax: **IF** [NOT] *condition command*

Purpose: Allows conditional processing of DOS commands.

Comments: The DOS *command* will execute when the IF *condition* is true. When the *condition* is false the DOS *command* will be bypassed and the next command in the file will execute.

The *condition* is one of the following:

EXIST *filespec*

If *filespec* is on the indicated drive, EXIST *filespec* is true. Path names are not allowed with the *filespec* when using the EXIST condition variable.

string1==string2

When *string1* and *string2* are identical, this condition is true.

ERRORLEVEL *number*

If the previous program had an exit code (specified as a binary value) of *number* or higher, ERRORLEVEL *number* is true. Your own programs may set an error code that can then be checked by the IF ERRORLEVEL command.

Example: **IF EXIST filespec command:**

```
if exist myfile goto xyz
dir b:
•
•
:xyz
command
```

The batch file with this command assuming "myfile" is named as the %1 parameter would exit to xyz provided "myfile" is on the default drive. The command following the label ":xyz" would be executed. If "myfile" is not found, the "goto xyz" would not be executed and processing would continue with the "dir b:" command in the batch file.

IF string1==string2 command:

```
if %1==Jamie echo Jamie ate it.
```

The batch file with this command, assuming Jamie is named as the %1 parameter, would perform the ECHO batch command. Jamie ate it, would be displayed. Mary, given as the %1 parameter, would make the condition false and the ECHO batch command would not be executed. The next command would then be processed.

IF ERRORLEVEL number command:

```
tstprog
if errorlevel 1 echo tstprog failure.
dir b:
```

BATCH

Assume that the above commands are in a batch file. TSTPROG is a program that sets the errorlevel when it ends. Assume that TSTPROG sets the errorlevel to 0 if it finishes successfully and sets the errorlevel to 1 if it fails. The batch file conditional if errorlevel 1 echo... tests for the situation when TSTPROG failed. If TSTPROG failed, the condition is true and the ECHO batch command is processed displaying the message immediately following the echo command. If TSTPROG finished successfully, the condition is false and the ECHO batch command would not execute. The next command in the batch file, dir b:, would be processeed.

IF NOT EXIST filespec command:

```
if not exist a:%1 copy b:%1 a:
myprog1
```

This IF batch command demonstrates the NOT condition. The batch file needs a certain file to be on Drive A. The IF condition is executed before MYPROG1 to ensure that the needed file is on Drive A. If the file is not on Drive A, the condition is true and the copy command will be processed, copying the file from Drive B to Drive A. If the file is on Drive A, the copy will not be processed. MYPROG1 will then be run.

PAUSE

Syntax: **PAUSE** [*remark*]

Purpose: Suspends the execution of a batch file and displays the message "Strike a key when ready..."

Comments: PAUSE commands within a batch file give you the chance to display messages, change disks between commands, etc. Press any key (except <Ctrl> <Break>) to continue execution of the batch file. <Ctrl> <Break> will end all execution.

The optional remark may also be shown. The optional remark is a string of characters no longer than 121 bytes.

The PAUSE command, inserted at strategic points in your file allows you to decide how much of a batch file you execute. At each PAUSE command, you have time to decide either to stop processing (by pressing <Ctrl><Break>), or to continue processing (by pressing any other key).

Example: This PAUSE command is in a batch file and this message is shown:

```
A >PAUSE Insert new disk in drive B
Strike a key when ready..._
```

Because the system stops, you have time to change disks between commands.

REM (Remark)

Syntax: **REM** [*remark*]

Purpose: Displays comments from within a batch file or provides spacing.

Comments: The *remarks* are shown as the batch processing arrives at the REM command.

Remarks may be any character string no longer than 123 bytes.

The REM command used without remarks, provides spacing and enhances readability within your batch file.

Example: The REM command contained in a batch file displays this:

```
REM This is the checkbook update program.
```

SHIFT

Syntax: **SHIFT**

Purpose: Allows command lines to access more than 10 (%0 through %9) replaceable variables.

Comments: Changeable variables are numbered %0 through %9. These numbers represent relative positioning in a parameter list. To use more than 10 parameters on a command line, execute the SHIFT command to get past the tenth parameter. The parameters on the command line will be shifted one position to the left; the parameter in %1 position moves to the %0 position. Each shift command moves all the parameters to the left by one more position. For example:

```
%0=Apples
%1=Bannans
%2=Carrots
%3=Dragons
•
•
%9
```

The SHIFT results are:

```
%0=Bannans
%1=Carrots
%2=Dragons
•
•
•
%9
```

Example: This example illustrates the use of the SHIFT command in a batch file. A batch file, CHECK.BAT contains the following commands.

```
echo %0 %1 %2 %3
shift
echo %0 %1 %2 %3
shift
echo %0 %1 %2 %3
shift
echo %0 %1 %2 %3
shift
echo %0 %1 %2 %3
```

Call the batch file with these parameters:

```
CHECK PROGA PROGB PROGC
```

These results are produced:

```
A>echo CHECK PROGA PROGB PROGC
CHECK PROGA PROGB PROGC

A>shift

A>echo PROGA PROGB PROGC
PROGA PROGB PROGC

A>shift

A>echo PROGB PROGC
PROGB PROGC

A>shift

A>echo PROGC
PROGC

A>shift

A>echo

A>
```

CHAPTER 8
EDLIN

EDLIN

INTRODUCTION

EDLIN is the line editor program used to create, change, and display source program files or text files.

EDLIN allows you to:

Create new source or text files and save them.

Update existing files and save both the updated and original files.

Delete, edit, insert, and display lines.

Search for, delete, or replace text within one or more lines.

You can edit files one line at a time, with up to 253 characters in each line.

Line numbers are generated and displayed by EDLIN during the editing process, but are not actually present in the saved file. When you insert or delete lines in a file, all line numbers following the inserted or deleted text are automatically adjusted by the number of lines inserted or deleted. As a result, lines always remain consecutively numbered in your file.

HOW TO START USING EDLIN

To start EDLIN, type:

EDLIN [*d:*] [*path*] *filename* [*.ext*] [*/B*]

The filename is the name of the file you want to create or edit. The screen will now display a message and prompt. The prompt for EDLIN is an asterisk (*).

The system will not allow you to edit a file with a filename extension of .BAK as EDLIN assumes that any .BAK file is a backup file. To edit a file with a filename extension of .BAK, you must rename the file to another extension, then start EDLIN and specify the new name.

To edit a new file:

Enter the name of the file you want to create. If EDLIN does not find this filename on a drive, it will create a new file with Command Description section for more information about how to use the Insert Command

```
New file
*_
```

Enter an I. This is the Insert command which allows you to insert lines of text into your new file. Refer to the EDLIN Command section for more information about how to use the Insert Command.

To editing an existing file:

Enter the name of the file you want to edit. When EDLIN finds the file you specify on the designated or default drive, the file is loaded into memory.

If the entire file can be loaded, the following message and prompt will be displayed on your screen:

```
End of input file
*_
```

You can then edit the file using EDLIN editing commands. Refer to the EDLIN Command Description section for more information about EDLIN editing commands.

If the file is too large to be loaded into memory, EDLIN will load lines until memory is 75% full. The * prompt will be displayed on your screen:

```
*_
```

You can then edit the portion of the file that is in memory.

To edit the remainder of the file, you must save some of the edited lines on disk to free memory. EDLIN can then load the unedited lines from disk into memory. Refer to the EDLIN Command Description section for information about how to use the Write and Append Commands.

When you complete the editing session:

Use the End command to save the original and the updated (new) files, or use the Quit command to exit the editing session without saving the updated files. Refer to the EDLIN Command Description section for more information about how to use the End and Quit commands.

If you use the End command, the original file is renamed with an extension of .BAK and the new file has the filename and extension you specified in the EDLIN command.

SPECIAL EDITING KEYS

You can recall an entire command line or modify it with the DOS special editing keys. The last command line you type is automatically placed in a special storage area called a template. The DOS special editing keys are used to edit your command lines. By using the special editing keys, you can:

Repeat a command line.

Correct a mistake in the command line without retyping the entire command line.

Edit and execute a command line which is similar to a preceeding command line with a minimum of typing.

The following table summarizes the special editing keys.

Special Editing Keys

Function	Key	Description
Copy one character	<F1>	Copies one character from the template to the command line.
Copy up to	<F2>	Copies all characters from the template to the command line, up to the character specified.
Copy template	<F3>	Copies all remaining characters in the template to the command line.
Skip one character	<Del>	Skips (does not copy) a character.
Skip up to	<F4>	Skips (does not copy) the characters in the template, up to the character specified.
Void input	<Esc>	Voids the current input; leaves the template unchanged.
Insert mode	<Ins>	Enters or exits insert mode. Allows characters to be inserted into a line.
New template	<F5>	Makes the new command line the new template, but the command line is not executed.

Key: <F1>

Purpose: Copies one character from the template to the command line.

Comments: Pressing the <F1> key copies one character from the template (the top line) to the command line (the bottom line). When the <F1> key is pressed, one character is inserted in the command line. Insert mode is automatically turned off.

Example: Assume that the screen shows:

```
        1:*This is a sample file.
        1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the command line. Pressing the <F1> key copies the first character (T) from the template to the command line.

```
        1:*This is a sample file.
<F1>    1:*T_
```

Each time the <F1> key is pressed, one more character appears:

```
<F1>    1:*Th_
<F1>    1:*Thi_
<F1>    1:*This_
```

EDLIN

Key: <F2>

Purpose: Copies multiple characters up to a specified character, from the template to the command line.

Comments: Pressing the <F2> key copies all characters up to a specified character from the template to the command line. The specified character is the next character typed after <F2>. This character is not copied or displayed on the screen.

Pressing the <F2> key causes the cursor to move to the location of the specified character. If the template does not contain that character, nothing is copied. Insert mode is automatically turned off.

Example: Assume that the screeen shows:

```
      1:*This is a sample file.
      1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the <F2> key copies all characters up to the specified character.

```
      1:*This is a sample file.
<F2>p 1:*This is a sam_
```

Key: <F3>

Purpose: Copies all remaining characters from the template to the command line.

Comments: Pressing the <F3> key copies all remaining characters from the template to the command line. Regardless of the cursor position at the time the <F3> key is pressed, the rest of the command line appears. The cursor is positioned after the last character on the line. Insert mode is then automatically turned off.

Example: Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the <F3> key copies all characters from the template to the command line.

```
        1:*This is a sample file.    (template)
<F3>    1:*This is a sample file._   (command line)
```

EDLIN

Key: <Del>

Purpose: Skips (does not copy) one character in the template.

Comments: Pressing the <Del> key skips one character in the template. Each time you press the <Del> key, one character is not copied from the template. When the edited command line is entered, the characters not copied from the template are deleted.

This key is the opposite of the <F1> key.

Example: Assume that the screen shows:

```
      1:*This is a sample file.
      1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the <Del> key skips the first character (T).

```
      1:*This is a sample file.
<Del> 1:*_
```

The cursor position does not change. Only the template is affected. To see how much of the line has been skipped, press the <F3> key, which copies all remaining characters from the template to the command line.

```
      1:*This is a sample file.
<Del> 1:*_
<F3>  1:*his is a sample file._
```

EDLIN

Key: <F4>

Purpose: Skips (does not copy) multiple characters in the template up to a specified character.

Comments: Pressing the <F4> key skips all characters up to a specified character in the template. If the template does not contain the specified character, nothing is skipped over. When the edited command line is entered, the characters not copied from the template are deleted.

This key is the opposite of the <F2> key.

Example: Assume that the screen shows:

```
        1:*This is a sample file.
        1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the <F4> key skips all characters in the template up to the specified character.

```
        1:*This is a sample file.
<F4>p   1:*_
```

The cursor position does not change. To see how much of the line has been skipped, press the <F3> key, which copies all remaining characters from the template to the command line.

```
        1:*This is a sample file.
<F4>p   1:*_
<F3>    1:*ple file._
```

EDLIN

Key: <Esc>

Purpose: Voids all input and empties the command line.

Comments: Pressing the <Esc> key empties the command line. The template remains unchanged.

<Esc> also prints a back slash (\), moves the cursor to the first position of the next line, and turns insert mode off. Pressing the <F3> key copies the original template to the command line again. The command line now displays the original template.

Example: Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you want to replace the line with "New Information".

```
1:*This is a sample file.
1:*New Information_
```

To cancel the line you just entered (New Line), and to keep "This is a sample file.", press <Esc>. Notice that a backslash appears on the New Information line to tell you it has been cancelled.

```
      1:*This is a sample file
<Esc> 1:*New Information.
          _
```

Press <ENTER> to keep the original line, or to perform any other editing functions. If <F3> is pressed, the original template is copied to the command line:

```
<F3>      This is a sample file._
```

EDLIN

Key: <Ins>

Purpose: Enters insert mode or enters replace mode.

Comments: When you are not in the insert mode, any characters you type replace characters in the template. Replace mode is in effect when you begin to edit a line. Pressing the <Ins> key causes EDLIN to enter insert mode. To return to replace mode, simply press <Ins> again.

Insert mode:

> The current cursor position in the template does not change. Characters are inserted *ahead of* the character the cursor points to. The cursor moves as each character is inserted. When you have finished inserting characters, the cursor will be in its original position in the template.

Replace mode:

> All the characters you type will overstrike and replace characters in the template. If the <ENTER> key is pressed, the remainder of the template will be deleted.

Example: For an example of inserting text, assume that the screen shows:

```
          1:*This is a sample file.
          1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you press the <F2> and f keys:

```
          1:*This is a sample file.
   <F2>f  1:*This is a sample _
```

To insert the word "edit" and a space, press <Ins> and enter "edit":

```
                1:*This is a sample file.
<F2>f           1:*This is a sample_
<Ins>edit       1:*This is a sample edit_
```

If you now press the <F3> key, the rest of the template is copied to the line:

```
       1:*This is a sample edit
<F3>   1:*This is a sample edit file._
```

To exit insert mode, simply press the <Ins> key again.

For an example of replacing text, assume that the screen shows:

```
       1:*This is a sample file.
       1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you then press <F2>m, lary, and then <F3>:

```
       1:*This is a sample file.
<F2>m  1:*This is a sa_
 lary  1:*This is a salary_
<F3>   1:*This is a salary file._
```

Notice that you *replaced* "mple" with "lary."

Key: <F5>

Purpose: Creates a new template.

Comments: Pressing the <F5> key makes the current command line the new template, replacing the contents of the old template. An @ ("at sign" character) is displayed at the end of the line. The command line is now empty and insert mode is turned off.

> **NOTE:** <F5> performs the same function as the <ESC> key, except that the template is changed and an @ ("at sign" character) is printed instead of a \ (backslash).

Example: Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you then press <F2>m, lary, and then <F3>:

```
          1:*This is a sample file
<F2>m     1:*This is a sa_
 lary     1:*This is a salary_
<F3>      1:*This is a salary file._
```

At this point, assume that you want this line to be the new template, so you press the <F5> key:

```
<F5>   1:*This is a salary file.@
       _
```

The @ indicates that this new line is now the new template. Additional editing can be done using the new template.

EDLIN

EDLIN COMMAND INFORMATION

EDLIN commands perform editing functions on lines of text.

Helpful information for using EDLIN commands:

1. The EDLIN prompt is an asterisk (*).

2. All commands are a single letter except the Edit Line command. Commands and string parameters may be uppercase or lowercase, or a combination of both.

3. Delimiters (spaces and commas) are only required between tow adjacent line numbers. For example, to delete line 6, the command 6D is the same as 6 D.

4. Commands execute after you press <ENTER>.

5. Stop commands by pressing <Ctrl> <Break>.

6. Suspend the display by pressing <Ctrl> <NumLock>. To restart the display press any other character.

7. Use the editing keys for editing *within a line* and EDLIN commands for editing on *entire lines.*

8. You can reference line numbers relative to the current line (the line with the asterisk).
   Use a minus sign with a number to indicate lines before the current line. Use a plus sign with a number to indicate lines after the current line. For example:

   ```
   -10, +10L
   ```

   Lists 10 lines before the current line, the current line, and 10 lines after the current line.

9. Multiple commands may be issued on one command line, without any special separators, unless you issue a command to edit a single line using a line number. In that case, a semicolon (;) must separate commands. The string of a Search or Replace command may be ended by a <Ctrl> <Z> instead of a <ENTER>. For example:

    15;−5,+5L

    Edits line 15 and then lists lines 10 through 20 on the screen.

    SThis string<Ctrl><Z>−5,+5L

    Searches for "This string" and then displays 5 lines before and 5 lines after the line containing the matched string.

10. You can insert a control characters into text by using <Ctrl> <V>. Pressing <Ctrl> <V> tells DOS to recognize the next *capital* letter typed as a control character.

    S<Ctrl><V>Z

    Will find the first occurrence of CONTROL-Z in a file.

    You can use a control character in any of the string arguments of Search or Replace by using <Ctrl><V>. For example:

    R<Ctrl><V>C<Ctrl><Z>xyz

    Will replace all occurrences of Ctrl-C in a file by xyz.

    It is possible to insert <Ctrl><V> into the text by typing Ctrl-V-V.

11. The Ctrl-Z character normally means "end-of-file". If you have Ctrl-Z characters elsewhere in your file, you must tell EDLIN that these other control characters do not mean end-of-file.
    Use the /B switch to tell EDLIN to ignore any Ctrl-Z characters in the file and to show you the entire file.

PARAMETERS

Parameter	Description
line	Indicates a line number must be typed. Line numbers must be separated by a comma or a space. A comma or space must also be used to separate line numbers from other options and from the command. The line number may be specified one of three ways: 1. Enter a number. You may enter an integer from 1—65534. If you enter a number larger than the largest existing line number the line will be given the next consecutive line number. 2. Enter a period. A period (.) indicates the current line number. The current line is the last line edited, and is not necessarily the last line displayed. the current line is marked on your screen by an asterisk (*) between the line number and the first character. 3. Enter a pound sign. A pound sign (#) indicates the line after the last line number. This entry has the same effect as entering a number larger than the last line number.
n	Indicates when must specify the number of lines. Use this parameter with the Write and Append commands only.

EDLIN

string *String* indicates an entry of one or more characters representing text to be found, replaced, deleted or to replace other text. Used only with the Replace and Search commands.

Each *string* must be ended by a <Ctrl> <Z> or an <ENTER> (see the Replace command for more information).

No spaces should be left between strings or between a string and its command letter, unless you want those spaces to be part of the string.

EDLIN

TABLE OF EDLIN COMMANDS

EDLIN commands are summarized in the following table. They are described in further detail following the table.

EDLIN Commands

Command	Purpose
line	Edits line number
A	Appends lines
C	Copies lines
D	Deletes lines
E	Ends editing
I	Inserts lines
L	Lists text
M	Moves lines
P	Pages text
Q	Quits editing
R	Replaces lines
S	Searches text
T	Transfers text
W	Writes lines

COMMAND DESCRIPTIONS

Append

Syntax: [*n*]A

Purpose: The **Append** command adjusts the file being edited in memory by adding, at the end of the existing lines in memory, the specified number of lines from the disk.

Comments: Edlin puts the maximum number of lines possible into memory for editing, until memory is 75% full. If memory is at 75% capacity, no action occurs.

The **Append** command is important when the file being edited is too large to fit in memory. It allows you to edit the lines remaining on disk.

First, use the **Write** command to write all edited lines in memory to disk. Then, use the **Append** command to load the unedited lines from disk to memory. When the last file is in memory, the screen displays the message "End of input file".

Copy

Syntax: [*line*], [*line*], *line*, [*count*]C

Purpose: The **Copy** command copies a range of lines, placing them just ahead of a specified line number. The lines can be copied more than one time by using the *count* option.

Comments: If you do not enter line numbers in the first or the second *line* parameter, the default is the current line.

If you do not specify a number in *count*, the default is one line. The system then copies the text once.

After each copy, the file is renumbered automatically. The current line is the first of the copied lines.

Do not overlap line numbers or your screen will display an "Entry error" message. For example:

```
3,20,15C
```

would result in an error message.

EDLIN

Example: The current file is:

```
1: This sample file is
2: used to show copying lines.
3: See what happens when you use
4: the COPY command
5: to copy text in your file.
```

To copy this file once, issue the following command;

```
1,5,6C
```

The file now reads:

```
 1: This is a sample file
 2: used to show copying lines.
 3: See what happens when you use
 4: the Copy command
 5: to copy text in your file.
 6: This is a sample file
 7: used to show copying lines.
 8: See what happnes when you use
 9: the Copy command
10: to copy text in your file.
```

When you copy text within other text, the copied lines will appear *before* the line specified in the third *line* parameter.

The current file is:

```
1: This file shows
2: how to use COPY
3: to the middle
4: of your file.
5: Now you know
6: very well.
```

The command 2,3,6C results in the following file:

```
1: This file shows
2: how to use COPY
3: to the middle
4: of your file.
5: Now you know
6: how to use COPY
7: to the middle
8: very well.
```

Delete

Syntax: [*line*] [,*line*]D

Purpose: The **Delete** command deletes a specified range of lines in a file.

Comments: When you use this command, lines are permanently deleted. If you want to reinstate the lines, you must use the **Quit** (Q) command to end the edit and begin again.

When lines are deleted, the line immediately after the deleted text becomes the current line. The current line and any following lines will then be automatically renumbered.

If you do not enter a line number in one or both of the *line* parameters, the default is the current line. For example:

,4D	results in all lines being deleted from the current line through line 4. You must enter the comma to indicate the first line number is omitted.
4D or 4,D	results in only line 4 being deleted. The line that follows then becomes the current line.
D	results in only the current line being deleted. The line that follows then becomes the current line.

Examples: The current file is:

```
1: This is a sample file
2: used to show dynamic
3: line numbers.
4: See what happens when you
5: use Delete and Insert.
6: Use the D and I commands to
7: edit the text
8: *in your files.
```

EDLIN

To delete multiple lines, such as lines 5 and 6, type:

```
5,6D
```

The result is:

```
1: This is a sample file
2: used to show dynamic
3: line numbers.
4: See what happens when you
5: *edit the text
6: in your files.
```

To delete a single line, type:

```
3D
```

The result is:

```
1. This is a sample file
2: used to show dynamic
3: *See what happens when you
4: edit the text
5: in your files.
```

To delete a range of lines beginning with the current line, type:

```
,4D
```

The result is:

```
1. This is a sample file
2: used to show dynamic
3: *in your files.
```

Notice that the lines are automatically renumbered whenever the delete command is executed.

EDLIN

Edit

Syntax: [*line*]

Purpose The **Edit** command allows you to edit a line of text by displaying the line to be edited.

Comments: Enter the line number you want to edit or a period (.) to indicate the current line. If you do not enter a line number or a period, and you press <ENTER>, the line after the current line will be ready to edit.

EDLIN displays the line numbers and the text on the screen. EDLIN also displays the line number again, below the line of text, ready for editing input.

If you press <ENTER> while the cursor is located at any position other than the beginning or end of the line, you will erase the remainder of the line.

You may use any of the EDLIN editing keys to edit the line. The existing text of the line serves as the template until the <ENTER> key is pressed.

If no changes to the current line are needed and the cursor is at the beginning or end of the line, press the <ENTER> key to accept the line as is.

Example: The following file is ready to edit:

```
1: This is a sample file.
2: used to show the editing
3: of line
4: *four.
```

To edit line 4, type:

```
4 <ENTER>
```

EDLIN

The contents of line 4 are now displayed on the screen, preceeded by an asterisk. The line number is repeated below, ready for input.

```
4:*four.
4:*_
```

Using the special editing keys, you could then perform the following edit:

<Ins>number	4: number_
<F3><ENTER>	4: number four.

After completing your edit, <ENTER> will save the edited line and make it the current line. If you want, you can type moe text after the changed line, as the system is in insert mode when the cursor is at the end of the line.

If you want to do additional editing to the changed line without changing the original line, press <F5>.

If you want to cancel your changes to the line, press <Esc> or <Ctrl> <Break>. The original line will then be unchanged.

End

Syntax: E

Purpose: The **End** command ends the editing session and saves the edited file.

Comments: If the disk does not contain enough free space for the entire file, the write will be aborted. Some or all of the edited file will be lost.

The system will save the edited file by writing it to the drive and filename you specified at the start of EDLIN. If the drive was not selected, the file will be saved on the disk in the default drive.

The original file will be given a .BAK filename extension. If you created a new file instead of updating an existing file, no .BAK file is created.

If you want to end the editing session without saving the edited file, use the **Quit** command.

Insert

Syntax: [*line*]I

Purpose: The **Insert** command inserts text immediately before the specified line. The **Insert** command must be used when you create a new file before you can insert text.

Comments: If you do not enter a line number, or if you enter the line number as a period (.), the insert is made immediately before the current line number.

If the line number you enter is greater than the highest existing line number, or if a pound sign (#) is specified as the line number, the insert is made after the last line in memory. The last line inserted then becomes the current line.

When you are in insert mode, successive line numbers are displayed automatically each time you press <ENTER>.

To exit insert mode enter <Ctrl> <Break>. After you have exited insert, the line immediately following the inserted lines becomes the current line. All line numbers following the inserted section are then automatically renumbered.

EDLIN

Examples: The following file is ready to edit:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: *in your file.
```

To insert text before a specific line that is *not* the current line, such as line 7, enter:

```
7I
```

The result is:

```
7:*_
```

Now, you can enter the new text for line 7:

```
7:*and renumber lines
```

To end the insertion, press <Ctrl> <Break> on the *next* line:

```
8:*<Ctrl><Break>
```

Now you can use the list command, L, to list the file. the screen displays:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: and renumber lines
8: *in your file.
```

To insert lines immdiately before the current line, enter:

```
I
```

The screen displays:

```
8:*_
```

To insert the following text and then terminate the insert with a <Ctrl> <Break> on the next line:

```
8:*so they are consecutive
9:*<Ctrl><Break>
```

Now you can use the List command, L, to list the file again and see the resulting lines. The screen displays:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: and renumber lines
8: so they are consecutive
9: *in your file.
```

To add new lines to the end of the file, enter:

```
10I
```

The screen displays:

```
10: *_
```

Now you can enter the following new lines:

```
10: * The insert command can place new lines
11: * in the file; there's no problem
12: * because the line numbers are dynamic;
13: * they'll go all the way to 65533.
```

End the insertion by pressing <Ctrl> <Break> on line 14. The new lines will appear at the end of all previous lines in the file. Now use the List command, L, to see the resulting lines:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit text
 7: and renumber lines
 8: so they are consecutive
 9: in your file.
10: The insert command can place new lines
11: in the file; there's no problem
12: because the line numbers are dynamic;
13: they'll go all the way to 65533.
```

EDLIN

List

Syntax: [*line*] [*line*]L

Purpose: The **List** command displays on your screen a specified range of lines. The current line is not changed.

Comments: this command displays file lines from the first line number specified through the second line number specified.

If you do not specify a line number for the first *line* the default is 11 lines before the current line. The beginning comma must be inserted to indicate the omitted first *line*.

If you specify a line number for the first *line* which is moe than 11 lines before the current line, the screen display will be the same as if you omitted both *line* parameters.

If you do not specify a line number for the second *line*, the screen displays 23 lines, beginning with the specified *line*.

If you do not specify a line number in either *line* parameter, the screen displays 23 lines: 11 lines before the current line, the current line, and 11 lines after the current line. If there are less than 11 lines before the current line, the screen displays extra lines after the current line, to total 23 lines.

Example: The following file is ready to edit:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
          •
          •
          •
15: *This current line contains an asterisk.
          •
          •
          •
26: to edit text
27: in your file.
```

To list a range of lines from line 2 through line 5, enter:

```
2,5L
```

The screen displays:

```
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
```

To list a range of lines beginning with the current line through line 26, enter:

```
.26L
```

```
15: *The current line contains an asterisk.
          •
          •
          •
26: to edit text
```

EDLIN

To list a range of 23 lines from a specified line number, enter:

```
3L
```

The screen displays:

```
 3: See what happens when you use
 4: Delete and Insert
        •
        •
        •
25: This is a way
```

To list a range of 23 lines centered around the current line, enter:

```
L
```

The screen displays:

```
 4: Delete and Insert
 5: (the D and I commands)
        •
        •
        •
13: The current line is listed in the middle.
14: The current line remains unchanged.
15: *The current line contains an asterisk.
        •
        •
        •
26: to edit text.
```

Since you did not specify any line numbers, the screen displayed 11 lines before the current line, the current line, and 11 lines after the current line.

Move

Syntax: [*line*], [*line*], *line*M

Purpose: The **Move** command moves a range of lines ahead of a specified line in the file being edited.

Comments: This command is used to move a block of text from one location in the file to another.
If you do not specify a number in the first or second *line*, the default is the current line. You *must* enter a line number in the third *line*.

The file lines are automatically renumbered after the text is moved. The first of the moved lines becomes the current line.

Example:

```
,+40,90M
```

This command moves the text from the current line plus 40 lines to line 90.

```
30,50,90M
```

This command moves lines 30–50 to line 90.

If the line numbers overlap, EDLIN will display an "Enter error" message.

Page

Syntax: [*line*] [*line*]P

Purpose: The **Page** command pages through a file displaying a specified range of lines. It also changes the current line.

The **List** command also displays a range of lines but does not change the current line.

Comments: If you do not specify a number in the first *line*, the default is the current line plus one.

If you do not specify a number in the second *line*, the system will display 23 lines.

The last line displayed by the **Page** command, and marked by an asterisk, becomes the new current line.

Quit

Syntax: Q

Purpose: The **Quit** command quits the editing session. This command does *not* save any editing changes. It exits to the DOS operating system.

Comments: After you enter the command, EDLIN displays a prompt on the screen, to make sure you really don't want to save the changes.

If you want to leave the editing session without saving any changes, enter 'Y'. No .BAK file will be created. (Refer to the END command for information about the .BAK file.) Your previous backup copy will no longer exist.

If you want to continue the editing session, enter 'N' or any other character except 'Y'.

Example:

```
Q
Abort edit (Y/N)?Y
A>_
```

EDLIN

Replace

Syntax: [*line*] [,*line*] [?] R [*string1*] [<Ctrl><Z>*string2*]

Purpose: The **Replace** command replaces all occurrences of a string of text in the specified range with a different string of text or blanks.

Comments: Each occurrence of *string1* is replaced by *string2*. The lines in which replacements occur will be displayed. When all occurrences of *string1* in the specified range are replaced by *string2*, the **Replace** command terminates and the asterisk prompt reappears. The last line changed becomes the current line.

If you include both strings, the first string must be separated from the second string with a <Ctrl> <Z>. The second string *must* be ended with a <Ctrl> <Z> <ENTER> combination or with just <ENTER>.

If *string1* is omitted, **Replace** will use the old *string1* as its value. If this is the first replace being done in this EDLIN session, then the replacement process will be terminated immediately.

If *string2* is omitted, then *string1* may be ended with <ENTER>. In this case all occurrences of *string1* are deleted.

Default values for the *line* parameters:

> If the first *line* is omitted (,*line*) then the first *line* defaults to the line *after* the current line.
>
> If the second *line* is omitted (*line* or *line*,), the second *line* defaults to the last line in memory.
>
> If both *line* parameters are omitted, the replace will occur from the line following the current line to the last line in memory.

The quesion mark parameter:

If you include the question mark (?), the Replace command will stop at each line that contains *string1* that matches. The screen will display the line and then display the prompt O.K.?.

If you press Y or <ENTER>, then *string2* will replace *string1*, and the search for a match will continue.

If you press any key besides Y or <ENTER> after the O.K.? prompt, *string1* will be left as it was in the line.

If *string1* occurs more than once in a line, each occurrence of *string1* will be replaced individually, and the O.K.? prompt will be displayed after each replacement. With this method you can choose to replace particular occurrences of *string1*.

This process will continue until the end of the range of lines or until the end of the file. EDLIN displays the asterisk prompt after the last occurrence of *string1*.

Example: The following file is ready to edit:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: in your file.
8: The insert command can place new lines
9: *in the file.
```

To replace all occurrences of and with AND in a specified range, type:

3,9 Rand<Ctrl><Z>AND<ENTER>

The result is:

```
4: Delete AND Insert
5: (the D AND I commands)
5: (The D AND I commANDs)
8: The insert commAND can place new lines
```

Note that in the above replacement, some unwanted substitutions have occurred. To avoid these the same original file can be used with a slightly different command.

In the next example, to replace only certain occurrences of the and with AND, enter:

2? Rand<Ctrl><Z>AND<ENTER>

The result is:

```
  4: Delete AND Insert
O.K.? Y
  5: (The D AND I commands)
O.K.? Y
  5: (The D AND I commANDs)
O.K.? N
  8: The insert commAND can place new lines
O.K.? N
*_
```

To see the result of these changes enter the list command L:

```
4: Delete AND Insert
5: (The D AND I commands)
8: The insert command can place new lines
```

Search

Syntax: [*line*] [,*line*] [?] S*string* <*ENTER*>

Purpose: The **Search** command searches the specified range of lines for a specified string of text.

Comments: The *string* must be ended with the <ENTER> key. The first line that matches *string* is displayed and becomes the current line. The **Search** command will terminate when a match is found unless the question mark parameter is included. If there is no match the message "Not found" will be displayed.

If *string* is omitted, Search will take the old string if there is one. (Note that "old" here refers to a string specified in a previous Search or Replace command.) If there is not an old string (i.e., no previous search or replace has been done), the command will terminate immediately.

Default values for the *line* parameters:

> If the first *line* is omitted (as in ,*line* S *string*), the first *line* will default to the line *after* the current line.
>
> If the second *line* is omitted (as in *line* S *string* or *line*, S *string*), the second *line* will default to # (line after last line of file), which is the same as *line*, # S *string*.

The question mark parameter:

> If the question mark parameter (?) is included in the command, after EDLIN displays the first line with a matching string it will prompt you with the message O.K.?. If you press either the Y or <ENTER> key, the line becomes the current line and the search terminates. If you press any other key, the search continues until another match is found, or until all lines have been searched.

Example: The following file is ready to edit:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: in your file.
8: The insert command can palce new lines
9: *in the file.
```

To search for the first occurrence of the string "and", enter:

```
3,9 Sand<ENTER>
```

The following line is displayed:

```
4: Delete and Insert
```

To search through several occurrences of the string "and" untill the correct string is found, enter:

```
1, ? Sand
```

The result is:

```
4: Delete and Insert
O.K.?_
```

If you press any key (except Y or <ENTER>), the search continues, so enter N here:

```
O.K.? N
```

The search continues:

```
5: (the D and I commands)
O.K.?_
```

EDLIN

To terminate the search you enter N.

To search for string ABC without the verification (O.K.?), type:

```
SABC
```

EDLIN will report a match and will continue to search for the same string if you issue the S command again:

```
S
```

EDLIN reports another match.

```
S
```

EDLIN reports the string is not found.

Note that *string* defaults to any string specified by a previous Replace or Search command.

Transfer

Syntax: [*line*]T[*d:*] *filename*

Purpose: The **Transfer** command merges the contents of a specified file with the file currently being edited.

Comments: This command is used to put the contents of one file into another file, or into the text you are creating. The transferred text is inserted ahead of the specified line in the file being edited.

The file lines are then automatically renumbered.

If you do not specify a line number, the default is the current line.

EDLIN

Write

Syntax: [*n*]W

Purpose: The **Write** command writes a specified number of lines to disk from the lines that are being edited in memory. **Write** begins with line number one (1).

Comments: Use this command if the file you are editing is too large to fit in memory. EDLIN automatically writes file lines into memory until memory is 75% full. This command allows you to edit the remainder of the file.

The **Write** command writes a specified number of edited lines in memory to disk. You can then load additional unedited lines from disk into memory by using the **Append** command.

If you do not specify the number of line to write, the system will write lines to disk until 25% of available memory is used. If available memory is already less than 25% used, no action will be taken.

File lines are automatically renumbered so that the first remaining line becomes line number 1.

MEMO

CHAPTER 9

LINK

LINK

INTRODUCTION

NOTE: If you are not going to compile and link programs, you do not need to read this chapter.

Read this entire chapter before you use LINK.

You write your programs in source code which is passed through a compiler (or assembler) to produce object modules. The object modules must go through the LINK process to produce a run file.

The Linker is a program that:

Combines separately produced object modules into a program you can run (relocatable executable object code).

Searches library files for definitions of unresolved external references.

Resolves external cross-references.

Produces a listing that shows both the resolution of external references and error messages.

LINK uses as much available memory as is possible. When available memory is exhausted, LINK creates a temporary disk file named VM.TMP.

Figure 9-1 illustrates the various parts of the LINK operation.

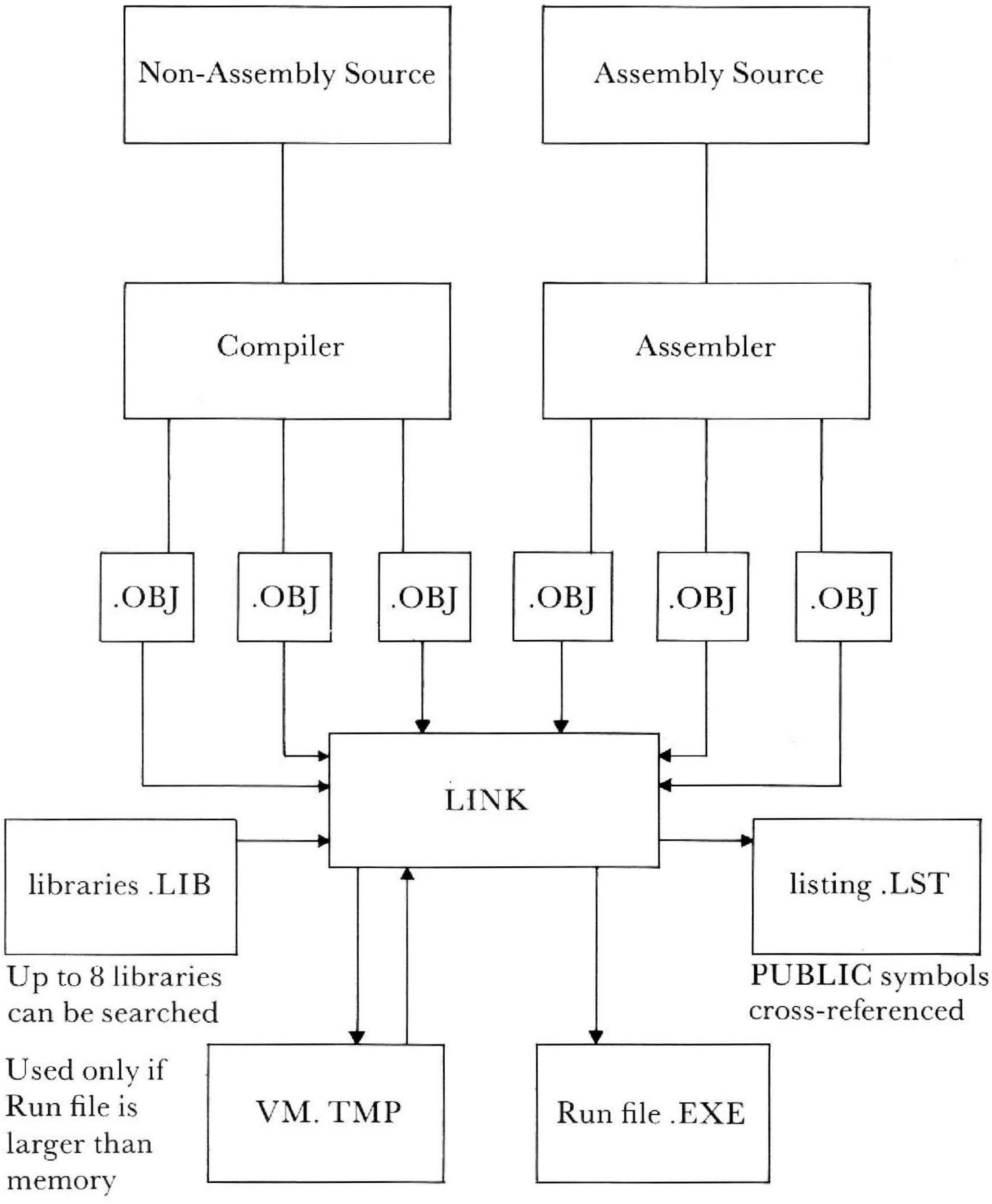


Fig. 9-1 THE LINK OPERATION

LINK

FILES

LINK manipulates the following four files.

INPUT:	Works with one or more input files
OUTPUT:	Produces two output files
TEMPORARY:	May create a temporary disk file
LIBRARIES:	May be directed to search up to eight library files

You may give a three-part file specification for each file. The format for LINK file specifications is the same as that of a disk file:

[*d:*]*filename*[*.ext*]

d: Drive designation. The colon is always required as part of the drive designation.

filename Any legal filename of one to eight characters.

.ext One-to-three character extension to the filename. The period is always required.

Input File Extensions

If you do not provide filename extensions in the input file specifications, LINK will recognize the following extensions by default:

File	Default
Object File	.OBJ
Library	.LIB

LINK

Output File Extensions

LINK appends the following default extensions to the output files:

File	Default
Run	.EXE (may not be overridden)
List	.MAP (may be overridden)

VM.TMP (Temporary) File

LINK uses available memory for the link session. If the files to be linked create an output file that exceeds available memory, LINK creates a temporary file, names it VM.TMP, and puts it on the disk in the default drive. If LINK creates VM.TMP, it will display the message:

```
VM.TMP has been created.
Do not change disk in drive, d:
```

Once this message has been displayed, you must not remove the disk from the default drive until the link session ends. If the disk is removed the operation of LINK will be unpredictable. LINK might display the error message:

```
Unexpected end of file on VM.TMP
```

VM.TMP is a working file only and is deleted at the end of the linking session.

*** **WARNING** ***

Do not use VM.TMP as a filename for any file. If you have a file named VM.TMP on the default drive and LINK requires the VM.TMP file, LINK will delete the VM.TMP already on disk and create a new VM.TMP. The contents of the previous VM.TMP file will be lost.

DEFINITIONS

In DOS, memory can be divided into segments, classes, and groups.

Segment

A contiguous area of memory that is up to 64K bytes in length. A segment may be located anywhere in memory on a 16-byte (paragraph) boundary. The segment contents are addressed by a segment register/offset pair. A program's location in memory is decided at load time by the relocation loader (located in COMMAND.COM).

Class

A collection of segments. Segments with different segment names may or may not have the same class names. The class name is determined by the assembler or compiler. The class affects the order and relative placement of segments in memory. All segments assigned the same class name are loaded into memory contiguously.

Group

A collection of segments that fit within a 64K byte area of memory. The segments do not need to be contiguous to form a group. The lowest address of the segments in that group is the address of that group. A program may consist of one or more groups.

If you are writing in assembly language, you may assign the group and class names in your program. In high-level languages (BASIC, COBOL, FORTRAN, Pascal), the naming is done automatically by the compiler.

COMMANDS

Prompts

LINK displays four prompts that appear one at a time. When the last prompt is answered, LINK begins linking automatically. When the link session is finished, LINK exits to the operating system. If LINK has finished successfully, the operating system prompt appears. If the LINK is unsuccessful, the appropriate error message is displayed.

LINK prompts you for the names of Object, Run, and List files and for Libraries.

The prompts are listed in order of appearance. The default response is shown in square brackets ([ ]) following the prompt. The Object Modules prompt has no default and requires you to type a filename.

Object Modules [.OBJ]

[*d:*] [*path*] *filename* [*.ext*] [+[*d:*] [*path*] *filename* [*.ext*]]...

Enter a list of the object modules to be linked. If the filename extension is omitted, LINK assumes that the filename extension is .OBJ. If an object module has any other filename extension, the extension must be given.

Modules must be separated by plus signs (+) or blanks.

LINK loads segments into classes in the order encountered. You can use this information to set the order in which the object modules will be read by LINK.

LINK

Run File [filename. EXE]

[*d:*] [*path*] [*filename* [*.ext*]]

The filename you enter creates a file for storing the Run (executable) file that results from the link session. All Run files have the filename extension .EXE, even if you specify an extension other than .EXE.

If no response is entered to the Run File prompt, LINK uses the first filename you typed in response to the Object Modules prompt as the RUN filename.
For Example:

```
B:ACCTS
```

in response to the Run File prompt creates the Run File ACCTS.EXE on drive B.

List File [NUL.MAP]
[*d:*] [*path*] [*filename* [*.ext*]

An entry for each segment in the input (object) modules is listed. Each entry also shows the addressing in the Run file.

Libraries [.LIB]
[*d:*] [*path*] *filename* [*.ext*] [+] [*d:*] [[*path*] *filename*[[*.ext*]]]...

The valid responses are up to eight library filenames separated by plus signs (+) or blanks, or a carriage return. A carriage return produces a search of the default library for files which have been created previously. The filename extension defaults to .LIB for library files.

Library file searches (to resolve external references) are in the order listed. When LINK finds the module that defines the external symbol, it processes that module as another object module.

If LINK cannot find a library file on the disks in the disk drives, it displays the message:

Cannot find library *library-name*
Type new drive letter:

Press the letter for the drive designation (for example, B).

Switches

LINK switches control various LINK functions. Regardless of which method is used to start LINK, switches must be typed at the end of a prompt response. Switches may be grouped at the end of any response, or may be scattered at the end of several responses. If more than one switch is used at the end of one response, each switch must be preceded by a forward slash (/).

Switches may be abbreviated in order from the first letter through the last typed. No gaps are allowed. For example, legal and illegal abbreviation for /PAUSE is:

Legal	*Illegal*
/P	/PSE
/PA	/PAE
/PAU	/PAS

/DSALLOCATE

Tells LINK to load all data defined to be in DGROUP at the high end of the Data Segment (DS). Otherwise, LINK loads all data at the low end of the Data Segment.

At runtime, the DS pointer is set to the lowest possible address to allow the entire DS segment to be used. The /DSALLOCATE switch in combination with the default load low (the /HIGH switch is not used) permits the user application to dynamically allocate any available memory below the area specifically allocated within DGROUP. The application will remain addressable by the same DS pointer. This dynamic allocation is needed for Pascal and FORTRAN programs.

NOTE: Your application program may dynamically allocate up to 64K bytes (or the actual amount of memory available) less the amount allocated within DGROUP.

/HIGH

Tells LINK to place the Run file as high as possible in memory. Otherwise, LINK places the Run file as low as possible.

*** **WARNING** ***

Do not use the /HIGH switch with Pascal or FORTRAN programs.

/LINENUMBERS

Tells LINK to include in the List file the line numbers and addresses of the source statements in the input modules.

NOTE: Not all object modules contain line number information. In these cases LINK cannot include line numbers.

/MAP

Tells LINK to list all public (global) symbols defined in the input modules. If /MAP is not included, LINK will list only errors (including undefined globals).

The symbols are listed alphabetically. For each symbol, LINK lists its value and its segment:offset location in the Run file. The symbols are listed at the end of the List file.

/PAUSE

Tells LINK to pause in the link session when the switch is encountered. This switch allows the user to change the disks before LINK outputs the Run (.EXE) file.

When LINK encounters the /PAUSE switch, it displays the message:

```
About to generate .EXE file
Change disks <hit ENTER>
```

LINK resumes processing when you press <ENTER> key.

*** **WARNING** ***

Do not remove the disk which will receive the List file, or the disk used for the VM.TMP file, if one has been created.

/STACK:number

The value of *number* is used to determine the size of the stack. *Number* represents any positive numeric value (in hexadecimal) up to 65536 bytes. If a value from 1 to 511 is typed, LINK will use 512. If the /STACK switch is not used for a link session, LINK will calculate the necessary stack size automatically.

All compilers and assemblers should provide information in the object modules that allow the linker to compute the required stack size.

At least one object (input) module must contain a stack allocation statement. If not, LINK will display the following error message:

```
Warning: No STACK segment
```

/NO

Tells LINK not to search the default libraries in the object modules. /NO is short for NODEFAULTLIBRARY-SEARCH. For example, if you are linking object modules in Pascal, the /NO switch tells LINK not to automatically search the library named PASCAL.LIB to resolve external references.

LINK

Characters

LINK uses three command characters.

Plus sign The plus sign (+) separates entries. (A blank space may be used to separate object modules.) The plus sign also extends the current line in response to the Object Modules: and Libraries: prompts. A plus sign at the end of a line, followed by a <ENTER> will cause LINK to prompt for additional object modules or libraries. Be sure the response line ends with a module name and a <ENTER> (not a plus sign) when all the object modules and libraries have been listed.
For Example:

```
Object Modules [.OBJ]: FILEA TEXT1 PROGB CARS+<ENTER>

Object Modules [.OBJ]:  MYFILE URFILE  OURFILE+<ENTER>

Object Modules [.OBJ]: ACCTFLE <ENTER>
```

LINK

Semicolon Selects default responses to the remaining prompts. Use a single semicolon (;) followed immediately by a carriage return at any time after the first prompt (Run File:). This is a time saving feature and overrides the need to press a series of <ENTER> keys.

> **NOTE:** Do not use the semicolon to skip some prompts. Once the semicolon is accepted by LINK, you will no longer be prompted for information. To skip some of the prompts, use the <ENTER> key.
> For Example:
>
> ```
> Object Modules[.OBJ]: FILEA FILEB FILEC <ENTER>
> Run File [FILEA.EXE]: ;<ENTER>
> ```
>
> No other prompts will appear.
> LINK will use the default values.

<Ctrl><C> The <Ctrl> <C> key aborts the link session at any time. If you type an erroneous response you press <Ctrl> <C> to exit LINK and then you restart LINK. If the error has been typed but you have not pressed the <ENTER> key, you may delete the mistakes with the backspace key, but for that line only.

HOW TO START THE LINKER

LINK requires two types of input:

A command to start LINK.
Responses to command prompts.

In addition, seven switches control LINK features.

You may type all the commands to LINK on the terminal keyboard. As an option, the answers to the command prompts and any switches may be stored in a response file. Command characters can be used to assist you while giving commands to LINK. Refer to the command character section of this chapter for more information.

LINK may be started in any of three ways:

Method 1 LINK

Type the commands in response to individual prompts.

Mothod 2 LINK *filenames*[*switches*]

Type all commands on the line used to start LINK.

Method 3 LINK @*filename*

Create a response file that contains all the necessary commands and tell LINK where that file is when you start LINK.

Method 1: Prompts

To start LINK with Method 1, type:

```
LINK
```

LINK is loaded into memory. LINK displays four text prompts that appear one at a time. Answer the prompts to tell LINK to perform specific tasks.

At the end of each line, you may type one or more switches, preceded by a forward slash.

Prompt	Defaults
Object Modules [.OBJ]:	There is no default; a response is required.
Run File [Object-file.EXE]:	The default is first-object-filename.EXE. (You cannot change the output extension.)
List File [NUL.MAP]:	If you don't specify the filename, List file is not created.
Libraries [.LIB]:	The default is to search for default libraries in the object modules. (Extensions will be changed to .LIB.)

Method 2: Command Line

To start LINK using Method 2, type all commands on one line.

LINK *object-list, runfile, listfile, lib-list* [*/switch...*]

The entries following LINK are responses to the command prompts. The entry fields for the different prompts must be separated by commas.

object-list	a list of object modules, separated by plus signs or spaces.
runfile	the name of the file to receive the executable output.
listfile	the name of the file to receive the listing.
lib-list	a list of library modules to be searched.
/switch	optional switches, which may be placed following any of the response entries (just before any of the commas or after the <lib-list>, as shown).

To select the default for a field, type a second comma with no spaces between the two commas.
For example:

```
LINK CAR+TIRE+WHEEL+BRAKE/P/M,, CARLIST,OURLIB.LIB
```

This command causes LINK to be loaded, then the object modules CAR.OBJ, TIRE.OBJ, WHEEL.OBJ, and BRAKE.OBJ are loaded. LINK pauses as a result of using the /P switch. When you press any key LINK will:

Link the object modules.
Produce a global symbol map (the /M switch).
Default to CAR.EXE Run file.
Create a list file named CARLIST.MAP.
Search the Library file OURLIB.LIB.

Method 3: Response File

To start LINK using Method 3, type:

LINK @[*d:*] [*path*]*filename*[*.ext*]

Enter a response file specification preceded by an @ symbol. To use this option, first create a response file which contains several lines of text, each of which is the response to a LINK prompt. A response file contains answers to the LINK prompts and may also contain any of the switches. The responses must be in the same order as the LINK prompts discussed in Method 1. If necessary, a long response to the Object Modules or Libraries prompt may be typed on several lines by using a plus sign (+) to continue the same response onto the next line.

Use switches and command characters in the response file the same way they are used for responses typed on the terminal keyboard.

When naming a response file, the use of filename extensions is optional. Method 3 permits the command that starts LINK to be entered from the keyboard or within a batch file without requiring you to take any further action.

When the LINK session begins, each prompt will be displayed in order with the responses from the response file. If the response file does not contain answers for all the prompts, (either filenames, the semicolon command character or carriage returns), LINK will display the prompt which does not have a response, then wait for you to type a legal response. When a legal response has been typed. LINK continues the link session.

For example:

```
CAR TIRE WHEEL BRAKE
/PAUSE/MAP
CARLIST
OURLIB.LIB
```

This response file tells LINK to load the four object modules named CAR, TIRE, WHEEL, and BRAKE. LINK pauses to permit you to swap disks. When you press the <ENTER> key, the output files will be named CAR.EXE and CARLIST.MAP. LINK will search the library file OURLIB.LIB, and will use the default settings for the switches.

SAMPLE LINKER SESSION

This sample shows you the type of information that is displayed during a LINK session.

In response to the DOS prompt, type:

LINK

The system displays the following messages and prompts (your answers are underlined):

```
    Microsoft Object Linker V.2.00
(C) Copyright 1982 by Microsoft Inc.

Object Modules [.OBJ]: SAMPLE FILEA
Run File [SAMPLE.EXE]: /MAP
List File [NULL.MAP]: PRN/LINE
Libraries [.LIB]:
```

By responding **PRN** to the List File prompt, you can redirect your output to the printer.

By specifying **/MAP**, you get both an alphabetic listing and a chronological listing of public symbols.

By specifying the **/LINE** switch, you get a listing of all line numbers for all modules. (The **/LINE** switch can generate a large volume of output.)

By pressing <ENTER> in response to the Libraries prompt, an automatic library search is performed.

MEMO

CHAPTER 10
DEBUG

DEBUG

INTRODUCTION

DEBUG is an interactive tool developed for debugging user software. DEBUG allows the user to check, modify, and test binary programs without having to recompile each and every time.

DEBUG requires a minimum of 13K bytes of memory. DEBUG can operate with one disk drive, however most practical applications require two drives.

Any DEBUG command can be terminated (aborted) by issuing a <Ctrl> <C>. The screen display scrolling action can be temporarily halted to facilitate reading by issuing a <Ctrl> <S>. The screen will resume scrolling by pressing any other key.

STARTING THE DEBUG PROGRAM

A DEBUG session can be started by typing:

DEBUG

The DEBUG utility will then prompt the user for commands by displaying a hyphen (-). Since no filename has yet been specified the DEBUG commands NAME and LOAD can be used to select the file to be debugged. (See the NAME and LOAD commands).

Another way to begin is to include the filename when starting DEBUG:

DEBUG [*d:*] [*path*] [*filename*[*.ext*]] [*parm1*] [*parm2*]

parm1 and *parm 2* are optional parameters.

DEBUG sets registers and flags to the following initial values:

Segment registers CS, DS, ES, and SS are initially set to the first contiguous segment of memory after the end of the DEBUG program.

Instruction Pointer (IP) is set to the value 0100H.

Stack Pointer (SP) is set to the end of the segment, or the bottom of the transient portion of the program loader, whichever is lower. The segment size at offset 6 is reduced by hex 100 to allow for a stack of that size.

Remaining registers (AX, BX, CX, DX, BP, SI, and DI) are set to zero. If you specify a filespec when starting the DEBUG, the CS register contains the size of the file in bytes. If the file is greater than 64K, the size is contained in registers BX and CX (the high portion in BX).

Flags are set to their cleared values. (See the Register command.)

Default disk transfer address is set to 80H in the code segment.

COMMAND PARAMETERS

Parameter	Definition
address	All numeric values are hexadecimal. Enter a one- or two-part designation in one of the following formats: An alphabetic segment register designation, a colon, then an offset value: CS:0100 A segment address, a colon, then an offset value: 4BA:0100 An offset value only: 100 In this case, the default segment is used. CS is the default segment for the commands G, L, T, U, and W. DS is the default segment for all other commands. **NOTE:** Memory locations specified by the address must be valid otherwise unpredictable results will occur.
byte	A one or two character hexadecimal value.
drive	A one-digit hexadecimal value to indicate which drive a file will be loaded from or written to. The valid values are 0–3. These values designate the drives as follows: 0=A:, 1=B:, 2=C:, 3=D:.

filespec A filename which can include a drive designation and filename extension.
(See the Name command.)

list One or more byte and/or string values. For example:

```
ECS:100 42 45 52 54 41
```

range *address address*

A range can be specified by a lower bound address and an upper bound address.
For example:

```
DS:510 590
```

address L *value*

A range can also be specified by a lower bound address and a hexadecimal number specifying the number of bytes to be included.
For example:

```
CS:400 L 11
```

NOTE: The limit for range is hex 10000. To specify a value of 10000 hex in four hexadecimal characters, enter 0000 (or 0).

registername See the Register command.

DEBUG

record A 1- to 3-digit hexadecimal value used to indicate the logical record number on the disk and the number of disk sectors to be written or loaded. Logical records correspond to sectors. However, their numbering differs since they represent the entire disk space.

string Any number of characters enclosed in quote marks. Quote marks may be either single (') or double ("). If the delimiter quote marks must appear within a <string>, the quote marks must be doubled. For example, the following strings are legal:

> 'This is a "string" is okay.'
> 'This is a ' 'string' ' is okay.'

However, this string is illegal:

> 'This is a 'string' is not.'

Similarly, these strings are legal:

> "This is a 'string' is okay."
> "This is a " "string" " is okay."

value Either a hexadecimal value up to 4 digits used to specify the port number or the number of times to repeat a command.

TABLE OF DEBUG COMMANDS

Command	Purpose	Format
Assemble	Assembles statements	A [*address*]
Compare	Compares memory	C *range address*
Dump	Displays memory	D [*address*] or D [*range*]
Enter	Changes memory	E *address* [*list*]
Fill	Changes memory blocks	F *range list*
Go	Executes with optional breakpoints	G [=*address*] [*address* [*address*...]]
Hex	Hexadecimal add-subtract	H *value value*
Input	Reads/displays input byte	I *value*
Load	Loads file or absolute diskette sectors	L [*address* [*drive record record*]]
Move	Moves memory block	M *range address*
Name	Defines files and parameters	N *filespec* [*filespec*...]
Output	Sends output byte	O *value byte*
Quit	Ends DEBUG program	Q
Register	Displays registers/flags	R [*registername*]

(cont.)

DEBUG

Search	Searches for characters	S *range list*
Trace	Executes and displays	T [=*address*] [*value*]
Unassemble	Unassembles instructions	U [*address*] or U [*range*]
Write	Writes file or absolute dis-k sectors	W [*address* [*drive record record*]]

DEBUG

DEBUG COMMANDS

The following notes are common to all DEBUG commands:

A command is specified by a single letter in upper or lower case. One or more parameters may follow.

Delimiters are only required between two consecutive hexadecimal values. Delimiters may be used to separate commands and parameters. The following commands are equivalent:

```
dcs: 200 210
d cs:200 210
d,cs: 200,210
```

Press <Ctrl> <Break> to abort any command.

Use the <ENTER> key to activate a command.

To stop the scrolling action of the display use <Ctrl> <NumLock>. Press any other key to resume scrolling.

The DEBUG program prompt is the hyphen (-).

The DEBUG program resides on your System Disk.

Assemble

Syntax: A[*address*]

Purpose: To assemble Assembler language statements directory into memory. DEBUG supports standard 8086/8087/8088 assembly language.

Comments: DEBUG accepts hexadecimal numeric input. Beginning at the specified address all assembler statements are placed in contiguous locations (addresses) in memory. The default address is the location following the last instruction assembled by a previous Assemble command or the area at CS:0100 if no previous Assemble command was used. When all statements have been entered, press <ENTER> to return to the DEBUG prompt.

DEBUG responds to invalid statements by displaying:

```
^ Error
```

All numeric values entered are hexadecimal and can be entered as 1–4 characters.

The segment override mnemonics are CS:, DS:, ES:, and SS:.

Prefix mnemonics must be entered in front of the opcode to which they refer. They may be entered on a separate line.

String manipulation mnemonics must state the string size. For example:

MOVSW

to move word strings.

MOVSB

to move byte strings.

The mnemonic for the far return is RETF.

The assembler will automatically assemble short, near, or far jumps and calls depending on byte displacement to the destination address. These may be overridden with the NEAR or FAR prefix. For example:

```
0100:0500 JMP 502            ; a 2 byte short jump
0100:0500 JMP NEAR 505       ; a 3 byte near jump
0100:0505 JMP FAR 50A        ; a 5 byte far jump
```

The NEAR prefix can be abbreviated to NE, but the FAR prefix cannot be abbreviated.

Operands must specify either word or byte memory locations. The type must be stated with the prefix "WORD PRT" or "BYTE PTR". DEBUG accepts the abbreviations "WO" and "BY". For example:

```
NEG BYTE PRT [128]
DEC WO [SI]
```

To display whether an operand refers to a memory location or to an immediate operand, DEBUG uses the convention that operands enclosed in square brackets refer to memory. For example:

```
MOV AX,21; Load AX with 21H
MOV AX,[21]; Load AX with the contents of memory
location 21H
```

DEBUG

The pseudo-instructions DB and DW are available. The DB opcode assembles byte values. The DW opcode assembles word values. For example:

```
DB  1,2,3,4, "THIS IS AN EXAMPLE"
DB  'THIS IS A QUOTE:" '
DB  "THIS IS A QUOTE:' "

DW  1000,2000,3000,"BACH"
```

For 8087 opcodes the WAIT or FWAIT prefix must be explicitly specified. For example:

```
FWAIT FADD ST,ST(3) ; This line will
                    ; assemble
                    ; an FWAIT
                    ; prefix.

LD TBYTE PRT [BX] ;This line will not.
```

Assemble supports all forms of the register indirect commands. For example:

```
ADD     BX,34[BP+2].[SI-1]
POP     [BP+DI]
PUSH    [SI]
```

All opcode synonyms are supported. For example:

```
LOOPZ       100
LOOPE       100

JA          200
JNBE        200
```

DEBUG

Compare

Syntax: C *range address*

Purpose: Used to compare two blocks of memory.

Comments: Ranges are used to select two blocks for comparison. If the areas of memory are identical the prompt reappears. If there are differences, their addresses and contents are displayed, in the form:

addr1 *byte1* *byte2* *addr2*

addr1 byte1 refers to the location and contents of the mismatching locations in range.
byte2 addr2 refers to the byte found in address.

```
C100, 1FF 300
```
or
```
C100L100 300
```

Example: The 100H bytes of memory beginning at DS:100 are compared with the 100H bytes beginning at DS:300.

DEBUG

Dump

Syntax: D [*address*]

or

D [*range*]

Purpose: Used to display the contents of a block of memory.

Comments: The hexadecimal and ASCII representations of the specified block are displayed. In the ASCII portion, unprintable characters are indicated by a period (.).

Each line begins on a 16-byte boundary and shows 16 bytes. There is a hyphen between the 8th and 9th bytes.

> **NOTE:** If the starting address of the dump is not on a boundary, the first line may have fewer than 8 or 16 bytes. In this case, the second line of the dump begins on a boundary.

The Dump command has two format options:

Option 1 D *address*

or

D

The contents are displayed starting with the specified address.

If no address is specified, the starting location is the address following the last address displayed by a previous D command. Each subsequent D displays the bytes immediately following those last displayed. If no previous D command has been issued, 0100H is used as an offset.

NOTE: If you specify only an offset for the starting address, the segment in DS register is used.

Option 2 D *range*

The contents of the specified address range are displayed.

Example: If you type the command:

DCS: 100 110

DEBUG displays the dump in the following format:

04BA:0100 42 45 52 54 41 ... 4E 44 TOM SAWYER

If you type the command:

DCS:100 L 20

the display is formatted as described above, but 20H bytes are displayed.

DEBUG

Enter

Syntax: E *address* [*list*]

Purpose: The Enter command can be used in two ways:

To replace the contents of one or more bytes, starting at the specified address, with the values contained in the list.

Displays and allows modification of bytes in a sequential manner.

Comments: If the optional *list* of values is typed, the replacement of byte values occurs automatically.

For examples,

```
E ds:100 E3 "abc" 7E
```

filles ds:100 through ds:104 with E3 61 62 63 7E.

If the *address* is typed without the optional *list*, DEBUG displays the address and its contents, and waits for your input. At this point, the Enter command waits for you to perform one of the following actions:

1. Replace a byte value with a value you type. Simply type the value after the current value. If the value typed in is not a legal hexadecimal value or if more than two digits are typed, the illegal or extra character is not echoed.

2. Press the <SPACEBAR> bar to advance to the next byte. To change the value, simply type the new value as described in (1.) above. If you space beyond an 8-byte boundary, DEBUG starts a new display line with the address displayed at the beginning.

DEBUG

3. Type a hyphen (-) to return to the preceding byte. If you decide to change a byte behind the current position, typing the hyphen returns the current position to the previous byte. When the hypen is typed, a new line is started with the address and its byte value displayed.

4. Press the <ENTER> key to terminate the Enter command. The <ENTER> key may be pressed at any byte position.

Example:

```
E cs:100
```

DEBUG displays:

```
04BA:0100 EB._
```

To change EBH to 41H, enter 41. 1.

```
04BA:0100 EB.41_
```

To step through the subsequent bytes, press the <SPACEBAR> to see:

```
04BA:0100  EB.41  10.  00.  BC._
```

To change hex BC to hex 42, enter 42.

```
04BA:0100  EB.41  10.  00.  BC.42_
```

To back up and change the hex 10 to hex 6F enter two hyphens and the hex 6F.

```
04BA: 0100  EB.41  10.  00.  BC.42_
04BA: 0102  00._
04BA: 0101  10.6F_
```

Press the <ENTER> key to end the Enter command. The hyphen (-) prompt will be displayed.

Fill

Syntax: F *range list*

Purpose: This command is used to specify a range of bytes in memory for replacement by a list.

Comments: If the number of bytes specified in the memory range is greater than the number of bytes in the list then DEBUG will attempt to repeat the list until the memory bytes are filled. If the list is bigger than the number of bytes specified in memory then the list is truncated.

> **NOTE:** If you specify only an offset for the starting address of the range, the segment in DS register is used.

Example:

```
F04BA:100 L 5 F8 "ABC" 2D
```

Memory locations 04BA:100 through 04BA:104 are filled with the 5 bytes in the list. The ASCII values of the list characters are stored. Locations 100–104 will contain F8 41 42 43 2D.

DEBUG

Go

Syntax: G [=*address*] [*address* [*address...*]]

Purpose: This command is used to execute the current program. The program will stop and display registers, flags etc. when a BREAKPOINT is encountered.

Comments: The Go command has two format options:

Option 1 G [=*address*]

This option executes the program you are debugging when you don't set breakpoints.

When you test the program with different parameters each time, use this option. (Refer to the Name command.) If not using =address be sure the CS:IP values are set properly before executing the G command.

Option 2 G [=*address*] *address* [*address...*]

This option will halt program execution at a breakpoint. The current registers and flag status are displayed. Up to ten breakpoints are allowed.

NOTES: When a program has terminated normally ("Program terminated normally" message has been displayed), you have to reload the program to execute it again.

Breakpoints must be set only at addresses containing the first byte of an 8088 opcode.

The stack pointer must be valid and have 6 bytes available to the Go command.

If you specify only an offset for a breakpoint, the segment in the CS register is used.

Example: GCS:7550

The program in memory executes up to the address 7550 in the CS segment. DEBUG displays the registers and flags. To resume program execution at the instruction after the breakpoint, type the GO command.

DEBUG

Hex

Syntax: H *value value*

Purpose: This command displays the sum and difference of the two hexadecimal values.

Example:

```
H 0A 8
0012 0002
```

The hexadecimal sum of 000A and 0008 is 0012, and their difference is 0002.

Input

Syntax: I *value*

Purpose: Inputs and displays (in hexadecimal) one byte from the port specified by *value*.

Example:

```
I2F8
```

If the byte at the port is 42H then DEBUG displays:

```
42
```

DEBUG

Syntax: L [*address* [*drive record record*]]

Purpose: This command is used to load a file into memory. It can also be used to load specified sectors from a disk.

Comments: Set BX:CX to the number of bytes read. The file must have been named either when DEBUG was started or with the N command. Both the DEBUG invocation and the N command format a filenmae properly in the normal format of a file control block at CS:5C.

If the L command is typed without any parameters, DEBUG loads the file into memory beginning at address CS:100 and sets BX:CX to the number of bytes loaded. If the L command is typed with an address parameter, loading begins at the memory *address* specified. If L is typed with all parameters, absolute disk sectors are loaded, not a file. The *record*s are taken from the *drive* specified (the drive designation is numeric here 0=A:, 1=B:, 2=C:, etc.); DEBUG begins loading with the first *record* specified, and continues until the number of sectors specified in the second *record* have been loaded.

Example: Assume that the following commands are typed:

```
A>DEBUG
-NFILE.COM
```

Now, to load FILE.COM, type:

```
L
```

DEBUG

DEBUG loads the file and then displays the DEBUG prompt. Assume that you want to load only portions of a file or certain records from a disk. To do this, type:

```
L04BA:100 2 0F 6D
```

DEBUG then loads 109 (6D hex) records beginning with logical record number 15 into memory beginning at address 04BA:0100. When the records have been loaded, DEBUG simply returns the − prompt.

If the file has a .EXE extension, it is relocated to the load address specified in the header of the .EXE file: the *address* parameter is always ignored for .EXE files. The header itself is stripped off the .EXE file before it is loaded into memory. Thus the size of an .EXE file on disk will differ from its size in memory.

If the file named by the Name command or specified when DEBUG is started is a .HEX file, then typing the L command with no parameters causes DEBUG to load the file beginning at the address specified in the .HEX file. If the L command includes the option *address*, DEBUG adds the *address* specified in the L command to the address found in the .HEX file to determine the start address for loading the file.

DEBUG

Move

Syntax: M *range address*

Purpose: This command is used to move the bloc of memory specified by *range* to the location beginning at *address*.

Comments: Overlapping moves (moves where part of the block overlaps some of the current addresses) are always performed without loss of data. Addresses that could be overwritten are moved first.

Moves from higher addresses to lower addresses, move data beginning at the block's lowest address and work towards the highest. Moves from lower addresses to higher addresses, move data beginning at the block's highest address and work towards the lowest. The sequence of the move is important because the MOVE command copies the data from one area into another, in the sequence described, and writes over the new addresses.

Example:

```
MCS:100 110 CS:500
```

DEBUG first moves address CS:110 to address CS:510; then CS:10F to CS:50F, and so on until CS:100 is moved to CS:500. The DUMP command can be used to view the results of a MOVE command.

Name

Syntax: N *filespec* [*filespec*...]

Purpose: This command specifies the filename of the file to be used with later LOAD and GO commands. This command must be used when DEBUG is started without a specified filename.

Comments: All specified filespecs and parameters are placed exactly as entered, including delimiters, in a parameter save area at CS:81. CS:80 containing the number of characters entered.

Example:

```
-NPROGA.EXE
-L
-NFILE1.DAT FILE2.DAT
-G
```

The effect of these commands is:

NAME sets PROGA.EXE as the filename.
LOAD loads PROGA.EXE into memory.
NAME is used again to specify the parameters to be used by PROGA. EXE.
GO executes PROGA.EXE as if FILE1.DAT and FILE2.DAT has been typed at the DOS command level.

> **NOTE:** If a WRITE command were executed at this point, the file would be saved with the filename FILE1.DAT. To avoid this problem, use the NAME command before LOAD or WRITE.

The regions of memory that can be affected by the NAME command are:

CS:5C	File Control Block for file 1.
CS:6C	File Control Block if file 2 is specified.
CS:80	Count of characters in the NAME command.
CS:81	Actual characters typed in the NAME command.

To execute PROG as if the following line had been typed:

```
PROG PARM1 PARM2/C
```

enter:

```
DEBUG PROG.COM
-NPARM1 PARM2/C
-G
-
```

DEBUG

Output

Syntax: O *value byte*

Purpose: This command sends the *byte* to the output port specified by *value*.

Example:

```
O2F8 4F
```

Sends the byte value 4FH to output port 2F8H.

DEBUG

Quit

Syntax: Q

Purpose: This command ends the DEBUG program.

Comments: Note that the QUIT command stops DEBUG operation without saving the current program. The Write command must be used to save the program before stopping DEBUG.

DEBUG returns to the DOS command level.

Example:

```
-Q
A>
```

DEBUG

Register

Syntax: R [*registername*]

Purpose: This command can be used to:

Display and modify the hex contents of any single register.

Display the hex contents of all registers, flags, and the next instruction to be executed.

Display and modify the flag settings.

Comments: If no *registername* is specified, the R command dumps the register save area and displays the contents of all registers and flags.

The valid register names are (IP and PC both refer to the instruction pointer):

AX	BP	SS
BX	SI	CS
CX	DI	IP
DX	DS	PC
SP	ES	F

The flags are:

FLAG NAME	TO SET	TO CLEAR
Overflow	OV	NV
Direction	DN (decrement)	IN (increment)
Interrupt	EI (enabled)	DI (disabled)
Sign	NG (negative)	PL (plus)
Zero	ZR	NZ
Auxillary Carry	AC	NA
Parity	PE (even)	PO (odd)
Carry	CY	NC

DEBUG

Example: R

DEBUG displays all registers, flags and the instruction for the current location.

RF

DEBUG displays all the flags. To change the value of a flag, just enter a valid flag value. To exit the R command press <ENTER>. The flags you did not list values for will remain unchanged.

R BX

Displays the contents of register BX. To change the contents of the register, just enter a 1–4 character hexadecimal value. To exit the R command press <ENTER>. The register contents will be unchanged if you do not enter a new value.

DEBUG

Search

Syntax: S *range list*

Purpose: This command is used to search for a specified character string within a range in memory.

Comments: The address of each occurrence of a match is displayed. When no match is found a hyphen (-) is displayed.

Example:

```
S CS:100 110 41
```

Searches addresses from CS:100 through CS:110 for hex 41.

The addresses of matches are displayed:

```
04BA:0210
04BA:021D
```

Trace

Syntax: T [=*address*] [*value*]

Purpose: This command executes instructions and displays the contents of all registers and flags after each instruction.

Comments: The number in *value* determines the number of instructions to execute. Pressing <Ctrl> <NumLock> will suspend the scrolling so that you can study the registers and flags for any particular instruction.

Example:

```
T
```

Assume that the current position is 04BA:011A. DEBUG will display:

```
AX=0E00 BX=00FF CX=0007 DX=01FF
SP=039D BP=0000 SI=005C DI=0000
DS=04BA ES=04BA SS=04BA CS=04BA
IP=011A NV UP DI NG NZ AC PE NC
04BA:011A   CD21     INT    21
```

```
T=110A 10
```

DEBUG executes sixteen instructions (10 in hex). The registers and flags for each instruction are displayed.

DEBUG

Unassemble

Syntax: U[*address*]

or

U[*range*]

Purpose: This command is used to take the specified contents of memory and translate them into their equivalent assembler command. The addresses and hexadecimal values are displayed with assembler-like statements.

Comments: If the contents of the addresses specified do not contain valid instruction codes then errors will occur.

If you do not include a *range* of bytes, DEBUG translates 20 hex bytes. If no address is specified then the default address is the address after the last one specified by a U command. With this technique it is possible to produce continuous translations.

If you include the *range*, all the instructions in the *range* will be unassembled.

Example: U 04BA:0100 L10

DEBUG displays:

```
04BA:0109    65   DB   65
04BA:010A    63   DB   63
04BA:010B    69   DB   69
04BA:010C    66   DB   66
04BA:010D    69   DB   69
04BA:010E    63   DB   63
04BA:010F    61   DB   61
```

If you type U 04BA:0100 0108 DEBUG will display:

```
04BA:0100  206472    AND[SI+72],AH
04BA:0103  69        DB          69
04BA:0104  7665      JBE 016B
04BA:0106  207370    AND [BP+DI+70],DH
```

DEBUG

Write

Syntax: W [*address*[*drive record record*]]

Purpose: This command is used to write the contents of the memory area specified to the indicated drive.

Comments: Writing to absolute sectors is very dangerous because you are bypassing the file handler. Be sure that the sectors specified on the disk DO NOT contain data that you want to save.

The current file must have been named either when DEBUG was started or with the NAME command.

If W is used with no parameters:
    BX:CX must be set to the number of bytes to be written.
    The file is written from CS:100.

If W is used with only an address:
    The file is written beginning at the address.

If W is used with parameters:
    The write begins from the address specified.
    The file is written to the drive specified.
        (Drive designation is numeric
            0=A:
            1=B:
            2=C: etc.)

A single W command can write a maximum of hex 80 sectors.

Example: W

DEBUG writes the file to disk and then displays the DEBUG prompt.

WCS:100 1 37 2B

DEBUG writes the contents of memory from CS:100. The data starts at sector hex 37 and continues for hex 2B sectors. The information is written on Drive B.

CHAPTER 11

CONTROL OF SCREEN AND KEYBOARD

CONTROL

INTRODUCTION

DOS allows user programs to issue special control characters. By use of these characters the programmer can move the cursor up, down, forward, backward, or to any addressable position on the screen. Other character sequences allow: screen and line erasure, change of screen display characteristics, and keyboard reassignment.

NOTE: These control character sequences must be issued through DOS function calls 1, 2, 6, and 9. Furthermore, the extended screen and keyboard control device driver (ANSI.SYS) must be present in memory. The following command, when included in the configuration file (CONFIG.SYS) will add the control device driver to DOS:

```
DEVICE=ANSI.SYS
```

(Be aware that this file will increase the amount of memory used by DOS.)

DOS uses the default value if no value is specified or zero is specified.

#(parameter) is the numberic value to be used in the control sequence.

When programming these control sequences use the 1 byte code for ESC (hex 1B) and not the three characters "ESC". For example, the control sequence ESC [2;10H would be coded as:

```
e200 1B "[2;10H"
```

CONTROL

MODE OF OPERATION

Set Graphics Rendition SGR

ESC [# ;...; # m

This sequence causes the character/screen #;...;#m attributes to change according to the numberic values as specified below (these values are enabled until a subsequent issuance of SGR):

Parameter	Function
0	All attributes OFF (normal white on black)
1	Bold On (high intensity)
4	Underscore On
5	Blink On
7	Reverse video On
8	Concealed On (invisible)
30	Black foreground
31	Red foreground
32	Green foreground
33	Yellow foreground
34	Blue foreground
35	Magenta foreground
36	Cyan foreground
37	White foreground
40	Black background
41	Red background
42	Green background
43	Yellow background
44	Blue background
45	Magenta background
46	Cyan background
47	White background

Set Mode **SM**

ESC [= # h or ESC [= h or ESC [= 0 h or ESC [= ? 7 h	These sequences set screen width/display or according to the numeric values as specified below:

Parameter	Function
0	40×25 black and white
1	40×25 color
2	80×25 black and white
3	80×25 color
4	320×200 color
5	320×200 black and white
6	640×200 black and white
7	automatic wrap-around (at end of line)

Reset Mode **RM**

ESC [= # l or ESC [= l or ESC [= 0 l or ESC [? 7 l	These sequences function in the same manner as SM (see above) except that parameter 7 will cause no-wrap at end of line (excess characters are ignored).

CONTROL

ERASING

Erase Entire Display **ED**

ESC [2 J	This sequence causes erasure of the entire screen. The cursor is left at the home position.

Erase Entire Line **EL**

ESC [k	This sequence causes erasure of everything from the current cursor position to the end of the line (to the right).

CONTROL

CURSOR CONTROL

Cursor Position **CUP**

ESC [# ; # H

This sequence causes the cursor to be moved to the screen position addressed by the two numeric values. The first gives the line number that the cursor is to be positioned at and the second gives the column number. The default value is one. If no values are specified, the cursor will move to the home position.

Cursor Up **CUU**

ESC [# A

This sequence causes the cursor to be moved directly up according to the number of lines specified by the numeric value. The default value is one. If the cursor is already on the top of the screen then this sequence has no effect.

Cursor Down **CUD**

ESC [# B

This sequence causes the cursor to be moved directly down according to the number of lines specified by the numeric value. The default value is one. If the cursor is already at the bottom of the screen then this sequence has no effect.

Cursor Forward **CUF**

ESC [# C

This sequence causes the cursor to be moved directly to the right according to the number of columns specified by the numeric value. The default value is one. If the cursor is already at the right edge then this sequence has no effect.

Cursor Backward — CUB

ESC [# D — This sequence causes the cursor to be moved directly to the left according to the number of columns specified by the numeric value. The default value is one. If the cursor is already at the left edge then this sequence has no effect.

Horizontal and Vertical Position — HVP

ESC [# ; # f — This sequence causes the cursor to be moved to the screen position addressed by the two numeric values. The first gives the line number that the cursor is to be positioned at and the second gives the column number. The default value is one. If no values are specified the cursor will move to the home position.

Device Status Report — DSR

ESC [6 n — This sequence causes the console driver to issue a CPR sequence. (see CPR)

Cursor Position Report — CPR

ESC [# ; # R — This sequence causes the current cursor position to be reported through the standard input device. The first numeric value gives the current line and the second numeric value gives the current column.

Save Cursor Position — SCP

ESC [s — This sequence causes the current cursor position to be saved.
(see RCP to restore the cursor position)

Restore Cursor Position — RCP

ESC [u — This sequence restores the cursor to its position prior to the issuance of the SCP sequence.

KEYBOARD KEY REASSIGNMENT

ESC [# ; # ; ... # p
or ESC ["string" ; p
or ESC [# ; " string " ; # ; # ; "string" ; # p
or any other combination of strings and decimal numbers

These control sequences reassign keyboard keys. The numeric value is the ASCII code for a particular key. The key to be mapped is specified by the first numeric value. Subsequent numeric values specify the sequence of ASCII character codes to be generated when the mapped key is depressed. Any DOS command can be invoked or assigned to a key.
Extended ASCII codes are indicated by specifying zero (NUL) as the first numeric value and the appropriate code as the second numeric value.

For example:

1. Reassign the A key to the B key.
   Reassign the a key to the b key.
   Reassign the B key to the A key.
   Reassign the b key to the a key.

ESC [6 5 ; 6 6 p	A becomes B
ESC [9 7 ; 9 8 p	a becomes b
ESC [6 6 ; 6 5 p	B becomes A
ESC [9 8 ; 9 7 p	b becomes a

2. Reassign the F10 key to a DIR command followed by a carriage return:

   ESC [0 ; 6 8 "dir" ; 1 3 p

   (0;68 is the extended ASCII code for the F10 key. 13 decimal is a carriage return.)

CONTROL

CHAPTER 12

INPUT AND OUTPUT OPTIONS

IN/OUT

REDIRECTION OF INPUT AND OUTPUT DEVICES

The following command lines are used to redirect standard input and output. Standard input to a program is from the keyboard and standard output is to the screen. Standard input can be redirected to a file from which a program will draw data.

*** **WARNING** ***

> The program will terminate if it is allowed to read more data than is contained in the file. If the program terminates, press <Ctrl> <Break> to return to the DOS prompt.

To redirect standard input:

<[*d:*] [*path*] *filename*

Standard output can be redirected to any file.

To create/open a file and assign it to standard output:
(the previously created file will be erased)

>[*d:*] [*path*] *filename*

To create/open a file and assign it to standard output:
(output will be added to the end of a previously created file)

>>[*d:*] [*path*] *filename*

Example:

The following command will direct the display of DIR to the printer:

```
DIR >PRN
```

The following command will direct the display of DIR to the file LISTIT:

```
DIR >LISTIT
```

The following command will cause the program PROG1 to obtain its input from the datafile INFO.TXT:

```
PROG1 <INFO.TXT
```

NOTE: If standard input/output is not performed via DOS function calls in an application program (e.g. writing directly to the video buffer) then redirection will not work.

PIPING INPUT AND OUTPUT

Programs may be chained so that the output specified program can be stored in a temporary datafile for input to another specified program. (Temporary files can be recognized in the root directory of the default drive by the form %PIPEx.$$$.) The vertical bar (¦) is used to chain two programs for automatic redirection of standard input and output.

Example:

Assume the program named SORT reads all it's standard input, sorts it, and then writes it to the standard output device. The command:

```
DIR ¦ SORT
```

generates a sorted directory listing. All standard output generated by the DIR command will be sent to the standard input of the SORT program.

To send the sorted directory to a file, you would enter:

```
DIR ¦ SORT >FILE
```

To create a file to contain only the directory entries for sub-directories, you would enter:

```
DIR ¦ FIND "DIR" ¦ SORT >FILE
```

FILTERS

Filters are programs/commands that receive data from the standard (or redirected) input device, process the data, and send the output to the standard (or redirected) output device.

SORT, FIND, and MORE are filters supplied with DOS. Other filters can be created by writing programs that read from standard input and write to standard output.

SORT	Will sort lines of text in alpha order.
FIND	Will search files for specified strings of text.
MORE	Used to output full screens of text with —MORE— displayed at the bottom of the screen.

Example:

```
SORT  <FILE1  >FILEOUT
```

SORT will read the file FILE1, sort the lines within it, and write the sorted output to file FILEOUT.

With the piping feature, a filter can receive its input from the output of another command, or to send its output to the input of another command. For example,

```
DIR ¦ SORT
```

Causes the output listing from the DIR command to be used by SORT as its input. The listing will be sorted and the result displayed on the screen.

NOTE: If standard input/output is not performed via DOS function calls in an application program (e.g. writing directly to the video buffer) then filters will not work.

IN/OUT

APPENDICES

CONTENTS

APPENDIX A

USER DIAGNOSTICS

USER DIAGNOSTICS

Built-into the ROM of your Sr. Partner, is a series of tests that help pinpoint problems in the operation of the computer. You run the User Diagnostic tests whenever your computer fails to operate or respond correctly.

During the course of the tests you may encounter certain error messages. Always write down or print out the message, and the procedure you were performing when the message appeared. Sometimes a simple phone conversation can allow your Panasonic dealer to interpret the message and help you get your computer operating again.

Even if the computer needs to be serviced, a list of error messages and their location within the testing sequence can be helpful in accelerating repairs.

Checking the Brightness Control

Normally, when you have turned on the power and inserted your System Disk, your computer will cycle through its series of power-on tests. When these tests are completed the system requests the date and the time and then displays the A> prompt.

If the date prompt does not appear, check the brightness control. When the control is too low you may not be able to read the screen. The brightness control is located on the rear panel of your system unit.

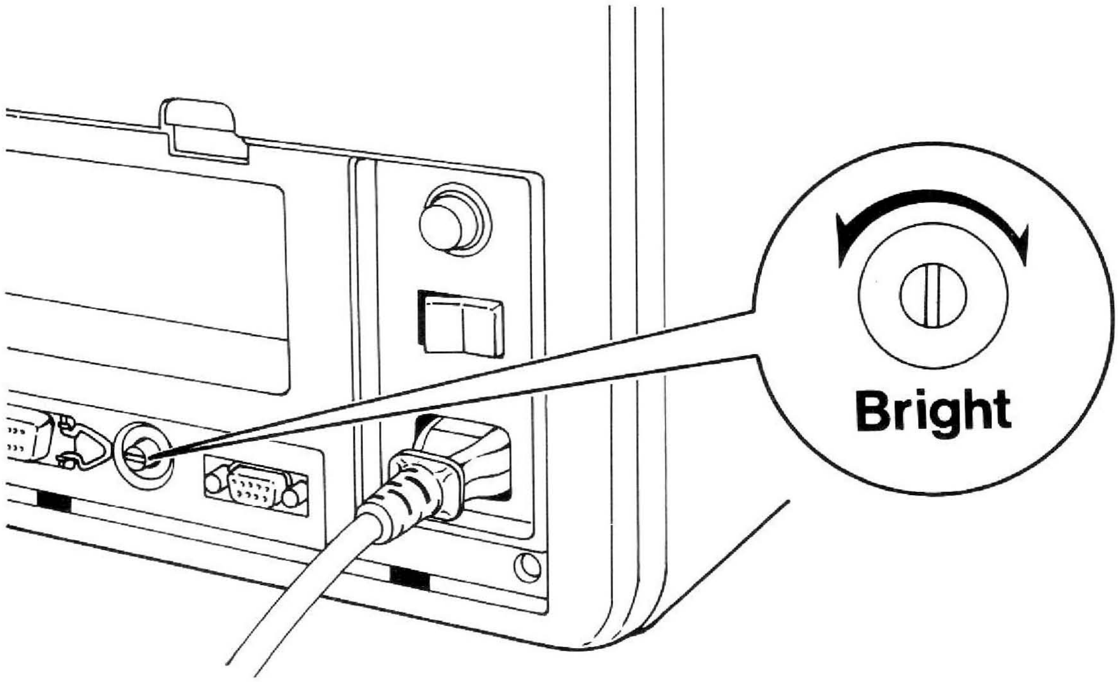


Fig. A-1. BRIGHTNESS CONTROL

Checking the Electrical Outlet

The date may not appear because the computer is not receiving electrical power. Set the power switch to OFF and unplug the computer. Now use a working appliance such as a lamp or television to check the outlet.

If the outlet is functioning, disconnect and reconnect the AC power cord to the back of the system unit. Plug the computer back into the wall outlet and turn ON the power.

If the computer still fails to respond, you may need to change the fuse.

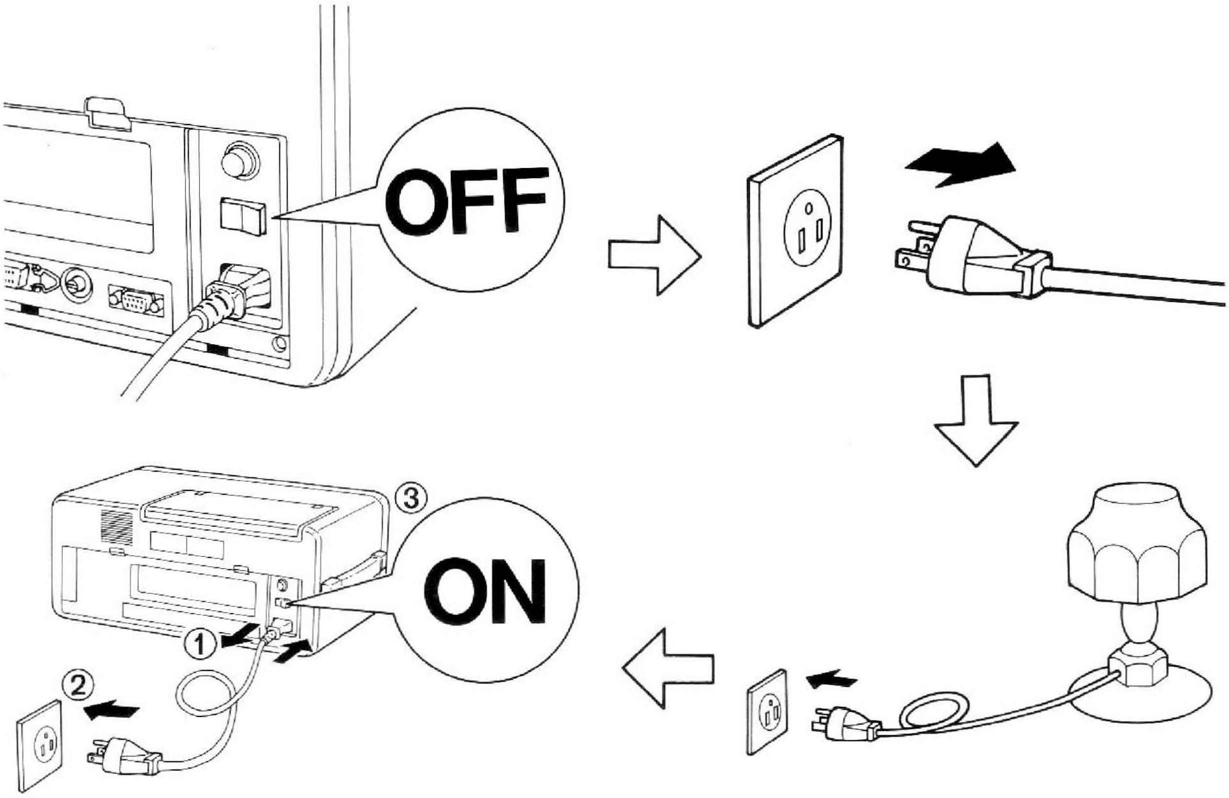


Fig. A-2. CHECKING THE ELECTRICAL OUTLET

Replacing the Fuse

Your Sr. Partner uses a single, 125 V 3A fuse. Before replacing the fuse make sure that the main power switch is OFF. Unplug the computer from the wall outlet.

Unscrew the fuse by turning the cap in a counterclockwise direction. The current fuse is removed with the cap.

Insert a new fuse in the cap.

Reinsert the cap and tighten the connection by screwing in a clockwise direction. The cap should be screwed in firmly.

Reconnect the power and try the computer again. If it still does not respond, make a note of your procedure and contact the nearest Panasonic dealer for service.

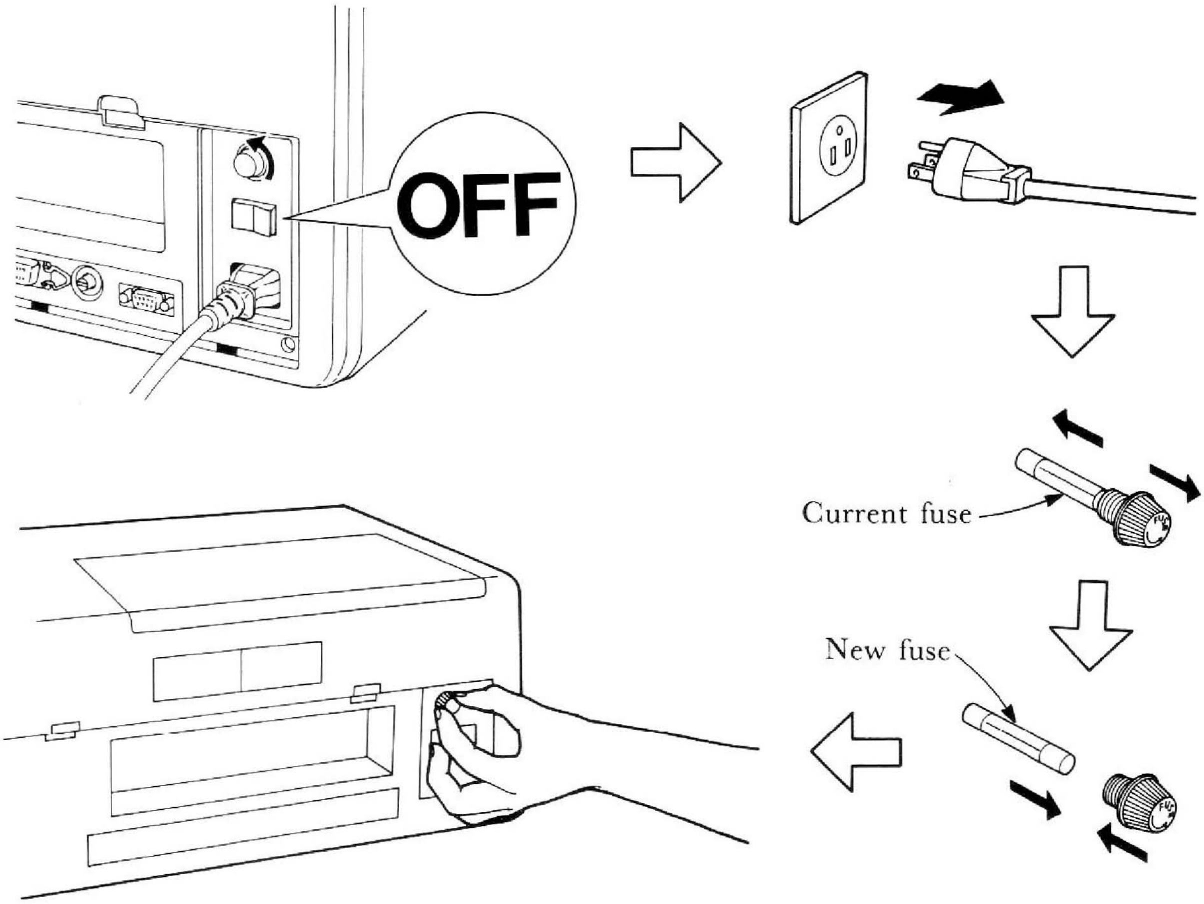


Fig. A-3. REPLACING THE FUSE

Entering Diagnostic Testing

To enter the diagnostic testing:

STEP 1 Turn ON the main power switch, **then** insert your System Disk into Drive A.

STEP 2 After you have entered the date and time the A> prmopt will apear.

YOU TYPE:

diag

After the title is displayed,

Do you want printed output ?(Y/N)

is displayed.

If you press y, messages on the screen are printed out.

STEP 3 SCREEN DISPLAYS:

DIAGNOSTICS MENU

1. TEST ALL DEVICES
2. TEST AUTOMATICALLY
3. CHANGE MENU
4. EXIT
5. MAIN BOARD
6. xxxKB MEMORY
7. KEYBOARD
8. GRAPHICS VIDEO
9. 2 FLOPPY DISK DRIVE(S) or 9. 1 FLOPPY DISK DRIVE(S)
10. 2 PARALLEL PORT(S)
11. 1 SERIAL PORT(S)

SELECT MENU : 1

The System Diagnostics Menu

This is the menu for the testing procedures. (A **menu** is a list of choices). This menu lists the various components of your Sr. Partner system.

If you chose:

1. TEST ALL DEVICES	Each device is checked in sequence. The computer will wait to proceed after each test. Notice that this selection is automatically programmed as the default choice. Press <ENTER> to begin the testing.
2. TEST AUTOMATICALLY	All the devices will be tested without further input from you. This testing will continue to operate until you press <Ctrl> <Break> to stop the cycle.
3. CHANGE MENU	Allows you to add or delete items from the menu. See A-29.
4. EXIT	If you decide you do not wish to proceed with the diagnostic testing, press this selection to return to DOS.
5. MAIN BOARD	Tests the main board.
6. xxxKB MEMORY	Tests the memory. (the number of KB will change depending on the configuration of your system.)

7. KEYBOARD	Tests the keyboard.
8. GRAPHICS VIDEO	Tests the built-in circuit for the built-in display or an optional color monitor, and the display or color monitor itself.
9. 2 FLOPPY DISK DRIVE(S)	Tests the drives connected to the system. The number of drives will be 1 or 2 depending on the configuration of your system.
10. 2 PARALLEL PORT(S)	Tests the internal printer (parallel port 1) and the built-in circuit for an optional printer (parallel port 2). The second test checks the internal capacity to run an additional printer if it is not connected. Thus the menu will read "2 parallel port(s)" even if no additional printers are part of the system configuration.
11. 1 SERIAL PORT(S)	Tests the built in RS232C port.

To select the option on the menu:

Move the inverse display to your desired option by the < ↑ > key (upward) or < ↓ > key (downward). At the same time the value of "SELECT MENU" at the bottom of the menu will be automatically changed. (You can type the value of your desired option instead of the < ↑ > or < ↓ > key.) Then press <ENTER>.

In most cases, you will test the entire system, using the first option on the menu. The testing example in this manual will cover the entire testing procedure as contained in the first selection. To run tests on specific devices, see the sections dealing with that test.

If the correct menu has appeared on your screen, proceed to STEP 4.

If the screen does not display the menu, or the menu does not match your system configuration, write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

Terminating Diagnostic Testing

If during the course or the testing you wish to stop the procedure, hold down <Ctrl> and press the <Break> key. You may need to finish the specific tests being run (complete the keyboard testing sequence, for example) but at the conclusion of the current test, you will return to the menu.

You must hold <Ctrl> and press <Break> to stop the automatic testing cycle started with Option 2.

The Diagnostics Testing Sequence

Once the correct menu is displayed, you are ready to begin the testing procedure.

STEP 4 YOU TYPE:

<ENTER>

Test all devices will start. One short beep will be heard. If the beep isn't heard, a checksum of ROM BIOS is wrong.

SCREEN DISPLAYS:

```
MAIN BOARD TEST
DIP SW STATUS=xxxx

Test done

Press any key when ready
```

Where xxxx depends on the configuration of your system.

If your screen matches this display, proceed to STEP 5.

If your screen does not look like this, write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 5

Press any key

SCREEN DISPLAYS:

```
MEMORY TEST
Test done

Press any key when ready
```

If your screen matches this display, proceed to step 6.

If your screen does not look like this, write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 6 Press any key.

SCREEN DISPLAYS:

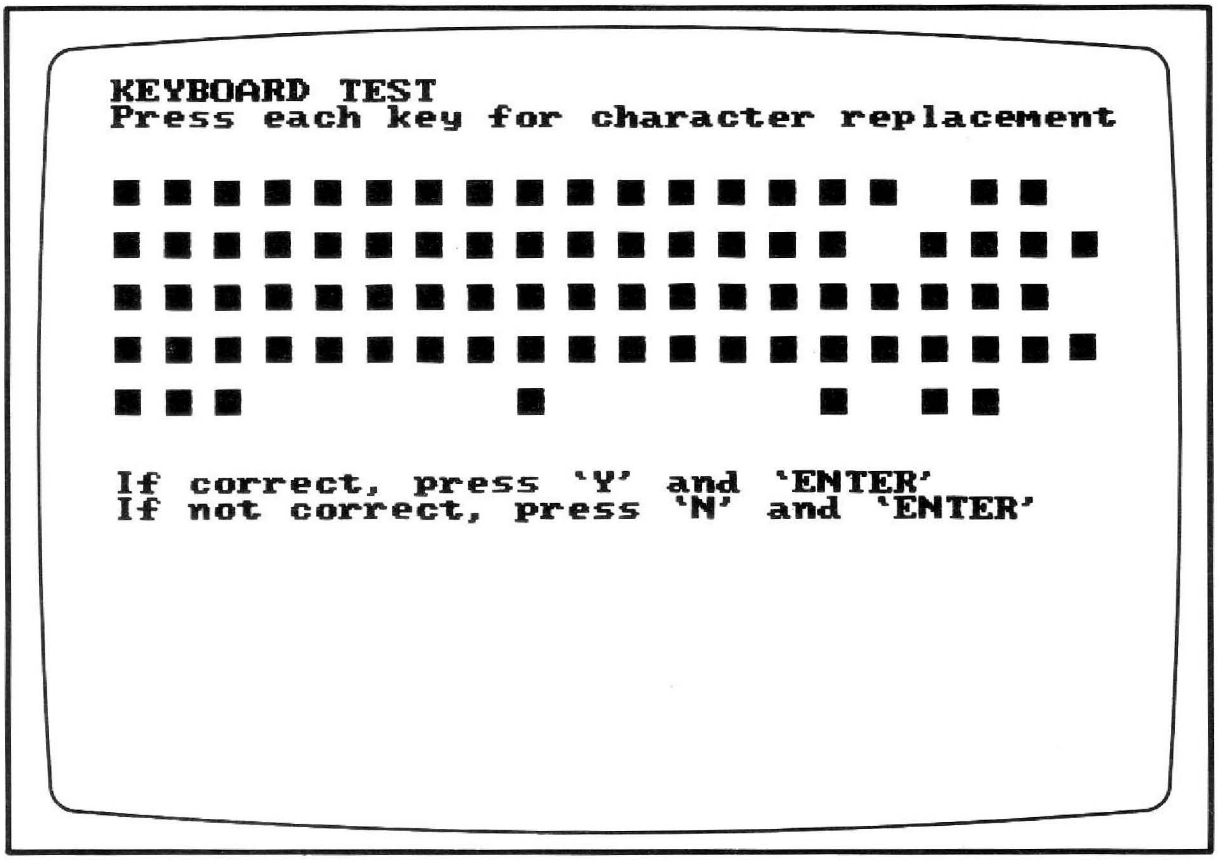


Fig. A-4. KEYBOARD TEST-1

When you hold down each key, the character will be displayed inversely. When you release it, the character will be displayed normally.

YOU TYPE:

hold down each key

and

release it.

```
KEYBOARD TEST
Press each key for character replacement
1 2 e 1 2 3 4 5 6 7 8 9 0 - = ←      n s
3 4 → Q W E R T Y U I O P [ ]   7 8 9 -
5 6 c A S D F G H J K L ; ' ` e 4 5 6
7 8 ↑ \ Z X C V B N M , . / ↑ * 1 2 3 +
9 0 a         _           c   0 .

If correct, press 'Y' and 'ENTER'
If not correct, press 'N' and 'ENTER'
```

Fig. A-5. KEYBOARD TEST-2

STEP 7 If your screen matches this display,

YOU TYPE:

Y(y)

and press <ENTER>

SCREEN DISPLAYS:

Keyboard is OK

Press any key when ready

Proceed to STEP 8.

If your screen does not look like this, write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

YOU TYPE:

N(n)

and press <ENTER>

SCREEN DISPLAYS:

Keyboard has failed

Press any key when ready

NOTE: You can proceed with testing even if the keyboard has failed.

STEP 8 Press any key.

SCREEN DISPLAYS:

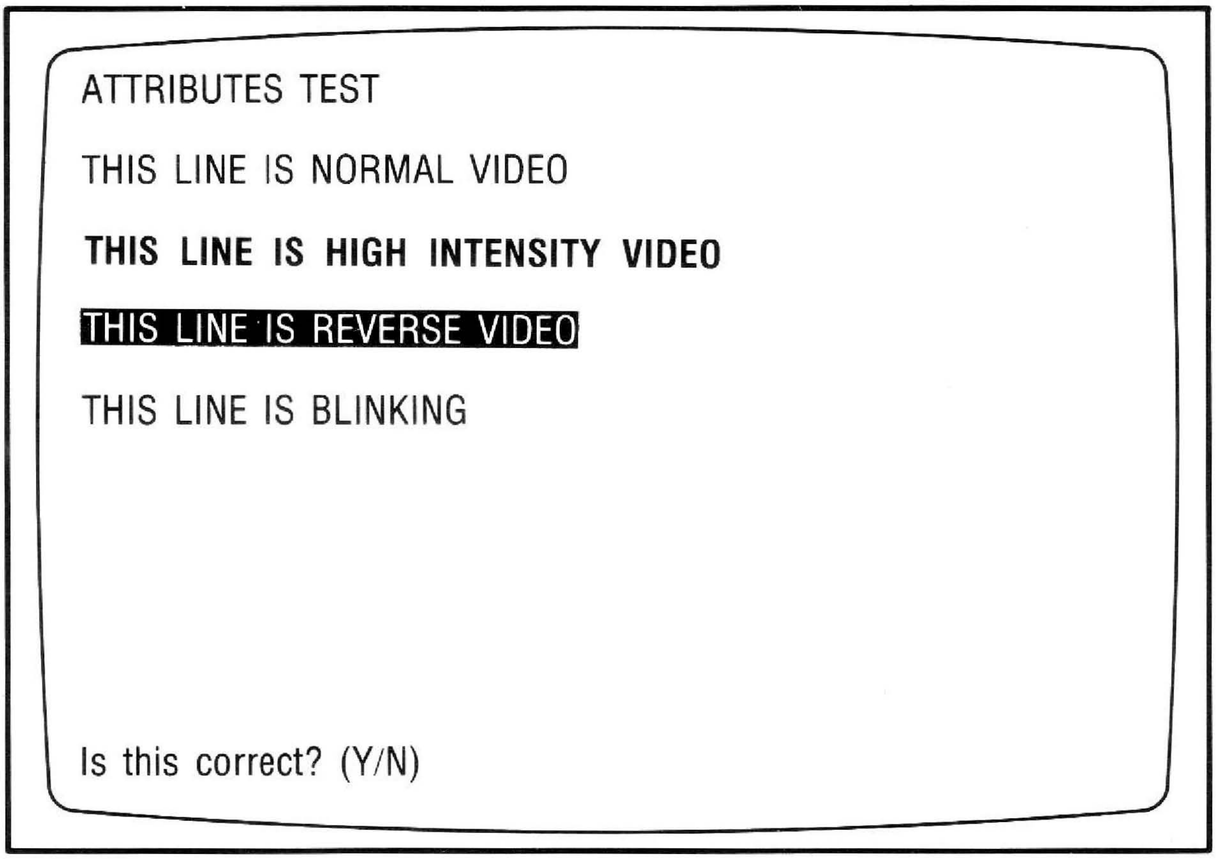


Fig. A-6 SCREEN MODE

If your screen matches this dispay, press y and proceed to STEP 9.

If your screen does not look like this, press n and write down your procedure, any error messages displayed, any call your Panasonic dealer for service.

STEP 9 SCREEN DISPLAYS:

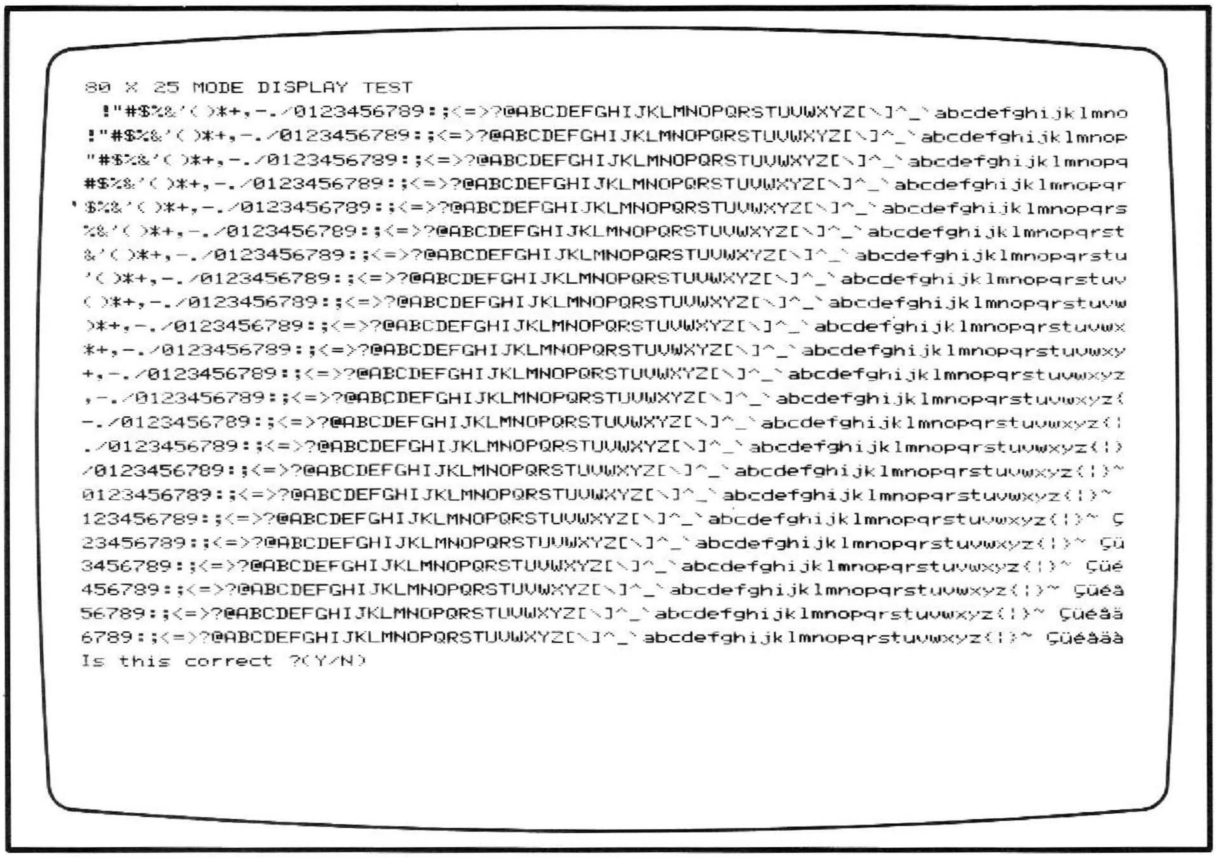


Fig. A-7. MODE DISPLAY 80×25

If your screen matches this display, press y and proceed to STEP 10.

If your screen does not look like this, press n and write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 10 SCREEN DISPLAYS:

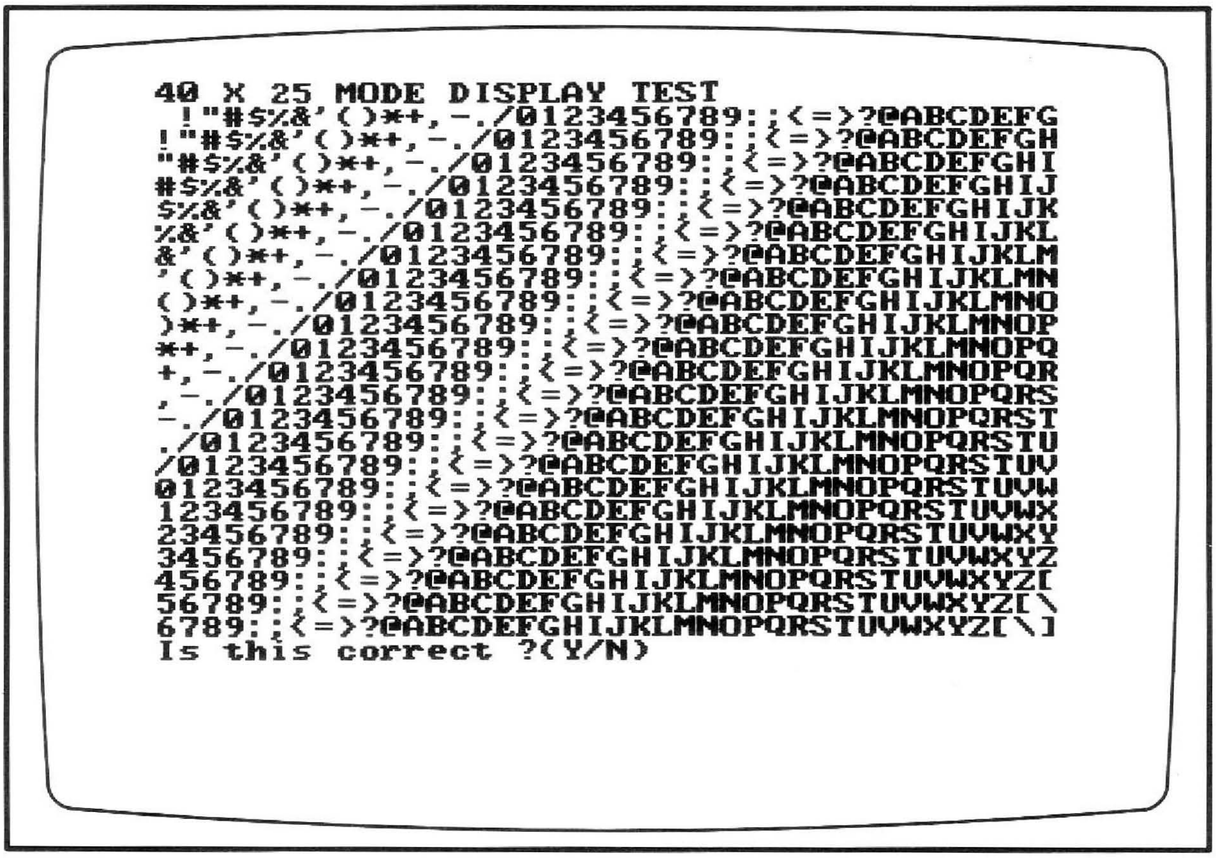


Fig. A-8. MODE DISPLAY 40×25

If your screen matches this display, press y and proceed to STEP 11.

If your screen does not look like this, press n and write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 11 SCREEN DISPLAYS:

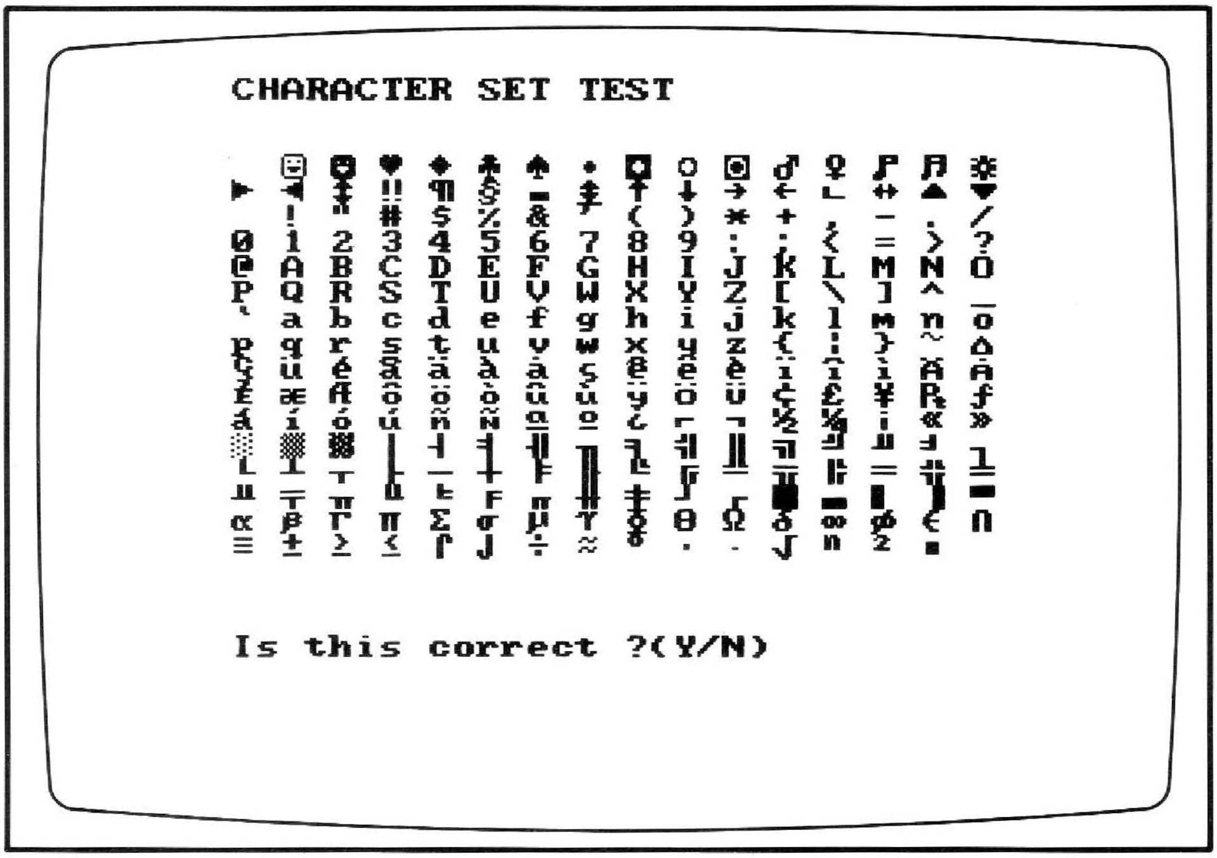


Fig. A-9. CHARACTER SET SCREEN

If your screen matches this display, press y and proceed to STEP 12.

If your screen does not look like this, press n and write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 12 SCREEN DISPLAYS:

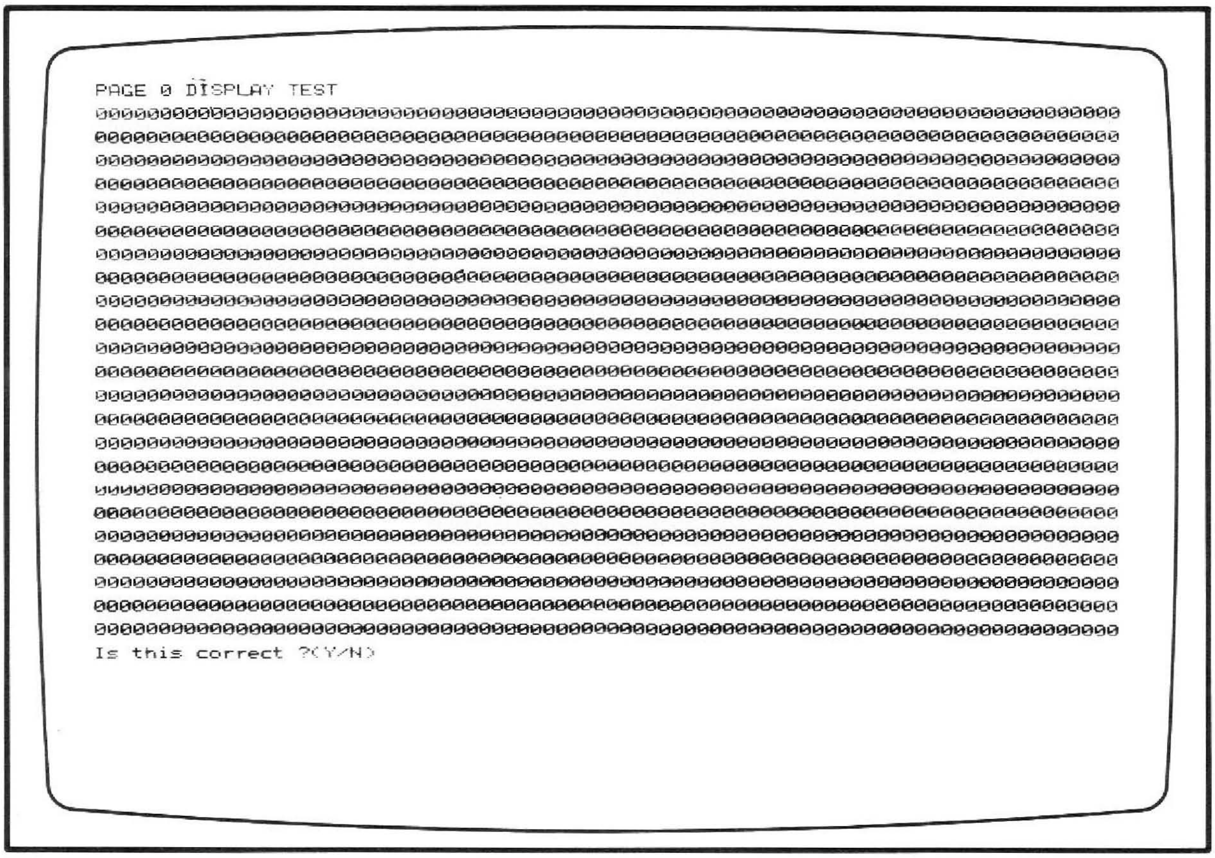


Fig. A-10. PAGE 0 DISPLAY

If your screen matches this display, press y and proceed to STEP 13.

If your screen does not look like this, press n and write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 13 The diagnostics testing now cycles through a series of page displays. Press y or n to move through the pages until page 3 is displayed.

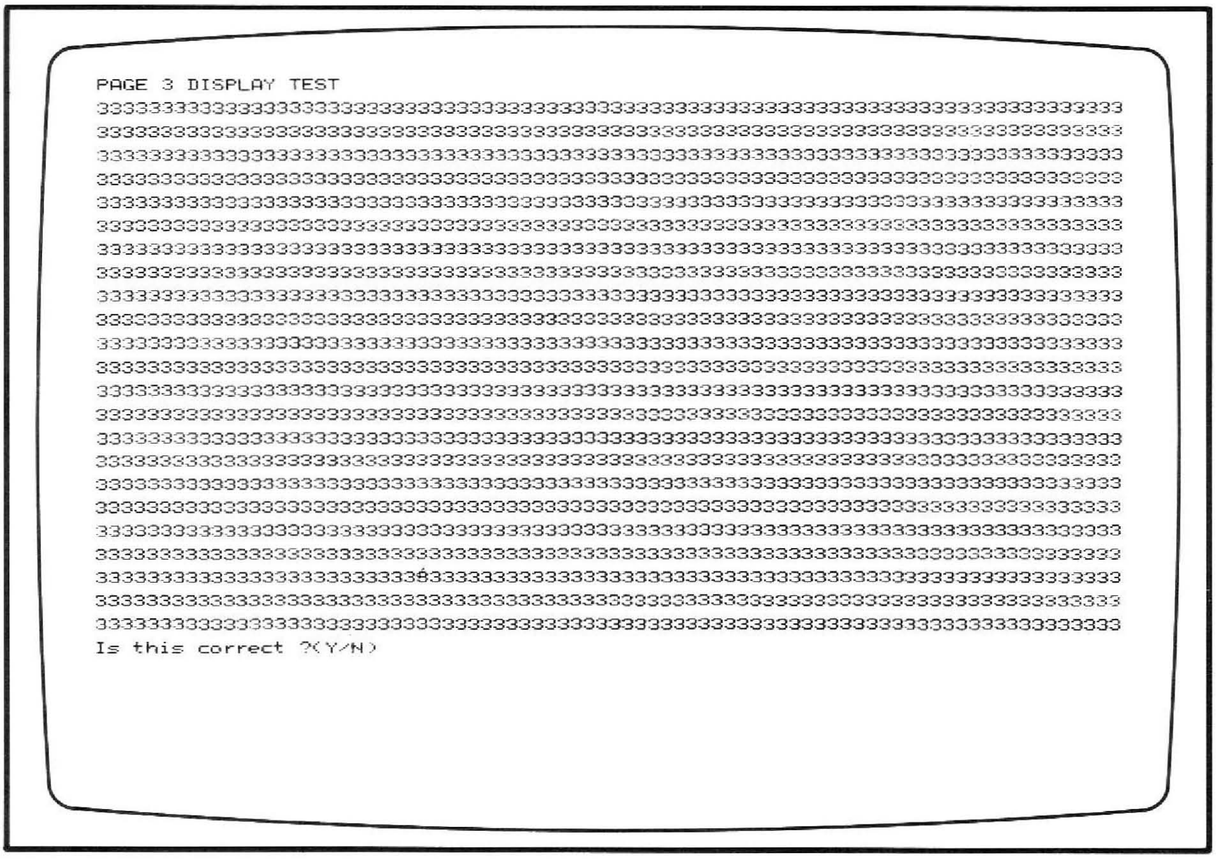


Fig. A-11. PAGE 3 DISPLAY

If your screen matches this display, proceed to STEP 14.

If your screen does not look like this, write down your procedure and any error messages displayed, and call your Panasonic dealer for service.

STEP 14 Brightness/color pattern is displayed.

NOTE: If you test with the built-in monitor, these tests will appear in shades of green. If you have a color monitor they will appear in color.

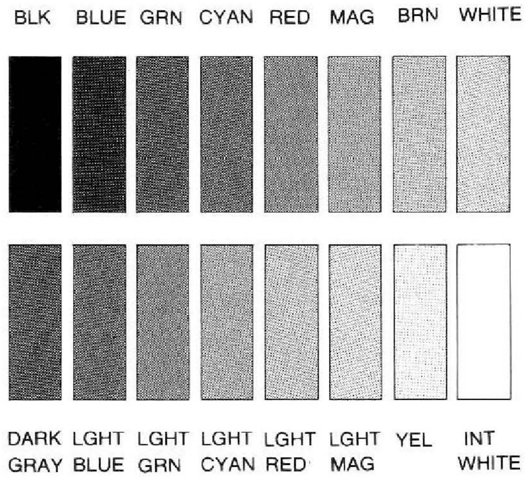


Fig. A-12. COLOR TEST PATTERN SCREEN

If your screen matches this display, press y and proceed to STEP 15.

If your screen does not look like this, press n and write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 15 SCREEN DISPLAYS:

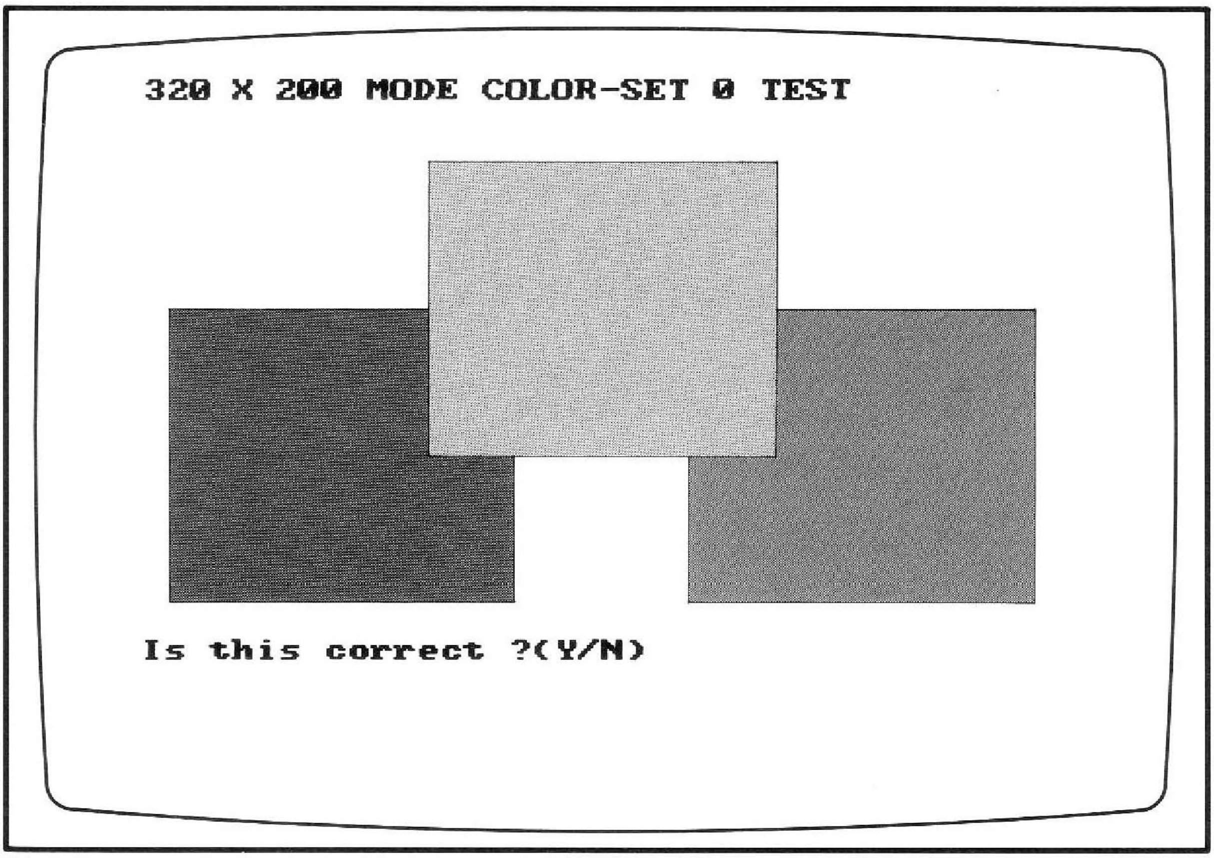


Fig. A-13. COLOR SET 0 TEST

If your screen matches the display, proceed to STEP 16.

If your screen does not look like this, write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 16 SCREEN DISPLAYS:

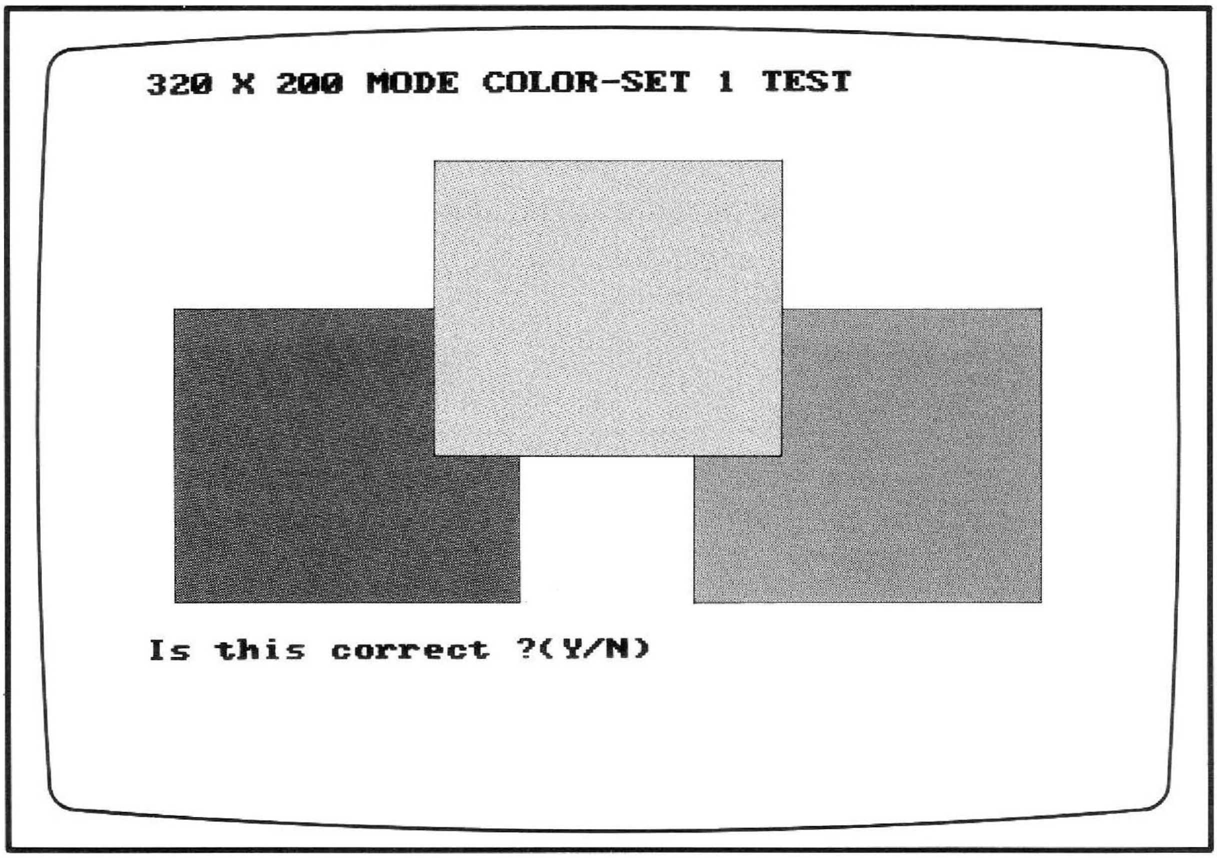


Fig. A-14. COLOR SET 1 TEST

If your screen matchs this display, proceed to STEP 17.

If your screen does not look like this, write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 17 SCREEN DISPLAYS:

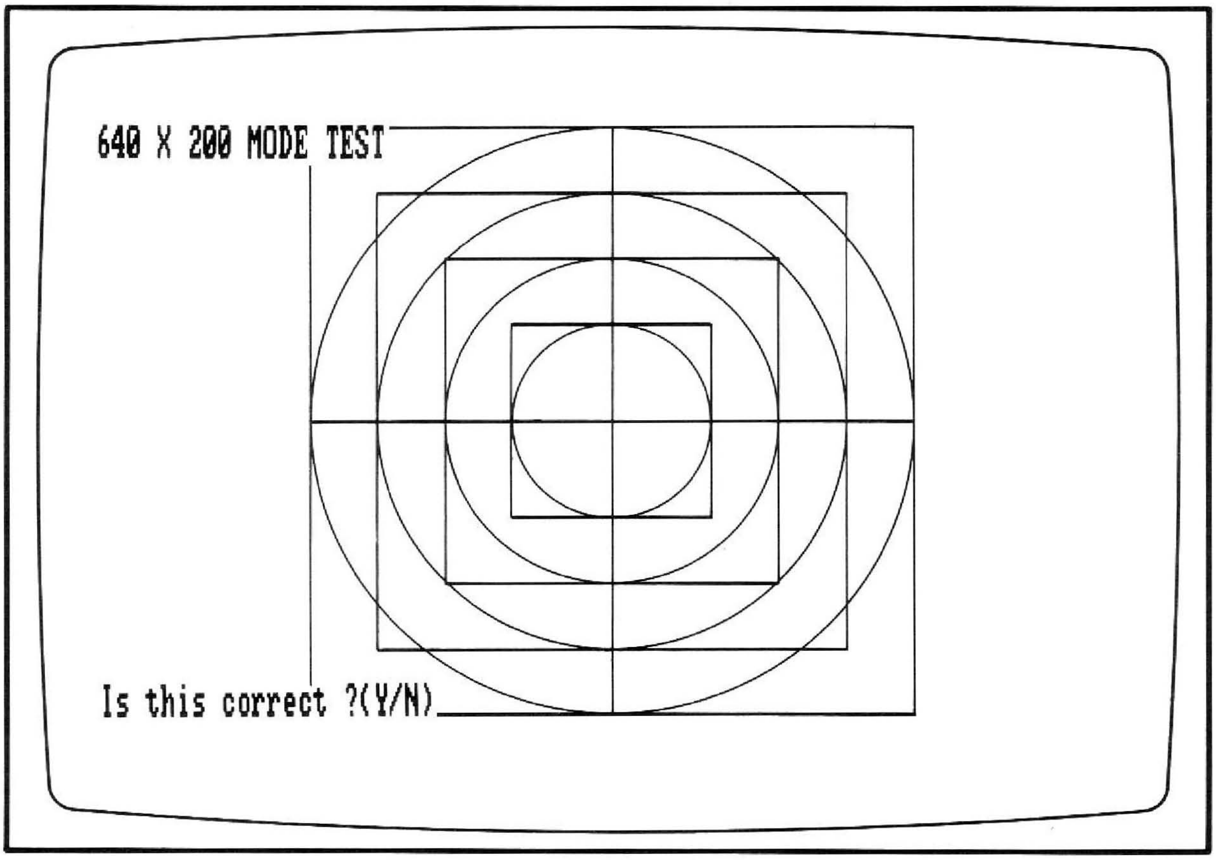


Fig. A-15. 640×200 MODE TEST

If your screen matches this display, proceed to STEP 18.

If your screen does not look like this, write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 18

Press any key.

SCREEN DISPLAYS:

```
********************************
*CONTENTS OF DISK(S) WILL BE DESTROYED*
********************************

Insert scratch disk into each drive
Press any key when ready
```

or

```
Insert scratch disk into drive(s)
Press any key when ready
```

This is a test of your disk drives. Druing this test the drives write to the disk. That means that any information on the disks will be written over and destroyed.

IF YOU HAVE NOT ALREADY DONE SO, REMOVE YOUR SYSTEM DISK FROM DRIVE A.

Insert a new disk in Drive A (and Drive B is you have a dual drive system.) These disks will be formatted as the part of the testing process.

When the disks are ready, proceed to STEP 19.

STEP 19

YOU TYPE:

Press any key

You will hear the drives operating and the In Use Indicators will come on.

SCREEN DISPLAYS:

DRIVE-A FORMATTING DISK
DRIVE-A SPEED STATUS=xxxx

DRIVE-A SEQUENTIAL SEEK TEST
DRIVE-A STATUS=xxxx xxxx xxxx xxxx......

DRIVE-A RANDOM SEEK TEST
DRIVE-A STATUS=xxxx xxxx xxxx xxxx ......

If you have two drives the display will continue with:

DRIVE-B FORMATTING DISK
DRIVE-B SPEED STATUS=xxxx
DRIVE-B SEQUENTIAL SEEK TEST
DRIVE-B STATUS=xxxx xxxx xxxx xxxx ......

DRIVE-B RANDOM SEEK TEST
DRIVE-B STATUS=xxxx xxxx xxxx xxxx ......

Press any key when ready.

If your screen matches this display, proceed to STEP 20.

If your screen does not look like this, and contains any error message, write down your procedure, any error messages displayed, and call your Panasonic dealer for service.

STEP 20

This test checks the internal printer and the built-in circuit which allows your Sr. Partner to support an external printer. This test will be performed even if no external printer is connected to the system.
Press any key.

SCREEN DISPLAYS:

PARALLEL PORT(S) TEST

TESTING INTERNAL PARALLEL PORT

You will hear the printer operating.

```
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmno
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnop
"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopq
#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqr
$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrs
%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrst
&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstu
'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuv
()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvw
)*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwx

THIS LINE IS NORMAL PRINTING
T H I S   L I N E   I S   E X P A N D E D   P R I N T I N G
THIS LINE IS COMPRESSED PRINTING
NEXT LINE IS NORMAL-DENSITY BIT IMAGE PRINTING

NEXT LINE IS DUAL-DENSITY BIT IMAGE PRINTING
```

Fig. A-16. PRINTER OUTPUT (1)

Your output should match this illustration.

SCREEN DISPLAYS:

TESTING EXTERNAL PARALLEL PORT

If no error, screen displays

Connect Printer
Press any key when ready

If you wish to test an external printer press any key after connecting it. If you don't wish, press <Ctrl> <Break>, then you will be returned to the System Diagnostics Menu, or press any key and ignore error messages.

```
TESTING EXTERNAL PARALLEL PORT
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdef
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefg
"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefgh
#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghi
$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghij
%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijk
&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijkl
'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklm
()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmn
)*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmno
```

Fig. A-17. PRINTER OUTPUT (2)

Your output should match this illustration.

STEP 22

This test checks the built-in serial interface and if you attach the plug for a loop back test, the serial port can be tested.

NOTE: The plug for a loop back test must be wired as follows:

Connect pin 2 and pin 3
Connect pin 4, pin 5 and pin 8
Connect pin 6, pin 20 and pin 22

SCREEN DISPLAYS

TESTING INTERNAL SERIAL PORT

If no error, screen displays

Need loopback test ?(Y/N)

If you press y, screen displays,

Connect loopback plug.
Press any key when ready

Press any key then the loopback test will begin.

If you press n, the loopback test will not be performed.

STEP 23 Press any key.

This returns you to the System Diagnostics Menu.

If you attach a joystick board to the option slot in Sr. Partner, you fill find "JOYSTICK" on the System Diagnostics Menu.

A sequence of the joystick test is as follows:

STEP 24a SCREEN DISPLAYS:

JOYSTICK TEST
Press each button—If correct press 'Y', if not correct press 'N'

STEP 24b SCREEN DISPLAYS:

A1-BUTTON was pressed

STEP 24c SCREEN DISPLAYS:

A2-BUTTON was pressed

If your screen looks like this, press Y and proceed to STEP 24d.

If your screen does not match this display, press N and proceed to STEP 24d.

STEP 24d SCREEN DISPLAYS:

JOYSTICK-A installed ?(Y/N)

If joystick A is installed, press Y.

If joystick A is not installed, press N.

STEP 24e SCREEN DISPLAYS:

JOYSTICK-B installed ?(Y/N)

If joystick B is installed, press Y.

If joystick B is not installed, press N.

If the joystick(s) is normal:

COUNT STATUS=xxxxxxxx

Press any key when ready

If the joystick(s) is wrong:

JOYSTICK FAILED-STATUS=xxxx

Press and key when ready

The diagnostics testing is now completed. You may choose to perform the tests again, perform only partial tests or exit the testing. To return to DOS, select option 4 on the menu.

Change Menu
You can add or delete options to or from the System Diagnostics Menu by Option 3, CHANGE MENU.

When you select Option 3 on the System Diagnostics Menu, the following will be appeared. If you delete the option press d, if you append, press a.

Press 'D'(lete) or 'A'(ppend)_

If you press d or a, the change menu will be displayed. Select your desired option.

DIAGNOSTICS MESSAGES

Various messages or error messages appeared in User Diagnostics are listed below.

NOTE: n_i (i=1, 2,...) in messages is a nibble noted by hexadecimal. Nibble consists of 4 bits as follows:

$n=D_3D_2D_1D_0$

For example,

$n_1=5$ (0101) then $D_3=0$, $D_2=1$, $D_1=0$, $D_0=1$
$n_1=D$ (1101) then $D_3=1$, $D_2=1$, $D_1=0$, $D_0=1$

MAIN BOARD TEST

DIP SW STATUS=$n_1n_2n_3n_4$

nibble	bit	meaning
n_1	D_3	1: 1 disk drive, 0: 2 disk drive
	D_2	1: Graphics Video, 0: Monochrome Video
	D_1	Output data from Timer 2 in Programmable Interval Timer
	D_0	1: Enable 8087, 0: Disable 8087
n_2	D_0–D_3	Memory size (See the following table)
n_3	D_3	1:80 column display, 0:40 column display

Memory size	D_3	D_2	D_1	D_0
128K bytes	0	0	1	0
256K bytes	0	1	1	0
320K bytes	1	0	0	0
384K bytes	1	0	1	0
448K bytes	1	1	0	0
512K bytes	1	1	1	0

SYS ERR1

System area cannot be saved into VRAM area.

PPIB ERROR

Port B register of Programmable Peripheral Interface (8255) has failed.

PIT0 ERROR

Register of Programmable Interval Timer (8253) has failed.

PIT1 ERROR

Count of Programmable Internal Timer (8253) has failed.

DMA0 ERROR

Register of Direct Memory Access Controller (8237) has failed.

DMA1 ERROR

Refresh counter of Direct Memory Access Controller (8237) has failed.

RAM0 ERROR

Dynamic RAM data has failed.

RAM 1 ERROR

Dynamic RAM address has failed.

RAM2 ERROR

Dynamic RAM refresh has failed.

PREG ERROR

Page register (LS670) has failed.

PIC0 ERROR

Register of Programmable Interrupt Controller (8259) has failed.

PIC1 ERROR

Handling of Programmable Interrupt Controller (8259) has failed.

MEMORY TEST

RAM n_1n_2 $n_3n_4n_5n_6$ $n_7n_8n_9n_{10}$

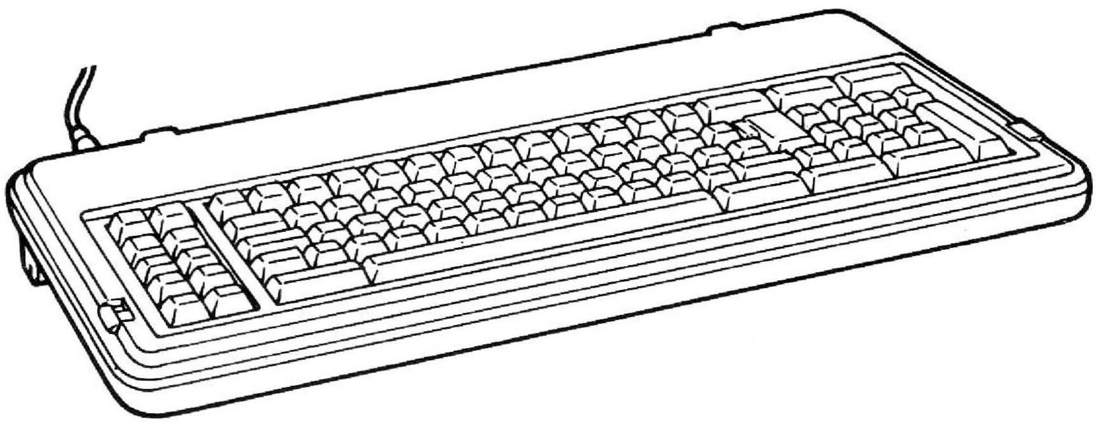
$n_1=0$: RAM data error. Data bus or DRAM chip has failed.

$n_2n_3n_4n_5n_6$ is absolute address where data error occured.
n_7n_8 is pattern written by the test program.
n_9n_{10} is failed read pattern.

$n_2=1$: RAM address error. Address bus or DRAM chip has failed.

$n_2n_3n_4n_5n_6$ is absolute address where address error occured.
$n_7n_8n_9n_{10}$ is error address bit pattern.

KEYBOARD TEST

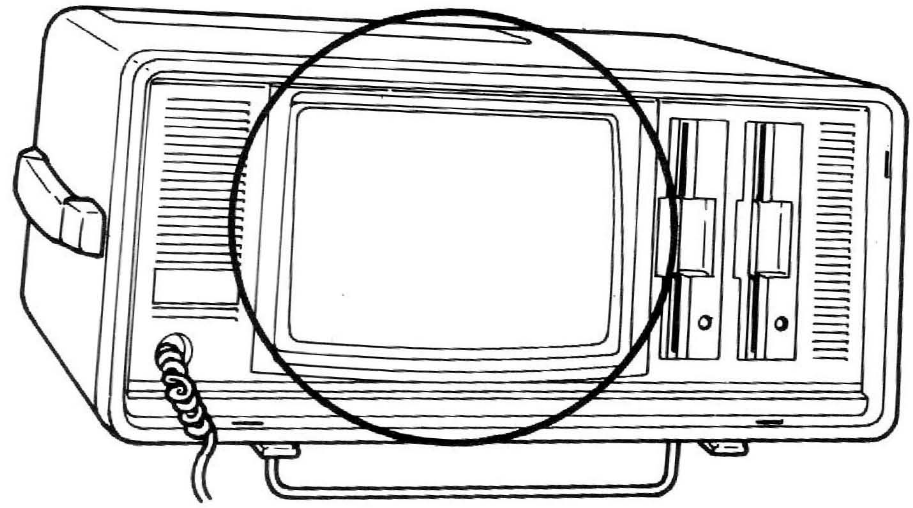


KEY n_1n_2 $n_3n_4n_5n_6$

Return code error. Key specified by return code is held down or keyboard cable is disconnected.

$n_1n_2n_3n_4$ is 0000.
n_5n_6 is failed return code.

GRAPHICS VIDEO TEST



CRTC ERROR

CRTC register has failed.

VSTS ERROR

Status has failed. Horizontal Sync. signal is not detected at address 3DAH.

VIDEO n_1n_2 $n_3n_4n_5n_6$

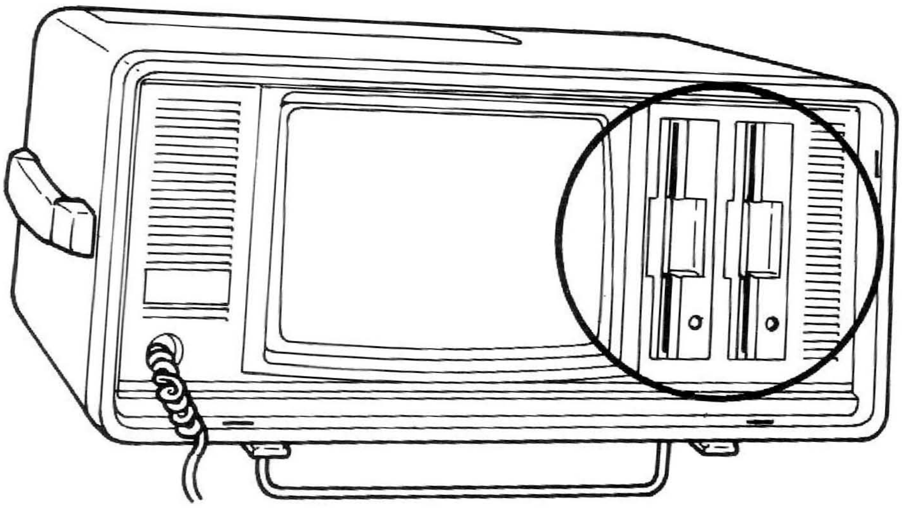
n_1n_2=00: VRAM data has failed. Data bus or VRAM chip has failed.

$n_3n_4n_5n_6$ is error address.

n_1n_2=01: VRAM address has failed. Address bus or VRAM chip has failed.

$n_3n_4n_5n_6$ is error address.

FLOPPY DISK DRIVE TEST



DISK n_1n_2 $n_3n_4n_5n_6$

n_1n_2=00: Floppy Disk Controller (FDC) master status has failed. Data bus or FDC chip has failed.

n_3n_4 is 00.
n_5n_6 is failed FDC master status.

n_1n_2=01: Floppy Disk Controller (FDC) initialization has failed. Data bus or FDC chip has failed.

n_3n_4 is 00.
n_5n_6 is BIOS disk error status listed in Table A-1.

n_1n_2=02: Seek has failed. Floppy Disk Controller (FDC) cable or FDC chip has failed.
n_3n_4 is 00.
n_5n_6 is failed FDC status register #0 listed below.

nibble	bit	meaning
n_5	D_3, D_2	Interrupt Code 00: Normal termination 01: Abnormal termination 10: Invalid command 11: Attention interrupt
	D_1	Seek end
	D_0	Equipment check
n_6	D_3	Not ready
	D_2	Head address
	D_1	Unit select 1
	D_0	Unit select 0

BIOS disk error status

Status	Meaning
80	Time out
40	Seek error
20	FDC failed
10	CRC error
09	DMA boundary
08	DMA overrun
04	No record
03	Write protected
02	No address mark
01	Invalid command

Table A-1

Error code...HEAD=n_1n_2 TRACK=n_3n_4 SECTOR=n_5n_6 STATUS=$n_7n_8...n_{28}$

Error code is:

TIME OUT
SEEK ERROR
FDC FAILED
CRC ERROR
DMA OVERRUN
NO RECORD
WRITE PROTECTED
NO ADDRESS MARK
INVALID COMMAND

n_1n_2 is side of disk. 0: side 0 1: side 1.
n_3n_4 is track number.
n_5n_6 is sector number.
n_7n_8 is function code.

02: Read error. FDC, Floppy disk drive (FDD) or data separater (9420) has failed.
03: Write error. FDC, FDD or data separater (9420) has failed.
04: Verify error. FDC, FDD or data separater (9420) has failed.
05: Format error. FDC or FDD has failed.

n_9n_{10} is BIOS disk error status listed Table A-1.
$n_{11}...n_{24}$ is FDC status. This value is only for service station.
$n_{25}n_{26}n_{27}n_{28}$ is total disk error occurancy.

SYS ERR 2

Rotation velocity cannot be measured.

SPEED FAILED–OUT OF 96%...104% OF ROTATION PERIOD

Rate of rotation period is out of valid range. (less than 96% or more than 104%)

SPEED STATUS=$n_1n_2n_3n_4$

$n_1n_2n_3n_4$: rotation period (ms)

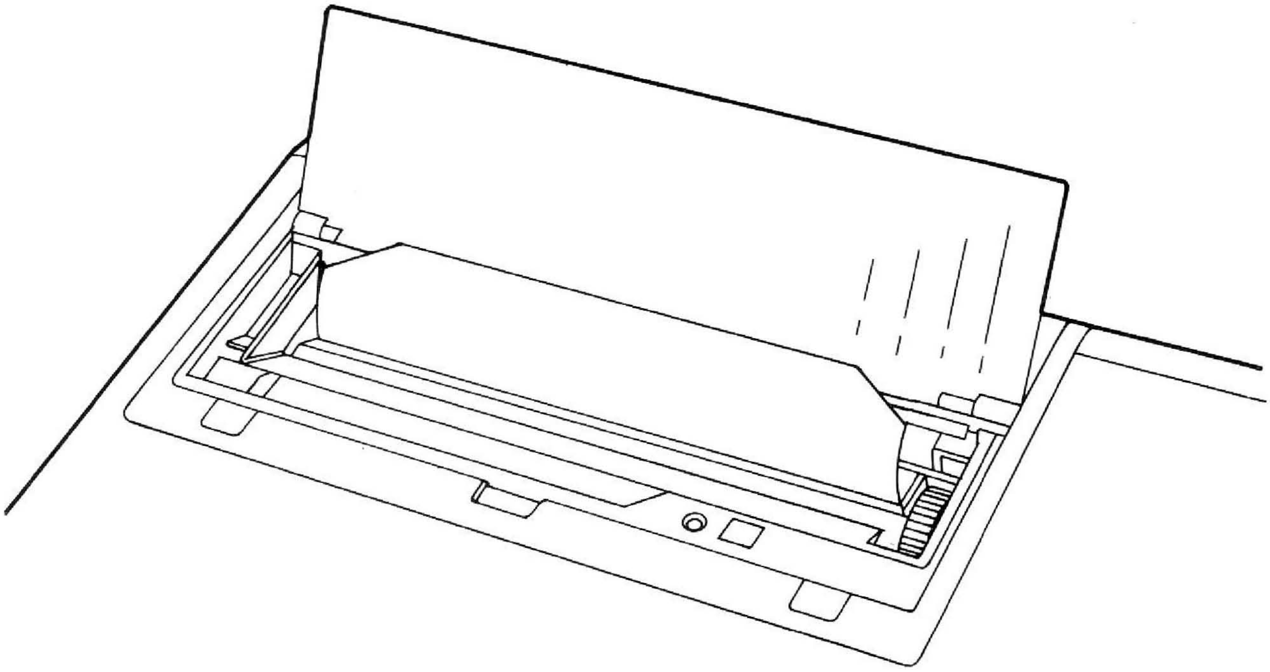
STATUS=$n_1n_2n_3n_4$ $n_5n_6n_7n_8 \ldots n_{33}n_{34}n_{35}n_{36}$

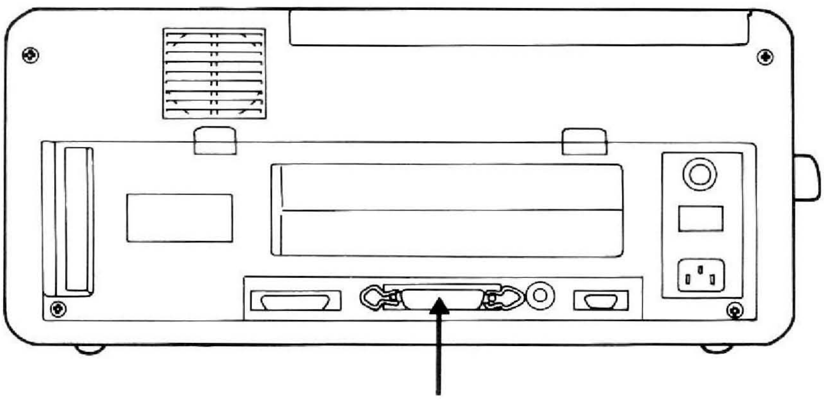
$n_1n_2n_3n_4$ is "TIME OUT" count
$n_5n_6n_7n_8$ is "SEEK ERROR" count
$n_9n_{10}n_{11}n_{12}$ is "FDC FAILED" count
$n_{13}n_{14}n_{15}n_{16}$ is "CRC ERROR" count
$n_{17}n_{18}n_{19}n_{20}$ is "DMA OVERRUN" count
$n_{21}n_{22}n_{23}n_{24}$ is "NO RECORD" count
$n_{25}n_{26}n_{27}n_{28}$ is "WRITE PROTECTED ERROR" count
$n_{29}n_{30}n_{31}n_{32}$ is "NO ADDRESS MARK" count
$n_{33}n_{34}n_{35}n_{36}$ is "INVALID COMMAND" count

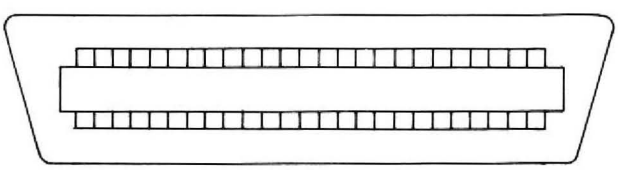
*** **WARNING** ***

This status sometimes depends on a condition of a disk or disk drive. If the errors occured, try to clean a head of the disk drive by a cleaning disk etc.

PARALLEL PORT TEST







PRT n_1n_2 $n_3n_4n_5n_6$

Parallel port data register has failed.

n_1n_2 is 00.
n_3n_4 is expected value.
n_5n_6 is failed value.

Error Code STATUS=$n_1n_2n_3n_4n_5n_6n_7n_8$

Error Code is:

TIME OUT
PAPER END
I/O ERROR

$n_1n_2n_3n_4$ is parallel port I/O address.

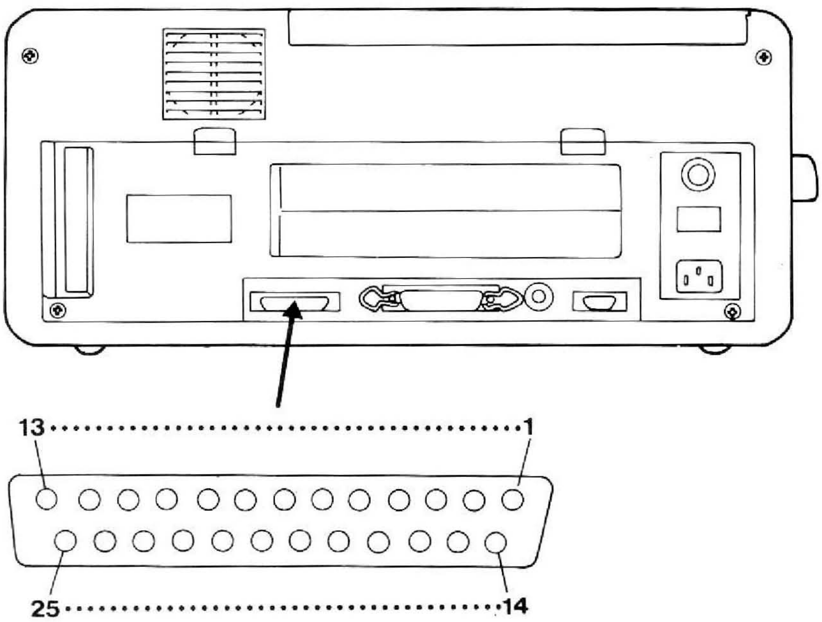
0FFFH:internal parallel port
0378H: external parallel port
0278H: optional parallel port
03BCH:optional parallel port in the MONOCHROME VIDEO ADAPTER

n_5n_6 is printer BIOS error status listed below.

nibble	bit	meaning
n_5	D_3	0: Busy
	D_2	1: Acknowledge
	D_1	1: Paper end
	D_0	1: Printer is selected
n_6	D_3	1: I/O error
	D_0	1: Time Out

n_7n_8 is the character sent to a printer

SERIAL PORT TEST



RS232 n_1n_2 $n_3n_4n_5n_6$

n_1n_2=00: Serial port controller (8250) could not be found.

$n_3n_4n_5n_6$ is 0000.

n_1n_2=01: Divisor register has failed

$n_3n_4n_5n_6$ is failed bit pattern.

n_1n_2=02: Loopback status has failed

n_3n_4 is 00.

n_5n_6: Refer to the following table.

nibble	bit	meaning
n_5	D_2	1: Transmit shift register is empty
	D_1	1: Transmit holding register is empty
	D_0	1: Break interrupt
n_6	D_3	1: Framming error
	D_2	1: Parity error
	D_1	1: Overrun error
	D_0	1: Data ready

n_1n_2=03: Loopback data has failed.

n_3n_4 is expected data.
n_5n_6 is failed data.

n_1n_2=04: Loopback status has failed.

n_3n_4 is expected status
n_5n_6 is failed status

See n_7n_8 in the serial port BIOS status table for the value n_3n_4 or n_5n_6 (Loopback status).

STATUS FAILED—STATUS=$n_1n_2n_3n_4$

The controller and/or signal drivers/receivers have failed.

n_1n_2 is expected status.
n_3n_4 is failed status.

nibble	bit	meaning
n_1 or n_3	D_3	Carrier detect
	D_2	Ring indicator
	D_1	Data set ready
	D_0	Clear to send

SEND: Error Code STATUS=$n_1n_2n_3n_4n_5n_6n_7n_8$
RECEIVE: Error Code STATUS=$n_1n_2n_3n_4n_5n_6n_7n_8$

The controller and/or signal drivers/receivers have failed.

$n_1n_2n_3n_4$ is I/O address of serial port error occured.

03F8H is internal serial port.
02F8H is optional serial port.

$n_4n_5n_6n_7$ is serial port BIOS status listed on the following page.

nibble	bit	meaning
n_5	D_3	1: Time out
	D_2	1: Transmit shift register is empty
	D_1	1: Transmit holding register is empy
	D_0	1: Break detect
n_6	D_3	1: Framming error
	D_2	1: Parity error
	D_1	1: Overrun error
	D_0	1: Data ready
n_7	D_3	1: Carrier detect
	D_2	1: Ring indicator
	D_1	1: Data set ready
	D_0	1: Clear to send
n_8	D_3	1: Delta carrier detect
	D_2	1: Trailing edge ring indicator
	D_1	1: Delta data set ready
	D_0	1: Delta clear to send.

COMPARE ERROR–STATUS=$n_1n_2n_3n_4n_5n_6n_7n_8$

$n_1n_2n_3n_4$ is I/O address of serial port error occured.
n_5n_6 is expected data.
n_7n_8 is failed data.

JOYSTICK TEST

JOYSTICK FAILED–STATUS=$n_1n_2n_3n_4$

Time count out. Timer chip has failed or potentionmeter is disconnected.

n_1n_2 is failed joystick register value listed below.

nibble	bit	meaning
n_2	D_3	Joystick-A Y-coordinate bit
	D_2	Joystick-A X-coordinate bit
	D_1	Joystick-B Y-coordinate bit
	D_0	Joystick-B X-coordinate bit

n_3n_4 is failed bit of n_1n_2

JOYSTICK VALUE FAILED

Time count of installed joystick equals zero. Timer chip has failed or potentionmeter is disconnected.

COUNT STATUS=n_1n_2 n_3n_4 n_5n_6 n_7n_8

n_1n_2 is time count of joystick-A X-coordinate
n_3n_4 is time count of joystick-A Y-coordinate
n_5n_6 is time count of joystick-B X-coordinate
n_7n_8 is time count of joystick-B Y-coordinate

MEMO

MEMO

APPENDIX B

ERROR MESSAGES

APP. B

INTRODUCTION

There are two parts to this chapter.

Device Error Messages

Messages written by DOS to indicate errors while reading from or writing to devices.

Other Messages

The other DOS messages are written in bold type with a description following the message and are presented in alphbetic order.

The program name or command which created the message appears as the first word of the description.

DEVICE ERROR MESSAGES

A message will be displayed on your system when an error is detected while reading from or writing to any of the system devices, such as disk drives, printer, etc. The following format is used:

type error writing *device* Abort, Retry, Ignore?
or
type error reading *device* Abort, Retry, Ignore?

device is the name of the device in error, such as CON, or drive A, drive B etc.

type is one of the following error types:

Bad call format

The length request header passed to a device driver is wrong. Contact your dealer for assistance.

Bad command

A command issued to a *device* by a device driver is incorrect.

Bad unit

The sub-unit number passed to a device driver is invalid. Contact your dealer for assistance.

Data

Data was written or read incorrectly. Your disk may have developed a defective spot.

Disk

A disk error has occurred which was not previously described.

No paper

The named printer is either not turned on or lacks paper.

Non-DOS disk

There is invalid information in the file allocation table.

Not ready

The indicated device is not ready. It cannot receive or send data.

Read Fault

The data was not successfully read.

Sector not found

The sector which is indicated as containing the data could not be located. The disk may have developed a defective spot.

Seek

The proper track on the disk could not be located.

Write fault

The data to the device was not successfully written from the device.

Write protect

An attempt was made to write on a write protected disk.

Warning:

DO NOT remove or insert disks if any of these messages appear concerning a disk drive.

What To Do When A Device Error Message Appears:

The system will wait for one of the following responses:

A for Abort. The system ends the program that requested the disk read or write.

R for Retry. The system will try again to do the disk read or write operation.

I for Ignore. The system processes the program as if the error had not happened.

The following order is suggested in responding:

R Retry the operation. The problem may not happen again.

A Abort the program.

I Ignore the problem and continue processing. Be aware that data is lost when you use the Ignore response.

OTHER MESSAGES

A

About to generate .EXE file
Change disks <hit ENTER>

LINK. When you specify the /PAUSE parameter, this is displayed. Put the Runfile disk into the appropriate drive and press <ENTER>.

All files canceled by operator

PRINT. This message appears when you cancel the printing of all queued files using the /T parameter. It is printed out.

All specified file(s) are contiguous

CHKDSK. All the files you requested to write are on the disk sequentially.

Allocation error, size adjusted

CHKDSK. An invalid sector number was in the file allocation table. The file whose name appears before this message, was truncated at the end of the last good sector.

Attempt to access data outside of segment bounds, possibly bad object module

LINK. An object file is most likely incorrect.

Attempted write-protect violation

FORMAT. An attempt was made to format a write-protected disk. Insert a new disk and press a key to begin formatting.

B

Bad command or file name

DOS. You have entered a DOS command incorrectly. Check your syntax and re-try the command. If the command is correct, be sure that the default drive contains either the external command or batch file you are trying to execute.

Command Interpreter
Bad or missing

DOS. The file COMMAND.COM was not contained in the disk being booted, or an error was found during loading the disk.

Bad numeric parameter

LINK. An invalid numberic constant was specified with the /STACK parameter.

BF Error

DEBUG. Bad Flag. An incorrect flag code setting was entered. See the Register command for the valid codes.

APP. B

BP Error

DEBUG. Too many breakpoints. No more than ten breakpoints may be specified for the Go command. Retype the Go command with ten or fewer breakpoints.

BR Error

DEBUG. Bad register. An incorrect register name was specified. See the Register command for the valid register names.

C

Cannot do binary reads from a device

COPY. /B switch was used with a device name during copying from the device.

Cannot edit .BAK file--rename file

EDLIN. .BAK files shouldn't be edited because these files are assumed to be backup files, with more up-to-date versions of the files.

To edit the .BAK file either rename the file, or copy it and give the copy another name.

Cannot find file xxxx.OBJ
Change disk <hit ENTER>

LINK. The specified object module was not found on the drive by linker. Insert the proper disk with the named module on it an press <ENTER>.

Cannot find library xxxx.LIB
Enter new drive letter:

LINK. The named library could not be located on the drive. Enter the proper letter for the drive the library is on.

Cannot open temporary file

LINK. The temporary file cannot be opened because the disk or the directory is full.

COMn: bbbb, p, d, s, t

MODE. The Serial (RS232C) Interface was initialized. The parameters are:

n COM1 or COM2

bbbb baud rate

p parity

n none

e even

o odd

s stop bits (1 or 2)

t type of serial device

p serial printer (serial timeouts are retried)

– other serial device (serial timeouts will not be retried)

Compare error(s) on Track xx, Side y

DISKCOMP. Contents on the track number xx on the side y were different.

Contains xxx non-contiguous blocks

CHKDSK. The file named has been written in sections on different areas of the disk rather than sequentially. However, since badly fragmented file take longer to read, copying the file sequentially could improve run time and enhance system performance.

D

DF Error

DEBUG. Double flag. There were conflicting codes denoted for a single flag. A flag may be altered only once per Register command.

Disk boot error

DOS. This disk was not for booting DOS. Insert the System Disk or other disk for booting and restart.

Disk error writing FAT X

CHKDSK. A disk error occurred while CHKDSK was trying to update the file allocation table (FAT) on the named drive. X (1 or 2) depends on which of the 2 copies of the file allocation table could not be written. If the message is for FAT 1 and 2, the disk is not usable.

Disk full—write not completed

EDLIN. The disk does not have enough free space to save the entire file; therefore an End Edit command terminated abnormally. Part of the file may be on the disk, but the file will be incomplete.

Disk unsuitable for system disk

FORMAT. The system program could not written onto the disk being formatted.

Duplicate filename or File not found

RENAME. While attempting to rename a file, the new filename either already existed on the disk, or the file to be renamed was not found on the specified (or default) drive.

E

Entry error

EDLIN. There is a syntax error on the last command.

Error: Dup record too complex

LINK. An object module created from an assembler program has DUP record which is too complex. To correct this problem you must debug the assembler program and then LINK it again.

`Error found, F parameter not specified.`
`Corrections will not be written to disk.`

CHKDSK. Even though the /F parameter was not used, CHKDSK will perform its analysis so that you can see the results, but the corrections will not actually be written on the disk.

`Error writing to device`

Commands. DOS could not write data to the device because the data was more than the device was expecting.

`EXE and HEX files cannot be written`

DEBUG. DEBUG cannot convert data which requires a backwards conversion.

F

`File cannot be copied onto itself`

DOS. When a request is made to COPY a file a different name should be given to the copy or it should be put in a different directory or on another disk.

`File creation error`

DOS and commands. You tried to add a new filename to the directory or to replace a file that was already there. If the file already existed, it was marked read-only and could not be replaced. If you run CHKDSK, you will see if the directory is full, or is something else caused to error.

File is cross-linked: on cluster XX

CHKDSK. This message renames the two files in error and appears twice for each cross-linked cluster number. Both files are allocated to the same data block. Since no corrective action is taken automatically, you must correct the problem by:

1. Copy both files using the COPY command.

2. Delete the original files using the ERASE command.

3. Review the files and edit.

File not found

DOS and commands. A file listed in either a command parameter or command could not be found in the directory on the specified (or default) drive.

First cluster number is invalid,
entry truncated.

CHKDSK. There is an invalid pointer to the data area on the file named before this message. If the /F parameter is specified, the file is truncated to a zero-length file.

Fixup offset exceeds filed width

LINK. A NEAR attribute instead of a FAR attribute was indicated in an assembler instruction address. You must edit the assembler source code and reprocess again.

Fix-ups needed—base segment (hex):

EXE2BIN. A load segment was indicated as required for the file in the source (.EXE) file. You must indicate the absolute segment address for loading the finished module.

*** **WARNING** ***

Because the program will be dependent upon being loaded at a specific memory location, we do not recommend using such a program as a .COM file.

FOR cannot be nested

BATCH. There are more than one FOR command on the command line.

Format failure

FORMAT. While creating the target disk, a disk error occurred. This disk is unusable. or a write-protected disk.

I

Illegal device name

MODE. The named printer must be LPT1:, LPT2:, or LPT3:; the named RS232C Interface must exist and be COM1: or COM2:. No more than one blank is allowed between MODE and its parameters.

Incompatible disk or drive types

DISKCOMP. The number of sectors or sides of disks being compared is different.

Incompatible system size

SYS. A system transfer did not occur because the disk being copied contains a larger copy of DOS then the target disk. One solution is to format a blank disk using the FORMAT /S command and then copy the files you want to the new disk.

Incorrect DOS version

Commands. In order to use the command you just entered, a different version of DOS is needed than the one you are running.

Input file read error

LINK. An object file is probably bad.

Insufficient disk space

DOS and commands. There is not enough free space on the disk to hold the file being written. If you think there is enough space, run CHKDSK to determine the status of the disk.

Insufficient memory

Commands. These commands need more memory than is available to function. Change the BUFFERS=parameter in the CONFIG.SYS file to a smaller value (if you have specified BUFFERS=), restart the system and retry the command. If the message appears again, your system does not contain enough memory to execute this command.

Insufficient room in root directory
Erase files from root and repeat CHKDSK.

CHKDSK. While creating files from the "lost" data blocks, CHKDSK found the root directory full; therefore all of the lost chains could not be recovered into files. Copy some of the recovered files to another disk for review, delete them from the disk you are checking, and rerun CHKDSK to recover the rest of the lost data.

Insufficient space on disk

DEBUG. There is not enough free space to hold the data being written on disk. You may insert a disk that does have enough free space and re-issue the Write command. Otherwise, erase the files from the disk and rerun DEBUG.

Invalid baud rate specified

MODE. The only valid baud rates are 110, 150, 300, 600, 1200, 2400, 4800, or 9600. You may even enter just the first two characters of the number.

Invalid characters in volume label

FORMAT. It is not a valid filename character.

Invalid date

DATE. Either you entered an invalid date or the delimiter was not a hyphen (-) or slash (/).

Invalid directory

DOS and commands. There isn't the directory in the path you specified.

Invalid drive in search pass

DOS. There was an invalid drive specification in the path you specified in the PASS command.

Invalid drive or file name

EDLIN. When starting EDLIN you did not specify a valid drive or filename.

Invalid drive specification

DOS and commands. The drive specified in a command or command parameter is invalid.

Invalid number of parameters

Commands. The number of parameters you specified does not agree with the number of parameters the command requires.

Invalid object module

LINK. The object module is either incomplete or it is incorrectly formed. This can occur when a language processor is stopped while it is processing.

Invalid parameters

DOS and Commands. A command request was made with an incorrect parameter. One of the things to check is that the drive specifier includes a colon. Check all parameters for invalid characters.

Invalid path, not directory, or directory not empty

RMDIR. The directory you specified did not exist, or the directory you wish to remove still has files or other subdirectories.

Invalid sub-directory entry

CHKDSK. There is invalid information in the sub-directory. CHKDSK will try to correct the error. Run CHKDSK with the /V parameter for more information about the error.

Invalid time

TIME. There is an invalid time or delimiter. A colon between the hours and minutes, and the minutes and seconds; and a period between the seconds and hundredths of a second are the only valid parameters.

L

Label not found

BATCH. The label to which GOTO command jumps does not exist.

Line too long

EDLIN. The Replace Text command has ended abnormally because the replacement causes the line to expand beyond the 253-character limit. Break the long line into shorter lines, and reissue Replace Text command.

List output is not assigned to a device

PRINT. The device named as the PRINT list device is invalid. Re-issue the PRINT command and give a valid list device name when prompted.

APP. B

M

Must specify destination line number

EDLIN. A destination line number must be entered with a Move or Copy command. Enter the command again but enter a valid destination line number.

N

Name of list device [PRN]:

PRINT. When you start print after DOS has been restarted, reply with the reserved device name which is to receive the printed output, or press <ENTER> to use the first parallel printer [PRN].

No Path

PATH. There is no path after DOS searches your working directory.

APP. B

No room for system on destination disk

SYS. The system cannot be transferred because the destination disk does not contain the required reserved space for DOS. One solution is to format a blank disk using the FORMAT /S command, and copy your other files to the new disk.

No room in directory for file

EDLIN. The file directory is full.

Non-System disk or disk error
Replace and strike any key when ready

Startup. The directory contains no entries for system files or there was a disk read error while starting up the system. Place a DOS disk in drive A: and restart your system.

Not enough room to merge the entire file

EDLIN. The entire contents of the specified file was not merged while issuing a Transfer command because there is insufficient memory. Only part of the file was merged.

Not found

EDLIN. While using a Replace text or Search text command the range of lines indicated does not contain the string being searched for or no further occurrences of the string were found after replying "N" to the "OK?" prompt.

Out of environment space

DOS. DOS could not execute the SET command since the area of the environment information could not be expanded.

P

PRINT queue is empty

PRINT. No files are being processed by PRINT.

PRINT queue is full

PRINT. There is a limit of 10 files that can be added to the print queue. After a file is printed, you can add another file to the print queue.

Printer error

MODE. Printer is not connected, or printer paper is empty etc.

Probable non-DOS disk.
Continue (Y/N)?

CHKDSK. There is invalid information in the file allocation table identification byte. The disk is either not formatted by DOS or has become damaged. CHKDSK will indicate its possible corrective actions without actually changing the disk if you have not used the /F parameter and a "Y" reply. It is recommended that you do this first, before you try using the /F switch and replying Y.

Program too big to fit in memory

DOS. The file for the command cannot be transfered in the memory since the file is too big to fit in it.

Program size exceeds capacity of LINK

LINK. Load module is too large for execution.

R

Read error in: <filename>

DOS and commands. Commands could not read the entire file.

Resident part of PRINT installed

PRINT. Resident part of the PRINT command were transfered into memory for printing. This message is displayed. When you use the PRINT command for the first time.

S

Segment size exceeds 64K

LINK. The addressing limit is 64K bytes. The LINK attempted to combine segments with the same name which resulted in a segment requirement greater than 64K bytes.

Stack size exceeds 65535 bytes

LINK. A stack size specified with /STACK must be less than 65535 bytes.

Symbol defined more than once

LINK. A single symbol name may not be defined in more than one module.

Symbol table capacity exceeded

LINK. The names exceeded 50K bytes. Use either fewer and/or shorter names.

Syntax error

DOS. An illegal string was typed.

T

Target disk may be unusable

DISKCOPY. The target disk may be unusuble because of unrecoverable read, write or verify error.

APP. B

Terminate batch job (Y/N)?

DOS. If you press Ctrl-Break while DOS is processing a batch file, this message appears. To stop processing the batch file press Y. If you press N, only the command that was executing when Ctrl-Break was pressed will stop and processing will continue with the next command in the batch file.

Too many external symbols in one module

LINK. There is a limit of 256 external symbols per module.

Too many groups

LINK. There is a limit of 10 groups, including DGROUP.

Too many libraries specified

LINK. There is a limit of eight libraries.

Too many public symbols

LINK. There is a limit of 1024 public symbols.

Too many segments or classes

LINK. There is a limit of 247 segments and classes, taken together.

Track 0 bad—disk unusable

FORMAT. The boot record, FAT and directory could not written on the disk being formatted.

Tree past this point not processed

CHKDSK. Because track 0 is bad, CHKDSK cannot continue processing the directory path being examined.

U

Unable to create directory

MKDIR. The divectory you specified already exists, the directory path name could not be found or there was no room on the disk to create the directory.

Unable to write BOOT

FORMAT. The boot record could not be written on the track 0 of the disk.

Unable to write FAT

FORMAT. The FAT (File Allocation Table) could not be written on the track 0 of the disk.

APP. B

Unexpected end-of-file on VM.TMP

LINK. The LINK could not be completed because the diskette with VM.TMP was removed.

Unrecognized command in CONFIG.SYS

Startup. An illegal command was found in the file CONFIG. SYS. Correct the illegal command by EDLIN.

Unrecoverable read error on source (target)-Track xx, side y

DOS. DOS could not read contents on the track XX on the side Y of the drive source (target) disk.

Unresolved Externals:

LINK. The modules or library files specified did not contain the definitions of the external symbols. When this error occurs, do not run the executable file created by the linker.

W

Warning: no STACK segment

LINK. A statement allocating stack space was not specified in any of the object modules.

WARNING—Read error on EXE file.

EXE2BIN. An error was detected during reading the input file. The result file may be unusable.

APPENDIX C

PIN CONFIGURATIONS

APP. C

RGB MONITOR PORT

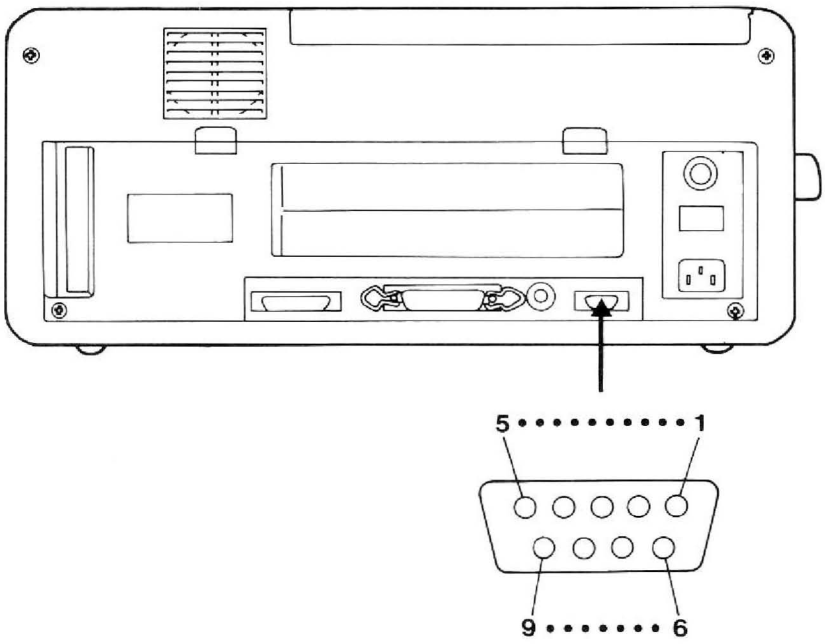


Fig. C-1. PIN ARRANGEMENT FOR RGB MONITOR

Pin Number	Signal Name/Description	Direction
1 & 2	Ground	———
3	Red	Out
4	Green	Out
5	Blue	Out
6	Intensity	Out
7	14 MHz Clock	Out
8	Horizontal Drive	Out
9	Vertical Drive	Out

*Use a **shielded** cable to connect the RGB monitor port and your RGB monitor display.

SERIAL PORT (RS232C)

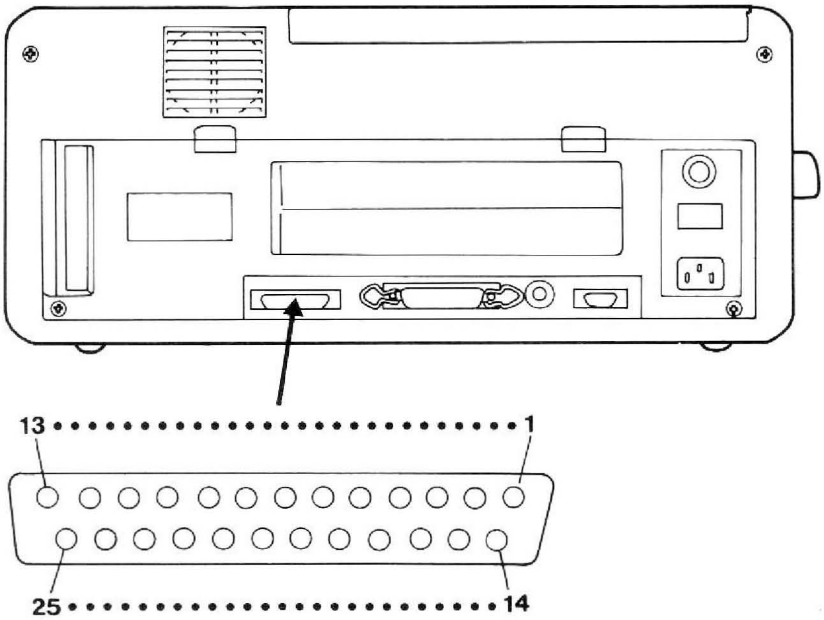


Fig. C-2. PIN ARRANGEMENT FOR SERIAL INTERFACE

Pin Number	Signal Name/Description	Direction
1	NC	———
2	Transmitted Data	Out
3	Received Data	In
4	Request to Send	Out
5	Clear to Send	In
6	Data Set Ready	In
7	Signal Ground	———
8	Carrier Detect	In
9	NC	———

APP. C

Pin Number	Signal/Description	Direction
10	NC	———
11	NC	———
12	NC	———
13	NC	———
14	NC	———
15	NC	———
16	NC	———
17	NC	———
18	NC	———
19	NC	———
20	Data Terminal Ready	Out
21	NC	———
22	Ring Indicator	In
23	NC	———
24	NC	———
25	NC	———

The serial port in Sr. Partner uses the address 3F8-3FF. If you wish to use other asynchronous adapter boards, refer to "I/O ADDRESS MAP" in this APPENDIX.

*Use a **shielded** cable to connect the serial port and your equipment with a serial interface.

PARALLEL PORT (Centronics)

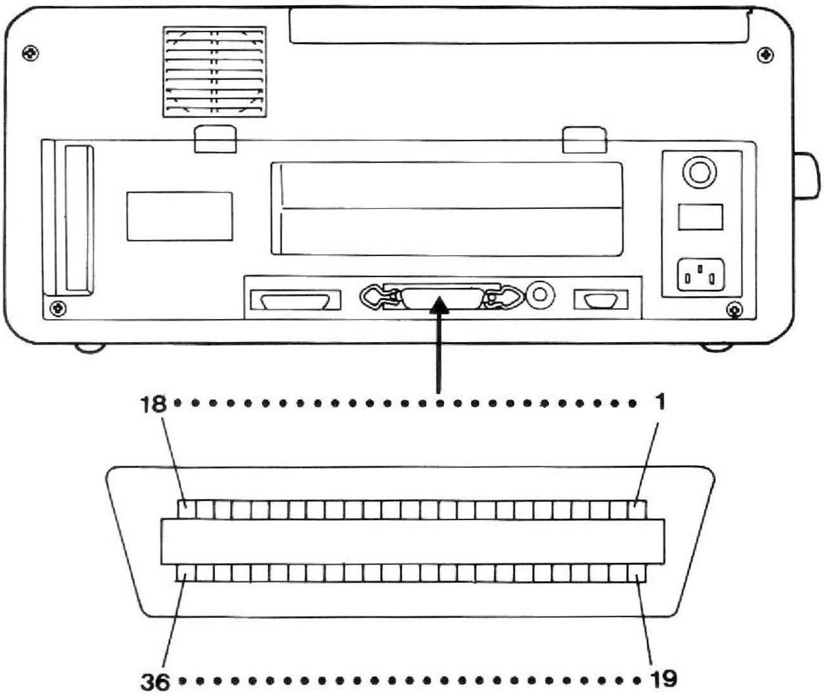


Fig. C-3. PIN ARRANGEMENT FOR PRINTER

Pin Number	Signal/Description	Direction
1	$\overline{\text{STROBE}}$ (STROBE Pulse to read data in)	Out
2	DATA 1	Out
3	DATA 2	Out
4	DATA 3	Out
5	DATA 4	Out
6	DATA 5	Out
7	DATA 6	Out
8	DATA 7	Out
9	DATA 8	Out
10	$\overline{\text{ACKNLG}}$ (Acknowledge data reception)	In

APP. C

Pin Number	Signal/Descrition	Direction
11	BUSY (Printer cannot receive data)	In
12	PE (out of paper)	In
13	SLCT (Printer in Selected state)	In
14	AUTO FEED XT (Paper fed automatically)	Out
15	NC	———
16	0 V (Logic Ground)	———
17	CHASSIS GND (Printer Chassis Ground)	———
18	NC	———
19 to 30	GND	———
31	INIT (Clear the print buffer)	Out
32	ERROR (Error State)	In
33	GND (Same as 19–30)	———
34	NC	———
35	NC	———
36	SLCT IN (Data Entry is Possible)	Out

The parallel port in Sr. Partner uses the address 378-37F. If you wish to use other parallel printer interface boards, Refer to "I/O ADDRESS MAP" in this APPENDIX.

*Use a **shielded** cable to connect the parallel port and your equipment with a parallel port.

I/O ADDRESS MAP

Address (HEX)	Function
278-27F	Reserved for Parallel Printer Interface
2F8-2FF	Reserved for Serial (RS232C) Interface
378-37F	Used by Parallel Interface.
3B0-3BF	Reserved for Monochrome Display/Parallel Printer Interface.
3D0-3DF	Used by Color/Graphics Interface.
3F0-3F7	Used by Disk Drive Interface.
3F8-3FF	Used by Serial (RS232C) Interface.

If you use an other serial interface board (Asyncronous Adapter), parallel interface board or other boards, you must select reserved address in above table. Refer to the user's manual of each interface board.

APPENDIX D

INTERRUPTS AND FUNCTION CALLS

INTERRUPTS

Interrupt types hex 20 to hex 3F are reserved for use by DOS. (Absolute memory locations hex 80 to hex FF are reserved.)

Interrupts follow with all values in hexadecimal.

20 Program terminate.

Call
    CS: Segment address of program Segment Prefix

Return
    None

This is the common way to exit from a program. Control is transferred to DOS. The directory entries for unclosed files are not correct (be sure files are closed).

21 Function Request

Call
    AH: Function number
    Other registers as specified in individual function

Return As specified in individual function

22 Terminate Address

When a program terminates, control transfers to the address at offset 0AH of the Program Segment Prefix. This address is copied into the Program Segment Prefix, from the Interrupt 22H vector, when the segment is created.

23 CONTROL-C Exit Address

If the user types <Ctrl> <C> during keyboard input or display output, control transfers to the INT 23H vector in the interrupt table. This address is copied into the Program Segment Prefix, from the Interrupt 23H vector, when the segment is created.

If the <Ctrl> <C> routine preserves all registers, it can end with an IRET instruction (return from interrupt) to continue program execution. When the interrupt occurs, all registers are set to the value they had when the original call to DOS was made. There are no restrictions on what a <Ctrl> <C> handler can do—including DOS function calls—so long as the registers are unchanged if IRET is used.

If Function 09H or 0AH (Display String or Buffered Keyboard Input) is interrupted by <Ctrl> <C>, the three-byte sequence 03H-0DH-0AH (EXT-CR-LF) is sent to the display and the function resumes at the beginning of the next line.

If the program creates a new segment and loads a second program that changes the <Ctrl> <C> address, termination of the second program restores the <Ctrl> <C> address to its value before execution of the second program.

24 Fatal Error Abort Address

If a fatal disk error occurs during execution of one of the disk I/O function calls, control transfers to the INT 24H vector in the vector table. This address is copied into the Program Segment Prefix, from the Interrupt 24H vector, when the segment is created.

BP:SI contains the address of a Device Header Control Block from which additional information can be retrieved.

Error Codes

When an error-handling program gains control from Interrupt 24H, the AX and DI registers can contain codes that describe the error. If Bit 7 of AH is 1, the error is either a bad image of the File Allocation Table or an error occurred on a character device. The device header passed in BP:SI can be examined to determine which case exists. If the attribute byte high order bit indicates a block device, then the error was a bad FAT. Otherwise, the error is on a character device.

The following are error codes for Interrupt 24H:

Error Code	Description
0	Attempt to write on write-protected disk
1	Unknown unit
2	Drive not ready
3	Unknown command
4	Data error
5	Bad request structure length
6	Seek error
7	Unknown media type
8	Sector not found
9	Printer out of paper
A	Write fault
B	Read fault
C	General failure

APP. D

The user stack will be in effect (the first item described below is at the top of the stack), and will contain the following from top to bottom:

IP	DOS registers from issuing INT 24H
CS	
FLAGS	
AX	User registers at time of original INT 21H request
BX	
CX	
DX	
SI	
DI	
BP	
DS	
ES	
IP	From the original INT 21H from the user to DOS
CS	
FLAGS	

The registers are set such that if an IRET is executed, DOS will respond according to (AL) as follows:

(AL)=0 ignore the error
=1 retry the operation
=2 terminate the program via INT 23H

APP. D

Notes:

1. Before giving this routine control for disk errors, DOS performs five retries.

2. For disk errors, this exit is taken only for errors occurring during an Interrupt 21H. It is not used for errors during Interrupts 25H or 26H.

3. This routine is entered in a disabled state.

4. The SS, SP, DS, ES, BX, CX, and DX registers must be preserved.

5. This interrupt handler should refrain from using DOS function calls. If necessary, it may use calls 01H through 0CH. Use of any other call will destroy the DOS stack and will leave DOS in an unpredictable state.

6. The interrupt handler must not change the contents of the device header.

7. If the interrupt handler will handle errors rather than returning to DOS, it should restore the application program's registers from the stack, remove all but the last three words on the stack, then issue an IRET. This will return to the program immediately after the INT 21H that experienced the error. Note that if this is done, DOS will be in an unstable state until a function call higher than 0CH is issued.

25 Absolute disk read.

Call

AL: Drive number (0: Drive A, 1: Drive B)
DS:BX: Disk transfer address
CX: Number of sectors
DX: Beginning relative sector

Return

AX: Error code if CF=1

FLAGSL:CF=0 if sucessful

=1 if not successful

This transfer control to the DOS BIOS. This number of sectors(CX) is read from the disk to the disk transfer address.

26 Absolute disk write. This vector is issued exactly as in 25 above but applies to a write operation.

APP. D

27 Terminate but stay resident.

Call

CS:DX: First byte following last byte of code

Return

None

This vector is used if the program is to remain resident when COMMAND gets control.

28-3F Reserved for DOS.

FUNCTION CALLS

Function calls are provided by DOS for:

Character device I/O.
File management.
Memory management.
Date and time functions.
Execution of other programs.

Error Return Table

When an error is found, some function calls set the carry flag, and put an error return code in AX. Refer to each function call. The error return codes are:

Code	**Condition**
1	Invalid function number
2	File not found
3	Path not found
4	Too many open files (no handles left)
5	Access denied
6	Invalid handle
7	Memory control blocks destroyed
8	Insufficient memory
9	Invalid memory block address
10	Invalid environment
11	Invalid format
12	Invalid access code
13	Invalid data
15	Invalid drive was specified
16	Attempted to remove the current directory
17	Not same device
18	No more files

Invoking DOS Functions

A function is requested of DOS by placing it in the AH register, setting other registers as appropriate, and then issuing a type 21H interrupt.

File handle

The calls which support files or devices use a 16-bit binary value, file handle, which is returned in AX. The file handle is used in referring to the file after it's been opened. The following values are set by DOS as the file handle and can be used by your program without opening the files:

0000 Standard input device. Input can be redirected.

0001 Standard output device. Output can be redirected.

0002 Standard error output device. Output cannot be redirected.

0003 Standard auxiliary device.

0004 Standard printer device.

Functions follow with all values in hexadecimal.

0 Program terminate. All closed files are written to disk. The directory entries for files left open will not be correct. CS must contain the segment address of program segment prefix.

1 Keyboard input. Waits for a character to be read from the keyboard (the character is echoed back to the standard output device and <Ctrl> <C> is checked). Returns the character in AL.

2 Display output. The character in DL is output to the standard output device.

3 Auxiliary input. Waits for a character from the standard auxiliary device [Internal RS232C interface (Serial Port)]. Returns the character in AL.

4	Auxiliary output. The character in DL is output to the standard auxiliary device [Internal RS232C interface (Serial Port)].
5	Printer output. The character in DL is output to the printer.
6	Direct console I/O. Sets flag to check if character is ready or not. When DL is 0 FFH (255)—If a character has been typed at the keyboard, it is returned in AL and the Zero flag is 0; if a character has not been typed, the Zero flag is 1. When DL is not 0 FFH—The character in DL is displayed:
7	Direct console input. The character read from the keyboard is returned in AL. This function does not echo the character or check for <Ctrl> <C>.
8	Console input without echo. Function 8 is identical to function 1, except the key is not echoed. (It checks for <Ctrl> <C>)
9	Print string. Prints a character string in memory that is terminated by "$" (hex 24). DX must contain the offset (from the segment address in DS) of a string that ends with "$".
A	Buffered keyboard input. Fills the type-ahead buffer until the key <ENTER> is pressed. DX must contain the offset (from the segment address in DS) of an input buffer of the following form:

Byte	Contents
1	Maximum number of characters in buffer, including the CR (you must set this value).

	2	Actual number of characters typed, not counting the CR (the function sets this value).
	3–n	Buffer; must be at least as long as the number in byte 1.

B Check standard input status. Checks if a character is available from the keyboard and sets AL (AL=hex FF if a character is available, AL=00 if no character is available).

APP. D

C Clear standard input buffer and invoke a standard input function. The type-ahead buffer is cleared and a function specified in AL is performed. The value in AL must be 1, 6, 7, 8 or A.

D Disk reset. Copies all files to disk. Files not properly closed will not be recorded correctly in the disk directory.

E Select disk. The default drive is determined by the drive specified in DL (0=A, 1=B, etc.). The number of drives is returned in AL.

F Open file. The indicated file is searched for in the directory. The file then is opened or made the current file. DX must contain the offset (from the segment address in DS) of an unopend File Control Block (FCB).

If a directory entry for the file is found, AL returns 0 and the FCB is filled as follows:

> If the drive code was 0 (default disk), it is changed to the actual disk used (1=A;, 2=B:, etc.). This lets you change the default disk without interfering with subsequent operations on this file.
>
> The Current Block field (offset 0CH) is set to zero.
>
> The Record Size (offset 0EH) is set to the system default of 128.
>
> The File Size (offset 10H), Data of Last Write (offset 14H), and Time of Last Write (offset 16H) are set from the directory entry.

Before performing a sequential disk operation on the file, you must set the Current Record field (offset 20H). Before performing a random disk operation on the file, you must set the Relative Record field (offset 21H). If the default record size (128 bytes) is not correct, set it to the correct length. If a directory entry for the file is not found, AL returns 0 FFH (255).

10 Close file. After an opened file has been modified this function must be called. This lists the modified file correctly in the directory. DX must contain the offset (to the segment address in DS) of an opened FCB. If a directory entry for the file is found, the location of the file is compared with the corresponding entries in the FCB. The directory entry is updated, if necessary, to match the FCB, and AL returns 0.

If a directory entry for the file is not found, AL returns 0 FFH (255).

11 Search for the first entry. This function is used to search for the first occurrance of the specified filename in the directory. DX must contain the offset (from the segment address in DS) of an unopened FCB. If a directory entry for the filename in the FCB is found, AL returns 0 and an unopened FCB of the same type (normal or extended) is created at the Disk Transfer Address.

If a directory entry for the filename in the FCB is not found, AL returns 0 FFH (255).

Notes:
If an extended FCB is used, the following search pattern is used:

1. If the FCB attribute is zero, only normal file entries are found. Entries for volume label, sub-directories, hidden, and system files will not be returned.

2. If the attribute field is set for hidden or system files, or directory entries, it is to be considered as an inclusive search. All normal file entries plus all entries matching the specified attributes are returned. To look at all directory entries except the volume label, the attribute byte may be set to hidden+system+directory (all 3 bits on).

12 Search for the next entry. This function searches for the next occurrance of the match found with function 11. DX must contain the offset (from the segment address in DS) of an FCB previously specified in function 11H. If a directory entry for the filename in the FCB is found, AL returns 0 and an unopened FCB of the same type (normal or extended) is created at the Disk Transfer Address.

If a directory entry for the filename in the FCB is not found, AL returns 0 FFH (255).

13 Delete file. Deletes the indicated file from the directory. DX must contain the offset (from the segment address in DS) of an unopened FCB. If a matching directory entry is found, it is deleted from the directory and AL returns 0.

If no matching directory entry is found, AL returns 0 FFH (255).

14 Sequential read. The indicated record is read. DX must contain the offset (from the segment address in DS) of an opened FCB. The record pointed to by the current block (offset 0CH) and Current Record (offset 20H) fields is loaded at the Disk Transfer Address, then the Current Block and Current Record fields are incremented.

This record size is set to the value at offset 0EH in the FCB.

AL returns a code that describes the processing:

Code	Meaning
0	Read completed successfully.
1	End-of-file, no data in the record.
2	Not enough room at the Disk Transfer Address to read one record; read canceled.
3	End-of-file; a partial record was read and padded to the record length with zeros.

15 Sequential write. The indicated record is written. DX must contain the offset (from the segment address in DS) of an opened FCB. The record pointed to by Current Block (offset 0CH) and Current Record (offset 20H) fields is written from the Disk Transfer Address, then the current block and current record fields are incremented.

The record size is set to the value at offset 0EH in the FCB. If the Record Size is less than a sector, the data at the Disk Transfer Address is written to a buffer; the buffer is written to disk when it contains a full sector of data, or the file is closed, or a Reset Disk system call (Function 0DH) is issued.

APP. D

AL returns a code that describes the processing:

Code	Meaning
0	Transfer completed successfully.
1	Disk full; write canceled.
2	Not enough room at the Disk Transfer Address to write one record; write canceled

16 Create file. DX must contain the offset (from the segment address in DS) of an unopened FCB. The directory is searched for an empty entry or an existing entry for the specified filename.

If an empty directory entry is found, it is initialized to a zero-length file, the Open File system call (Function 0FH) is called, and AL returns 0. You can create a hidden file by using an extended FCB with the attribute byte (offset FCB-1) set to 2.

If an entry is found for the specified filename, all data in the file is released, making a zero-length file, and the Open File system call (Function 0FH) is issued for the filename (in other words, if you try to create a file that already exists, the existing file is erased, and a new, empty file is created).

If an empty directory entry is not found and there is no entry for the specified filename, AL returns 0 FFH (255).

17	Rename file. DX must contain the offset (from the segment address in DS) of an FCB with the drive number and filename filled in, followed by a second filename at offset 11H. The disk directory is searched for an entry that matches the first filename, which can contain the ? wild card character. If a matching directory entry is found, the filename in the directory entry is changed to match the second filename in the modified FCB (the two filenames cannot be the same name). If the ? wild card character is used in the second filename, the corresponding characters in the filename of the directory entry are not changed. AL returns 0. If a matching directory entry is not found or an entry is found for the second filename, AL returns 0 FFH (255).
18	Used by DOS.
19	Current disk. The current default drive is returned in AL (0=A, 1=B, etc.).
1A	Set disk transfer address. DX must contain the offset (from the segment address in DS) of the Disk Transfer Address.
1B–20	Used by DOS.

21 Random read. Reads a random record specified by filed and sector. DX must contain the offset (from the segment address in DS) of an opened FCB. The Current Block (offset 0CH) and Current Record (offset 20H) fields are set to agree with the Relative Record field (offset 21H), then the record addressed by these fields is loaded at the Disk Transfer Address.

AL returns a code that describes the processing:

Code	Meaning
0	Read completed sucessfully.
1	End-of-file; no data in the record.
2	Not enough room at the Disk Transfer Address to read one record; read canceled.
3	End-of-file; a partial record was read and padded to the record length with zeros.

22 Random write. Writes a record to a random address. DX must contain the offset from the segment address in DS of an opened FCB. The Current Block (offset 0CH) and Current Record (offset 20H) fields are set to agree with the Relative Record field (offset 21H), then the record addressed by these fields is written from the Disk Transfer Address. If the record size is smaller than a sector (512 bytes), the records are buffered until a sector is ready to write.

AL returns a code that describes the processing:

Code	Meaning
0	Write completed sucessfully.
1	Disk is full.
2	Not enough room at the Disk Transfer Address to write one record; write canceled.

APP. D

23 File size. This function call returns the number of records in the specified file. DX must contain the offset (from the segment address in DS) of an unopened FCB. You must set the Record Size field (offset 0EH) to the proper value before calling this function. The disk directory is searched for the first matching entry.

If a matching directory entry is found, the Relative Record field (offset 21H) is set to the number of records in the file, calculated from the total file size in the directory entry (offset 1CH) and the Record Size field of the FCB (offset 0EH). AL returns 00.

If no matching directory is found, AL returns 0 FFH (255).

24 Set random record field. This function call sets the random record field to the current block and record fields. DX must contain the offset (from the segment address in DS) of an opened FCB.

25 Set interrupt vector. This function call uses the interrupt type specified in AL and sets the vector table to the address contained in DS:DX.

26 Create a new program segment. This function sets up a new program segment. It is best to use function call 4BH in place of this call.

27 Random block read. This function call reads the specified number of records. DX must contain the offset (to the segment address in DS) of an opened FCB. CX must contain the number of records to read; if it contains 0, the function returns without reading any records (no operation). The specified number of records—calculated from the Record Size field (offset 0EH)—is read starting at the record specified by the Relative Record field (offset 21H). The records are placed at the Disk Transfer Address.

AL returns a code that describes the processing:

Code	Meaning
0	Read completed sucessfully.
1	End-of-file; no data in the record.
2	Not enough room at the Disk Transfer Address to read one record; read canceled.
3	End-of-file, a partial record was read and padded to the record length with zeros.

28 Random block write. This function call writes the specified number of records. DX must contain the offset (to the segment address in DS) of an opened FCB; CX must contain either the number of records to write or 0. The specified number of records (calculated from the Record Size field, offset 0EH) is written from the Disk Transfer Address. The records are written to the file starting at the record specified in the Relative Record field (offset 21H) of the FCB. If CX is 0, no records are written, but the File Size field of the directory entry (offset 1CH) is set to the number of records specified by the Relative Record field of the FCB (offset 21H); allocation units are allocated or released, as required.

AL returns a code that describes the processing:

Code	Meaning
0	Write completed successfully.
1	Disk full. No records written.
2	Not enough room at the Disk Transfer Address to read one record; read canceled.

CX returns the number of records written; the Current Block (offset 0CH), Current Record (offset 20H), and Relative Record (offset 21H) fields are set to address the next record.

29 Parse filename. This function call checks the command line for a filename of the form d:filename.ext.

SI must contain the offset (to the segment address in DS) of a string (command line) to parse; DI must contain the offset (to the segment address in ES) of an unopened FCB. The string is parsed for a filename of the form d:filename. ext; if one is found, a corresponding unopened FCB is created at ES:DI.

Bits 0–3 of AL control the parsing and processing. Bits 4–7 are ignored:

Bit	Value	Meaning
0	0	All parsing stops if a file separator is encountered.
	1	Leading separators are ignored.
1	0	The drive number in the FCB is set to 0 (default drive) if the string does not contain a drive number.
	1	The drive number in the FCB is not changed if the string does not contain a drive number.
2	1	The filename in the FCB is not changed if the string does not contain a filename.
	0	The filename in the FCB is set to 8 blanks if the string does not contain a filename.
3	1	The extension in the FCB is not changed if the string does not contain an extension.
	0	The extension in the FCB is set to 3 blanks if the string does not contain an extension.

If the filename or extension includes an asterisk (*), all remaining characters in the name or extension are set to question mark (?).

Filename separators:

: . ; , = + / " [ ]\ < > ¦ space tab

Filename terminators include all the filename separators plus any control character. A filename cannot contain a filename terminator; if one is encountered, parsing stops.

If the string contains a valid filename:

1. AL returns 1 if the filename or extension contains a wild card character (* or ?); AL returns 0 if neither the filename nor extension contains a wild card character.

2. DS:SI point to the first character following the string that was parsed.

   ES:DI point to the first byte of the unopened FCB.

If the drive letter is invalid, AL returns 0FFH (255). If the string does not contain a valid filename, ES:DI+1 points to a blank (ASCII 32).

2A Get date. This function call returns the date in the following registers. The year (1980–2099 in binary) is in CH. The month (1=Jan, 2=Feb, etc.) is in DH. The day is in DL. The day of week (0=Sunday, 1=Monday, etc) is in AL.

2B Set date. This function call sets the date if CX has a year (1980–2099), DH has a month (1=Jan, 2=Feb,...) and DL has a day. AL returns 00 if the date was valid and the set operation was successful.

2C Get time. This function call returns the time-of-day in the following registers. The hours (0–23) are in CH. The minutes (0–59) are in CL. The seconds (0–59) are in DH. The 1/100 seconds (0–99) are in DL.

2D Set time. This function call sets the time if CH has hour, CL has minutes, DH has seconds and DL has 1/100 seconds. AL returns 00 if the time was valid.

2E Set/reset verify switch. This function call sets verify ON (AL=1) or OFF (AL=0). This is useful if you wish to verify the recording of critical data.

2F Get Disk Transfer Address (DTA). The current DTA transfer address is specified in ES:BX.

30 Set DOS version number. AL will contain the major version number and AH willcontain the minor version number.

31 Terminates process. This function call terminates the current process. AL must contain the exit code and DX must contain the memory size in paragraph.

32 Used by DOS.

33 Ctrl-Break check. This function call either sets [AL=1, DL=1 (Set), DL=0 (Reset)] or checks [AL=0, Returns 0 (Off) or 1 (On) in DL] the state of control-break checking.

34	Used by DOS.
35	Get vector. The interrupt vector is returned in ES:BX.
36	Get disk free space. This function call returns information on the free space on the disk. DL must contain the drive designator (0=default, 1=A, etc.). This call returns available clusters in BX, clusters per drive in DX, bytes per sector in CX. If FFFFH is in AX drive number is invalid, otherwise sectors per cluster.
37	Used by DOS.
38	Return country dependent information. This function call is used to set up information to be used internationally. Error return code is 2.
39	Create a sub-directory. This function call sets up a sub-directory. If any part of the directory path does not exist the path is not changed. DX:DS must contain the pointer to pathname. If carry is set and 3 is returned in AX, path is not found, or 5 in AX, access is not denied. If carry is not set, no error.
3A	Remove a sub-directory entry. This function call removes a sub-directory. Using this function call is same as "Create a sub-directory". Error return codes are 3, 5 and 16.
3B	Change the current directory. This function call changes the current directory. If any part of the directory path does not exist the path is not changed. Using this function call is same as "Create a sub-directory". Error return code is 3.

3C Create a file. This function call creates a new file or sets an old file to a length of zero so that it can be written. DS:DX must contain the pointer to pathname and CX must contain the file attribute. Error return codes are 3, 4 and 5.

3D Open a file. This function call opens a file and sets the access code in AL. The following are valid access codes:

0=open for reading.

1=open for writing.

2=open for both reading and writing.

DS:DX must contain the pointer of a filename. Error return codes are 2, 4, 5 and 14.

3E Close a file. This function call closes the file. All internal buffers are cleared. BX must contain the file handle. Error return code is 6.

3F Read from a file or device. This function call reads from a file or device however, the number of bytes read will vary from device to device. For instance, reading from the keyboard will read one line of text. DS:DX must contain pointer to buffer, CX: bytes to read and BX:file handle. Error return codes are 5 and 6.

40 Write to a file or device. This function writes to a file or device and returns the number of bytes written in CX. Using this function is same as "Read from a file or device" except CX. Error return codes are 5 and 6.

41 Delete a directory entry. This function call removes a directory entry associcated with a file name. DS:DX must contain the pointer to path name. Error return codes are 2 and 5.

42 Move file pointer. This function call moves the read/write pointer according to one of the following method:

AL=0	The pointer is moved to offset bytes from the beginning of the file.
AL=1	The pointer is moved to the current location plus offset.
AL=2	The pointer is moved to the end of file plus offset.

CX:DX must contain the distance to move (bytes) and BX must contain the file handle. Error return codes are 1 and 6.

43 Change file mode. This function call will either set the file to the attribute in CX (if AL=01) or return the current attribute in CX (if AL=00). Error return codes are 1, 3 and 5.

44 I/O control for devices. This function call uses function values in AL to set or get device information, or send/recieve control strings. The following values are allowed for function.

0	Get device information (returned in DX)
1	Set device information (as determined by DX)
2	Read CX number of bytes into DS:DX from device control channel
3	Write CX number of bytes from DS:DX to device control channel
4	Same as 2 only drive number in BL 0=default, A:=1, B:=2,...
5	Same as 3 only drive number in BL 0=default, A:=1, B:=2,...
6	Get input status
7	Get output status

The following registers have to be set calling this function.

BX: File handle
BL: Drive
DS:DX: Data or buffer
CX: Bytes to read or write

Calls AL=0 and AL=1

The bits of DX are defined as follows for calls AL=0 and AL=1. Note that the upper byte MUST be zero on a set call.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	CTRL	Reserved						ISDEV	EOF	RAW	SPECL	ISCLK	ISNVL	ISCOT	ISCIN

ISDEV=1 if this channel is a device
=0 if this channel is a disk file
(Bits 8–15 =0 in this case)

If ISDEV=1

EOF =0 if End of File on input
RAW =1 if this device is in Raw mode
=0 if this device is cooked
ISCLK =1 if this device is the clock device
ISNUL =1 if this device is the null device
ISCOT =1 if this device is the console output
ISCIN =1 if this device is the console input
SPECL =1 if this device is special
CTRL =0 if this device can not do control strings via calls AL=2 and AL=3.
CTRL =1 if this device can process control strings via calls AL=2 and AL×3.
Note that this bit cannot be set.

If ISDEV=0
EOF=0 if channel has been written
Bits 0–5 are the block device number
for the channel (0=A:, 1=B:,...)

Bits 15, 8–13, 4 are reserved and should not be altered.

Calls 2..5:

These four calls allow arbitrary control strings to be sent or received from a device. The call syntax is the same as the read and write calls, except for 4 and 5, which take a drive number in BL instead of a handle in BX.

An invalid function error is returned if the CTRL bit (see above) is 0.

An access denied is returned by calls AL=4,5 if the drive number is invalid.

Calls 6, 7:

These two calls allow the user to check if a file handle is ready for input or output. Status of handles open to a device is the intended use of these calls, but status of a handle open to a disk file is allowed, and is defined as follows:

Input:

Always ready (AL=FF) until EOF reached, then always not ready (AL=0) unless current position changed via LSEEK.

Output

Always ready (even if disk full).

Error return codes are 1, 5, 6 and 13.

45 Duplicate a file identifier. This function call takes the identifier for an opened file and returns a new file identifier that refers to the same file at the same position. BX must contain the file handle.

APP. D

46 Force a duplicate of a identifier. This function call sets the file handle in CX to the same stream as the handle in BX. Error return codes are 4 and 6.

47 Get Current directory. This function call places the full path name (starting from the root directory) of the current directory in the area of memory pointed to by DS:SI. DL must contain the drive number. Error return code is 15.

48 Allocate memory. BX must contain the size of memory to be allocated. If carry is not set, memory is allocated and the pointer to the allocated memory is in AX. Error return codes are 7 and 8.

49 Free allocated memory. A block of memory that was allocated by function call hex 48 is returned to the system pool. ES must contain the segment address of memory area to be freed. Error return codes are 7 and 9.

4A Modify allocated memory blocks. This function call will either enlarge or shrink the block depending on the block size request. ES must contain the segment address of memory area and BX must contain the requested memory area size. Error return codes are 7, 8 and 9.

4B Load or execute a program. This function call loads a program into memory and allows for program execution. DS:DX must contain the pointer to pathname and ES:BX must contain the pointer to parameter block.

A function must be in AL:

AL	Function
0	Load and execute the program. A program header is established for the program and the terminate and CONTROL-C addresses are set to the instruction after the EXEC system call.

3 Load (do not create) the program header, and do not begin execution. This is useful in loading program overlays.

For each value of AL, the block has the following format:

AL=0 → load/execute program

WORD segment address of environment
DWORD pointer to command line at 80H
DWORD pointer to default FCB to be passed at 5CH
DWORD pointer to default FCB to be passed at 6CH

AL=3 → load overlay

WORD segment address where file will be loaded.
WORD relocation factor to be applied to the image.

Note that all open files of a process are duplicated in the child process after an EXEC. This is extremely powerful; the parent process has control over the meanings of stdin, stdout, stderr, stdaux and stdprn. The parent could, for example, write a series of records to a file, open the file as standard input, open a listing file as standard output and then EXEC a sort program that takes its input from stdin and writes to standard output.

Also inherited (or passed from the parent) is an "environment." This is a block of text strings (less than 32K bytes total) that convey various configuration parameters. The format of the environment is as follows:

(paragraph boundary)

BYTE ASCIZ* string 1
BYTE ASCIZ string 2
...
BYTE ASCIZ string n
BYTE of zero

*ASCIZ string consists of ASCII string and code 00 at the end of the string.

Typically the environment strings have the form:

parameter=value

For example, COMMAND.COM might pass its execution search path as:

PATH=A:\BIN;B:\BASIC\LIB

A zero value of the environment address causes the child process to inherit the parent's environment unchanged. Error return codes are 1, 2, 8, 10 and 11.

4C Terminate a process. This function call will end the current process and transfer control to the calling process. A return code (must be in AL) can be sent and interrogated by the batch subcommands IF and ERRORLEVEL and by the wait function call (4DH). All open files are closed.

4D	Retrieve the return code of a sub-process. An Exit code which was specified by a function call hex 4C or function call hex 31 is returned in AX.
4E	Find first matching file. This function call finds the first occurrance of the filename in the directory. Filespec must be in DS:DX and search attribute in CX. (Refer to function call 11)
4F	Find next matching file. This function call will find the next directory entry matching the name that was specified on the previous Find First call. Error code 18 is returned in AX if there is no match on the filename.
50–53	Used by DOS.
54	Get verify state. This function call sets AL to 00 if verify is OFF. AL is set to 01 if verify is ON.
55	Used by DOS.
56	Rename a file. This function call renames a file allowing the file to be moved to another directory while it is being renamed. DS:DX must contain the pointer to filespec of existing file and ES:DI must contain the pointer to new filespec. Error return codes are 2, 5 and 17.
57	Get/Set a file's date and time. This function sets the date and time if AL=01 (CX: time to be set, DX=date to be set) and gets the date and time in CX and DX if AL=00. Bx must contain file handle. Error return codes are 1 and 6.

APPENDIX E

CONTROL BLOCKS

APP. E

STANDARD FILE CONTROL BLOCK

The Standard File Control Block (FCB) is defined as follows:
(The offsets are in decimal)

Byte	Function
0	Drive number. 0: default drive, 1: drive A, 2: drive B
1-8	Filename. This must be left-justified with trailing blanks.
9-11	Filename extension. This must be left-justified with trailing blanks.
12-13	Current block number. Each block contains 128 records. Zero indicates the first block of the file; one, the second and so on. Sequential READ or WRITE function requires the current block number as well as the current record field contained in byte 32.
14-15	Logical record size (bytes). OPEN and CREATE function calls set the logical record size to 80H.
16-19	File size (bytes). It may be read by a program but not changed. This is a 2-word field, the first word is the low-order part of the size.

APP. E

20-21	Date of file creation or update. It may be used by a program but not changed. The format of the 16-bit field is :

0-4	dd
5-8	mm
9-15	yy (the range for year is 0 to 119. This is year minus 1980.)

22-31	Reserved for use by DOS.
32	Current relative record number. It is within the current block for the current file. This indicates one of 128 records accessed by a sequential READ or WRITE. This field must be set before sequential READ or WRITE.
33-36	Random record field. This field must be set before doing random read/write operations.

Only the first three bytes are used if the record size is greater than or equal to 64 bytes. Four are used otherwise.

EXTENDED FILE CONTROL BLOCK

The Extended File Control Block is a 7 byte prefix used to search for files in the disk directory with special attributes.

The 7 byte prefix is formatted as follows:

Byte	Function
FCB-7	Flag byte. FFH indicates this is an extended FCB.
FCB-6 to FCB-2	Reserved.
FCB-1	Attribute byte. If bit #1 is set to 1 then searches will include hidden files. If bit #2 is set to 1 then searches will include system files.

APP. E

APPENDIX F

CHARACTER SET

HEXA-DECIMAL	0	1	2	3	4	5	6	7
0	0 BLANK (NULL)	16 ►	32 BLANK (SPACE)	48 0	64 @	80 P	96 ‘	112 p
1	1 ☺	17 ◄	33 !	49 1	65 A	81 Q	97 a	113 q
2	2 ☻	18 ↕	34 "	50 2	66 B	82 R	98 b	114 r
3	3 ♥	19 ‼	35 #	51 3	67 C	83 S	99 c	115 s
4	4 ♦	20 ¶	36 $	52 4	68 D	84 T	100 d	116 t
5	5 ♣	21 §	37 %	53 5	69 E	85 U	101 e	117 u
6	6 ♠	22 ▬	38 &	54 6	70 F	86 V	102 f	118 v
7	7 •	23 ↨	39 ´	55 7	71 G	87 W	103 g	119 w
8	8 ◘	24 ↑	40 (	56 8	72 H	88 X	104 h	120 x
9	9 ○	25 ↓	41)	57 9	73 I	89 Y	105 i	121 y
A	10 ◙	25 →	42 *	58 :	74 J	90 Z	106 j	122 z
B	11 ♂	27 ←	43 +	59 ;	75 K	91 [	107 k	123 {
C	12 ♀	28 ∟	44 ,	60 <	76 L	92 \	108 l	124 ¦
D	13 ♪	29 ↔	45 —	61 =	77 M	93]	109 m	125 }
E	14 ♫	30 ▲	46 •	62 >	78 N	94 ^	110 n	126 ~
F	15 ☼	31 ▼	47 /	63 ?	79 O	95 —	111 o	127 △

HEXA-DECIMAL VALUE	8	9	A	B	C	D	E	F
0	128 Ç	144 É	160 á	176 ░	192 └	208 ╨	224 α	240 ≡
1	129 ü	145 æ	161 í	177 ▒	193 ┴	209 ╤	225 β	241 ±
2	130 é	146 Æ	162 ó	178 ▓	194 ┬	210 ╥	226 γ	242 ≥
3	131 â	147 ô	163 ú	179 │	195 ├	211 ╙	227 π	243 ≤
4	132 ä	148 ö	164 ñ	180 ┤	196 ─	212 ╘	228 Σ	244 ⌠
5	133 à	149 ò	165 Ñ	181 ╡	197 ┼	213 ╒	229 σ	245 ⌡
6	134 å	150 û	166 a	182 ╢	198 ╞	214 ╓	230 μ	246 ÷
7	135 ç	151 ù	167 o	183 ╖	199 ╟	215 ╫	231 τ	247 ≈
8	136 ê	152 ÿ	168 ¿	184 ╕	200 ╚	216 ╪	232 Φ	248 °
9	137 ë	153 Ö	169 ⌐	185 ╣	201 ╔	217 ┘	233 θ	249 •
A	138 è	154 Ü	170 ¬	186 ║	202 ╩	218 ┌	234 Ω	250 ·
B	139 ï	155 ¢	171 ½	187 ╗	203 ╦	219 █	235 δ	251 √
C	140 î	156 £	172 ¼	188 ╝	204 ╠	220 ▄	236 ∞	252 η
D	141 ì	157 ¥	173 ¡	189 ╜	205 ═	221 ▌	237 φ	253 ²
E	142 Ä	158 Pt	174 »	190 ╛	206 ╬	222 ▐	238 ∈	254 ■
F	143 Å	159 f	175 «	191 ┐	207 ╧	223 ▀	239 ∩	255 BLANK (FF)

APP. F

APPENDIX G

SPECIFICATIONS

MPU	
CPU	8088 (16 bit) 4.77 MHz
Socket for co-processor	8087 (option) 4.77 MHz
Memory	
RAM (standard)	256K bytes
Expandable up to	512K bytes
ROM	16K bytes

Storage

5¼" Floppy Disk Drive (F.D.D.)×1
(360K bytes, Double Sided, Double Density)
Optional F.D.D. or 5¼" F.D.D.×2

Display	
CRT	9" High-resolution
Display Character (Switchable)	80 columns×25 lines 40 columns×25 lines
Graphics	640×200 dot (monochrome) 320×200 dot (color)
Keyboard	83 Keys (including 10 Key Numeric Pad 10 Function Keys)
Printer	80/132 columns, selectable

APP. G

Printer paper	8½" Thermal Type
Expansion	1 Port for Optional Board
I/O	Parallel Port (Centronics) Serial Port (RS232C)
Monitor out	RGB
Software O.S.	MS-DOS 2.11
Power Supply	AC 120 V 60 Hz
Current Consumption	1.6A (Max)
Operating Temperature	50°F–95°F (10°C–35°C)
Weight	31 lbs. (14 kg.) (1 F.D.D. is installed) 33 lbs. (15 kg.) (2 F.D.D. are installed)
Size	18-1/2 (W)×13-3/16 (D)×8-1/4 (H) inch 470 (W)×335 (D)×210 (H) mm
Control	Power Switch ON-OFF Brightness Control Line Feed Button

APPENDIX H

INDEX

A

B

C

APP. H

D

E

F

APP. H

APP. H

APP. H

APP. H

V

W

APP. H

MEMO

USA

Panasonic Industrial Company
Division of Matsushita Electric Corporation of America
One Panasonic Way,
Secaucus, New Jersey 07094

Panasonic Hawaii Inc.
91-238 Kauhi St. Ewa Beach
P.O. Box 774
Honolulu, Hawaii 96808-0774

Panasonic Sales Company
Division of Matsushita Electric of Puerto Rico, Inc.
Ave. 65 De Infanteria, KM 9.7
Victoria Industrial Park
Carolina, Puerto Rico 00630

CANADA

Matsushita Electric of Canada Limited
5770 Ambler Drive, Mississauga,
Ontario L4W 2T3

OTHERS

Matsushita Electric Trading Co., Ltd.
32nd floor, World Trade Center Bldg.,
No. 4-1, Hamamatsu-Cho 2-Chome,
Minato-Ku, Tokyo 105, Japan
Tokyo Branch P.O. Box 18 Trade Center

Printed in Japan RQX4308ZD F0184T0