p—System®

Supplement for the

MS®—DOS—Hosted p—System

IBM® Personal Computer

Published under licence in United Kingdom
by
TDI Limited
29 Alma Vale Road
Bristol BS8 2HL
Telephone: (0272) 742796

1—310.42
Preface

This supplement accompanies the standard p-System documentation. It contains MS-DOS-Hosted p-System information for specific use with the IBM Personal Computer and its relatives.

If you are familiar with the p-System, the information presented here will supplement your knowledge of it. However, if you are a beginner or have never used the p-System, you can execute the p-System by referring to Chapter 2 in this supplement. Then, it’s easy to become familiar with p-System operation by performing the hands-on tutorial procedures presented in the following document:

Personal Computing with the UCSD p-System

For further information about the system and its use, refer to the following publications:

Operating System Reference Manual
Program Development Reference Manual
UCSD Pascal Handbook

All of these publications are available from SofTech Microsystems. Personal Computing with the UCSD p-System and the UCSD Pascal Handbook are published by Prentice-Hall and are available in bookstores, as well.

The organization and general content of this MS-DOS-Hosted p-System Supplement is as follows:

- Chapter 1 presents an overview of the MS-DOS-Hosted p-System, backup and installation procedures, and a discussion of DOS file name syntax.

- Chapter 2 contains the procedures required for executing the p-System and provides information concerning its initial use.
Preface

- Chapter 3 describes all DOS file commands and gives individual procedures and examples for using them.

- Chapter 4 covers the Configure utility, with detailed procedures and displays for using all of its p-System configuration capabilities.

- Chapter 5 describes the MS-DOS units and explains the purpose and use of the machine-independent Driver Configuration unit (DRVCONFIG).

- Chapter 6 covers the MS-DOS device driver set and describes the RAM disk, with related discussions of memory allocation and extended memory. It also describes machine-specific drivers and provides information about low level I/O.

- Chapter 7 provides information to assist you in writing device drivers.

- Chapter 8 contains special machine-specific considerations involving hardware requirements, special keys, and Turtlegraphics.

- Chapter 9 covers the Diskformat, Extrapicks, Fdisk, NCICopy, and Startup utilities—and gives procedures and supporting information for using them.

- Chapter 10 contains programmer information, comprising unit and special software interfaces.
# Table of Contents

**Introduction** ........................................... 1–3  

**Overview** ............................................... 1–3  

**Backup Procedures** ............................... 1–4  
  Making a Floppy Disk Backup ....................... 1–4  
  Making a Fixed Disk Backup ....................... 1–6  

**Installing Optional Products** ................. 1–7  
  Copying Onto Your DOS Disks ..................... 1–7  
  Copying Onto Your p–System System Disk .......... 1–9  
  Copying Onto Your Fixed Disk .................. 1–9
# Table of Contents

DOS FILE NAMES ............................................. 1-11

EXECUTING THE p–SYSTEM ................................. 2-3

GENERAL .................................................... 2-3

EXECUTING THE p–SYSTEM ................................. 2-3
Executing – By Default .................................. 2-4
Executing – Mounting Virtual Volumes ............... 2-4
Error Messages ............................................ 2-5

USING THE p–SYSTEM ..................................... 2-6
Menus and Prompts ........................................ 2-6
Creating a New Boot Volume ............................. 2-7

DOS FILER ..................................................... 3-3

GENERAL .................................................... 3-3

FILER COMMANDS .......................................... 3-4
B(uild) ....................................................... 3-5
C(hange) ..................................................... 3-7
D(ismount) .................................................. 3-8
E xtended Directory ....................................... 3-9
Transfer F(rom DOS) ...................................... 3-11
H(elp) ........................................................ 3-14
L(ist Directory) ............................................ 3-15
M(ount) ....................................................... 3-16
P(refix) ....................................................... 3-17
Q(uit) ........................................................ 3-18
Table of Contents

R(emove File .............................................. 3-19
Transfer T(o DOS ...................................... 3-20
V(olumes .................................................. 3-22

CONFIGURE UTILITY ..................................... 4-3

GENERAL .................................................. 4-3
Main Configure Menu ................................. 4-5
Changing Driver Parameters ...................... 4-6
  Console Driver ......................................... 4-9
  Printer Driver ........................................ 4-13
  Disk Driver ............................................ 4-14
    Floppy Driver ...................................... 4-15
    RAM Disk Driver ................................... 4-18
    User-Defined Disk Drivers ..................... 4-19
  Remote Driver ....................................... 4-20
  Serial Driver ........................................ 4-22
  User-Defined Driver ............................... 4-22
Change OS Parameters ............................ 4-24
Install Device Drivers ............................ 4-28
Matrix Printer ......................................... 4-37
Table of Contents

MS-DOS ACCESS AND CONFIGURATION UNITS... 5-3

GENERAL ........................................... 5-3

FILEIO ........................................... 5-4
  dos_reset ...................................... 5-5
  dos_rewind ..................................... 5-5
  dos_rewrite .................................... 5-5
  dos_blk_read ................................... 5-6
  dos_blk_write .................................. 5-7
  dos_close ...................................... 5-7
  dos_rename ..................................... 5-8
  dos_flush ...................................... 5-8
  dos_prefix ..................................... 5-8
  dos_scan_title .................................. 5-9

DOSDIRINFO ..................................... 5-10
  DD_StartSearch .................................. 5-11
  DD_GetInfo ..................................... 5-11
  DD_FreeSpace ................................... 5-11

FILERUNIT ...................................... 5-12
  dos_show_dir ................................... 5-13
  dos_transfer .................................... 5-13
  dos_scan_name .................................. 5-14

VIRTVOl ........................................ 5-15
  VVMount ....................................... 5-15
  VVDismount ..................................... 5-16
  VVCreate ....................................... 5-16
**Table of Contents**

- **VVInfo** ............................................. 5–16
- **VVDevices** ........................................... 5–17
- **CALLDOS** ........................................... 5–18
- **DOS Function** ....................................... 5–19
- **StrToAsciiz** ......................................... 5–20
- **AsciizToStr** .......................................... 5–20
- **ExitToDOS** ........................................... 5–20

- **CLOCK UNIT** .......................................... 5–21

- **DRIVER CONFIGURATION UNIT** ....................... 5–22
  - **Description of the Routines** ......................... 5–23
  - **Installing and Replacing Drivers** .................. 5–24
  - **Device Numbers and Indexes** ....................... 5–25

- **DEVICE DRIVERS AND MEMORY ALLOCATION** .......... 6–3

- **GENERAL** ............................................. 6–3

- **MS-DOS DRIVER SET** ................................ 6–3
  - **Virtual Volume Driver** ............................. 6–3
  - **Virtual Console Driver** ............................ 6–4
  - **Virtual System Driver** .............................. 6–5
  - **Virtual Printer/Remote Driver** .................... 6–6
  - **Bridge Driver** ....................................... 6–6

- **MEMORY ALLOCATION** ................................ 6–7
Table of Contents

EXTENDED MEMORY ........................................ 6-8

RAM DISK .................................................... 6-9

MACHINE-SPECIFIC DEVICE DRIVERS .................. 6-10
  Console Driver .......................................... 6-10
  Fixed Disk Driver ...................................... 6-11
  Floppy Disk Driver .................................... 6-12
  Serial/Parallel Driver .................................. 6-13

LOW LEVEL I/O ........................................... 6-15
  Console Status Records ................................ 6-16
  Disk Status Records .................................... 6-16
  Printer Status Records ................................ 6-17

WRITING A DEVICE DRIVER ............................... 7-3

GENERAL .................................................. 7-3

WRITING A DRIVER ....................................... 7-3

SPECIAL CONSIDERATIONS ................................. 8-3

GENERAL .................................................. 8-3

HARDWARE REQUIREMENTS ................................. 8-3

SPECIAL KEYS .............................................. 8-4
Table of Contents

USING TURTLEGRAFICS ................................. 8-7

UTILITY PROGRAMS ......................................... 9-3

GENERAL ..................................................... 9-3

DISKFORMAT .................................................. 9-5

EXTRAPOOLS ................................................ 9-11

FDISK ........................................................ 9-13
   General Description ..................................... 9-14
   Main Menu ................................................ 9-18
   Select Fixed Disk Drive ................................ 9-20
   Create p—System Partition ............................. 9-20
   Delete p—System Partition ............................. 9-23
   Backup p—System Partition ............................. 9-24
   Restore p—System Partition ............................. 9-27
   Quit ....................................................... 9-30

NCICOPY ...................................................... 9-31

STARTUP ...................................................... 9-33
   Program File .............................................. 9-33
   Units Needed .............................................. 9-33
   Command File .............................................. 9-33
   Command Summary ........................................ 9-34
   Operating System Note .................................. 9-35
   System Program Names .................................. 9-35
   Example of Command Lines ............................. 9-36
Table of Contents

PROGRAMMER INFORMATION .................. 10–3
   GENERAL .................................. 10–3
   IBM SPECIAL UNIT ....................... 10–4
   TURTLEGRAPHICS ......................... 10–11
   PRNCONTROL UNIT ....................... 10–13
   FORMATTER UNIT ......................... 10–15
   FDISK UNIT ............................... 10–17
   KEYBOARD CODES ......................... 10–22
   VIDEO CONTROL CODES ................... 10–25
   EVENTS AND INTERRUPTS ................. 10–27

APPENDIX ..................................... A–1
   IORESULTS ................................ A–3

INDEX ........................................ 1–1
CHAPTER 1

INTRODUCTION
Introduction

The MS-DOS-Hosted p-System enables you to package p-System applications, making them fully compatible with the MS-DOS environment and allowing them to access MS-DOS data files. In addition, after running the applications, you’re not required to reboot DOS. In other words, you can execute and run the p-System just like a regular MS-DOS program. So, once you execute the p-System from the DOS prompt, both DOS and p-System files and disks can be accessed.

With the MS-DOS-Hosted p-System, you can also access MS-DOS devices which are supported by MS-DOS, but which are not supported by the p-System. These devices can be hard disks, tape units, and so on.

(This product requires a minimum memory of 128K and version IV.13 or later of the p-System. It is for use on an 8086/8088 microprocessor.)

OVERVIEW

The following components make up the MS-DOS-Hosted p-System:

* PSYSTEM.COM—a DOS program that starts the p-System.

* PSYSTEM.VOL—a file containing the p-System.

The DOS program that executes the p-System is called PSYSTEM.COM. In addition to executing the p-System, it gives you the option of listing parameters to specify the virtual volumes you want to be mounted. The first parameter you list specifies the boot volume, while the subsequent parameters you list specify any additional volumes to be mounted. (For detailed information concerning the DOS bootstrap program, refer to the "Installation" section presented later in this chapter.)
Introduction

The p-System, located within PSYSTEM.VOL, contains all the standard p-System features (filer, editor, and so on), plus many features which are specific to the DOS environment. Some of these features include:

- **DOSFILER**—a program for managing DOS files and for moving files between DOS and the p-System.

- **CONFIGURE**—a program for changing the p-System configuration.

- Automatic access to DOS files from p-System programs without requiring any change to the program.

- For programmers—a number of units are provided for accessing DOS features easily from your Pascal programs.

**BACKUP PROCEDURES**

You shouldn’t use the diskettes (disks) received in the p-System release package for day-to-day operations. Instead, make copies of these disks.

The following two backup procedures tell you how to back up the disks you received. The first procedure is for backing up your disks on a regular (floppy) disk, while the second procedure covers backing up your disks on a fixed (hard) disk.

**Making a Floppy Disk Backup**

To back up your disks on a floppy disk, use the DISKCOPY command in DOS.

With the A> prompt displayed, enter:

```
A$DISKCOPY A: B: <ENTER>
```

1—4 1700101:01
Introduction

A prompt now tells you to insert the source disk into A: and the destination disk into B:. So, place the first disk you received with this release into drive A: and a blank disk into B:; then, press <space>. This makes the backup copy of the disk. Be sure to label the new disk so you will know what disk it was made from.

Once the copy process is completed, you are asked if you want to copy another disk. Respond by pressing Y, to repeat this process, until all the disks you received with this release have been copied.

If you wish to make a bootable p-System disk, perform the following procedure:

1. Place your DOS system disk into drive A:.
2. Place a blank formatted disk into drive B:
3. Enter SYS B:<enter>. This copies DOS onto the p-System disk.
4. Enter COPY COMMAND.COM B:<enter>. This copies the DOS file COMMAND.COM to the p-System disk.
5. Place the p-System disk labeled PSYSTEM into drive A:.
6. Enter COPY A:*.* B:<enter>. This copies all the p-System files to the new disk.

A directory listing of your p-System system disk should now list the following files:

PSYSTEM.COM
PSYSTEM.VOL
CONFIG.SYS
COMMAND.COM
Introduction

This disk is now a bootable DOS disk that also contains the p-System.

Making a Fixed Disk Backup

If you have a fixed disk, you may want to copy the files from the p-System release onto your fixed disk, rather than making a floppy disk copy.

The disk that is labeled PSYSTEM is a DOS disk that contains the following files:

PSYSTEM.COM
PSYSTEM.VOL
CONFIG.SYS

NOTE: The CONFIG.SYS file is used to tell DOS that more open files are needed than are normally given. If you already have such a config file, this one is not needed.

Copy these files to your fixed disk by placing the PSYSTEM disk into drive A and then entering this command:

COPY A:\*.* C: (except)

The second disk, MISC, contains the following files:

MISC.VOL
DRIVERS.VOL
2WORD.VOL
Introduction

You should also copy the file MISC.VOL to your hard disk. Do this by placing the MISC disk into drive A and entering this command:

```
COPY A:\MISC.VOL C:\<enter>
```

The remaining files are used if you want to change your p—System configuration. If desired, you can also copy these files to your fixed disk in the same way that the MISC.VOL file was copied.

INSTALLING OPTIONAL PRODUCTS

Optional products, such as the compiler and the Advanced Development Tool Kit, are distributed on Universal Medium® disks. The p—System can read these disks—but they are not compatible with DOS. To copy files from this type of a volume onto a DOS volume, use the following procedures.

Copying Onto Your DOS Disks

To copy files from Universal Medium disks onto DOS disks, perform these steps:

1. Place your p—System system disk into drive A:. (This is the disk that contains PSYSTEM.COM and PSYSTEM.VOL.)

2. Execute the p—System by entering PSYSTEM<enter>.

3. When the p—System command menu appears, press X.

4. Then, in response to the prompt—Execute what file?—enter DOSFILE<enter>.
Introduction

5. Place a formatted disk into drive B.

6. Create a p-System virtual volume on the disk in drive B by pressing B.
   
   This prompts you for a file name. Respond to the prompt by entering "B:\SYS.VOL<enter>.
   
   The next prompt is for the p-System volume name. Respond to it by pressing <enter>.
   
   Next, you are asked for the volume size. Enter the maximum size indicated and press <enter>.
   (This volume size must be large enough to accommodate all of the files on the p-System disks.)
   
   Finally, a prompt is displayed asking if you want the volume mounted. Respond to this prompt by pressing Y.

7. Exit the DOS filer by pressing Q.

8. Enter the p-System filer by pressing F.

9. Lock the filer in memory by pressing F.

10. Remove your p-System disk from drive A.

11. Do the following for each of the Universal Medium disks in your p-System release package:
   
   Place the disk into drive A and press T.
   
   Respond to the "Transfer what file?" prompt by entering #4:=.SYS:.<enter>.

12. To return to the p-System command menu, place the p-System system disk back into drive A and press Q.
Introduction

Copying Onto Your p—System System Disk

You may wish to install an optional product, such as a compiler, on your p—System system disk. To do this, use the following procedure in place of the previous one:

1. Place your p—System system disk into drive A:. (This disk contains PSYSTEM.COM and PSYSTEM.VOL.)

2. Execute the p—System by entering PSYSTEM<enter>.

3. Then, enter the p—System file by pressing F.

4. Place the Universal Medium release disk into drive B: and press T.

5. Respond to the "Transfer what file?" prompt by entering #5:=,*$<enter>.

6. When all the files have been copied, press Q to return to the p—System command menu.

Again, to perform this procedure successfully, you must have sufficient space on your p—System volume for these additional files.

Copying Onto Your Fixed Disk

To copy the files from the optional products disks onto your fixed disk, use the following procedure:

1. Execute the p—System by entering PSYSTEM<enter>.

2. When the p—System command menu appears, press X.
Introduction

3. This displays the "Execute what file?" prompt. Respond to it by entering DOSFILER <enter>.

4. Create a p-System virtual volume by pressing B.

   Respond to the prompt for a file name by entering PSYS.VOL<enter>.

   The next prompt requests the p-System volume name. Respond to it by pressing <enter>.

   Next, you are asked for the volume size. Enter 1500 for the size.

   Finally, you are asked if you want the volume mounted. Respond to this prompt by pressing Y.

5. Exit the DOS flier by pressing Q.

6. Enter the p-System flier by pressing F.

7. Do the following for each of the remaining disks in your p-System release package:

   Place the disk into drive A and press T.

   Respond to the "Transfer what file?" prompt with #4=PSYS:.

   For any of the optional products you may wish to install, 1500 blocks should provide ample room.
DOS FILE NAMES

This section describes the file specification syntax used by DOS. When opening a file, the p-System examines the file name to determine whether the file is a DOS file or a p-System file. If the file can be either a DOS file or a p-System file, the system first attempts to open the file as a DOS file. Once it is determined that the file is not a DOS file, an attempt is made to open the file as a p-System file. Thus, when referring to DOS files, you use the DOS file name syntax; and when referring to p-System files, you use p-System file name syntax.

NOTE: When referring to a p-System file on the p-System prefix volume, you should start the name with a colon (:) in order to remove any ambiguity. When referring to a p-System file on other than the prefix volume, you should start the file name with the unit number; for example: #4 or a volume name, such as MYVOL, followed by a colon. When referring to a DOS file, always begin the file specification with the DOS drive letter; for example—A:—followed by a colon, as shown. In addition, if p-System code files are to be executed, they must be on a p-System volume, not on a DOS volume.

A DOS file specification begins with an optional drive letter that may be from A to Z. It is followed by a colon (:) to separate it from the rest of the file specification. If it is omitted, the default drive is automatically assumed.

The path specification (optional) follows the drive letter. It is needed to access subdirectories on disks with subdirectories created by DOS, version 2.0 and later. The path is a list of directories (one or more) that specifies which directory to look in to find the next directory in the list or to find the file (if no more directories exist in the list). The directories in the list are separated by the backslash character (\). Each directory is a file name that must meet the file name syntax requirements specified in the next paragraph. If the path begins with a backslash, the search starts with the root directory of the disk.
Introduction

specified; otherwise, the search starts with the current directory.

The file name follows the path specification. File names consist of 1 to 8 characters followed by an optional extension of up to 3 characters. The extension is separated from the name by a period (.) The this is the only valid use of a period (.) in a DOS file name.

DOS wild cards ('*' and '?') may be used in file names to indicate a group of files.

The following are valid file names showing the various options that may be used; a short description is to the right of the file name. Uppercase or lowercase characters may be used anywhere in a file name.

A:\DATABASE.COM - drive A, file DATABASE, extension COM
E:\GARY\LIST\WORK.LST - drive E, path \GARY\LIST, file WORK, extension LST
\GARY\LIST\WORK.LST - default drive, path \GARY\LIST, file WORK, extension LST
WORK.LST - default drive, current directory, file WORK, extension LST
c:\list\work.list - drive A, current directory plus directory LIST, file WORK, extension LST
B:\*.* - all files in root directory of disk in drive B:
Introduction

Formal file name syntax diagrams are given below. These define the valid file names that may be used. Any parts in capital letters must be used exactly as shown, except that all actual entries may be uppercase or lowercase letters.

Items named with lowercase names must be supplied by the user. Items enclosed in square brackets ([ and ]) are optional items. Items separated by a bar (|) mean that you can enter any one of the items separated by bars. An ellipsis (…) means that you can repeat the item as many times as desired. Items separated by a dash (–) mean a range of items; for example, A–Z means any letter from A to Z, including A and Z. Except for square brackets and vertical bars, you must include all punctuation, such as colons and backslashes.

<table>
<thead>
<tr>
<th>Item</th>
<th>Definition</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>char</td>
<td>A-Z[0-9]</td>
<td>Max of 8 characters</td>
</tr>
<tr>
<td>filename</td>
<td>char[char][char]…</td>
<td>Example: .COM</td>
</tr>
<tr>
<td>dirname</td>
<td>.….filename</td>
<td>Example: A:</td>
</tr>
<tr>
<td>ext</td>
<td>.char[char][char]</td>
<td></td>
</tr>
<tr>
<td>drive</td>
<td>A-F;</td>
<td></td>
</tr>
<tr>
<td>path</td>
<td>.[dirname] [.dirname]…</td>
<td>Max of 63 characters</td>
</tr>
<tr>
<td>filespec</td>
<td>[drive][path][filename][extension]</td>
<td></td>
</tr>
</tbody>
</table>
CHAPTER 2
EXECUTING
AND USING
THE p-SYSTEM
Executing/Using The p-System

GENERAL

This chapter describes how to execute the p-System under MS-DOS on the IBM Personal Computer, the IBM Portable Computer, and the IBM Personal Computer XT. In this supplement, references to the IBM Personal Computer (or IBM PC) generally include the IBM Portable Computer and the IBM Personal Computer XT.

The first section in this chapter explains how to execute the p-System under DOS on the IBM Personal Computer. The remaining sections describe various aspects of the p-System which are important to understand when you first start using it. The topics include prompts and menus, and creating a new bootable p-System volume.

Note that underlining is used throughout this supplement to indicate that something is supposed to be typed at the keyboard. For example: "Press <enter> to display...."

EXECUTING THE p-SYSTEM

You can execute the p-System by default (with a prefixed directory) or by specifying parameters (options) with which to mount virtual volumes. (For the minimum hardware configuration required to run the p-System, refer to Chapter 8, "Special Considerations.")
Executing/Using The p-System

Executing — By Default

Executing the p-System in this manner requires that both PSYSTEM.COM—the DOS program that executes the p-System—and a file called PSYSTEM.VOL—which contains the p-System operating system—be in the current directory. Then, all that is required to execute the p-System is to enter PSYSTEM after the DOS prompt and press <enter>, as shown here:

A>PSYSTEM\ENTER

After several seconds, this displays the p-System command menu, as follows:

Command: Edit, Run, File, Comp, Link, Execute, Assem, Debug, ? [ ]

NOTE: At any time, you can return to the DOS prompt (A>) from the p-System command menu by pressing H for H(alt).

Executing — Mounting Virtual Volumes

When executing the p-System, you can also mount virtual volumes. This is done when you enter PSYSTEM in response to the DOS prompt. To mount virtual volumes, you must enter parameters (after PSYSTEM) that are the file names of the virtual volumes you want to have mounted. If you don’t enter any parameters, there is an implied (default) parameter of PSYSTEM.VOL. That’s why—if you have PSYSTEM.VOL in your prefixed directory—you’re not required to enter anything else, other than PSYSTEM.
Executing/Using The p-System

To mount more than one virtual volume (the boot virtual volume) while booting, you must specify all the parameters. First, you must specify the boot volume. (This is the DOS file that contains the p-System.) Then, separate the parameters (file names) with spaces, listing just the DOS file names. Enter such a parameter listing of file names like this:

```
>psystem.psystem.vol.units.vol<enter>
```

This mounts PSYSTEM.VOL and UNITS.VOL, and displays the p-System command menu. Refer to the next section—"Using the p-System"—in this chapter.

Error Messages

The following error messages indicate problems you may encounter while executing the p-System:

- Bad command format—No virtual volumes configured.
- Can't find directory
- Can't find <filename>—Virtual volume file was not found.
- Can't find SYSTEM.CONFIG
- Can't find SYSTEM.PME.86
- Disk error reading directory
- Disk error reading SYSTEM.CONFIG
- Disk error reading SYSTEM.PME.86

1700101:02 2-5
Executing/Using The p—System

- Incorrect DOS version, 2.0 or later required
- No Directory on volume—The boot virtual volume doesn’t have a p—System directory.
- Not enough memory—At least 128K must be available to the p—System.

USING THE p—SYSTEM

If you are just starting to use the p—System, you should read the information in the rest of this chapter.

Menus and Prompts

As shown in the previous section, the p—System command menu looks like this:

```
Command: E(dit), R(un), F(ile), C(omp), L(ink), X(ecute), A(ssign), D(egment), ? [ ]
```

You can select commands—or options—from p—System menus similar to this one by pressing a single letter (the letter that appears before the parenthesis). If a question mark appears at the end of a menu, it indicates that more options are on the menu than can be displayed on one line. Pressing ? displays more of the menu. The command menu options include the editor, filer, compiler, and so on. These command menu options usually have menus or prompts of their own.

A prompt is a question that requires you to enter a response followed by <enter>. The following is a typical prompt (the response is underlined):

```
Execute what file ? PROGS (enter)
```
Executing/Using The p–System

Complete descriptions of menus and prompts are given in the standard p–System documents, including Personal Computing with the UCSD p–System.

NOTE: When using the p–System editor(s), all files should be created with p–System names. If DOS file names are used, unpredictable results may occur.

Creating a New Boot Volume

To create a new bootable p–System volume, copy an existing bootable volume onto another disk. (To do this, follow the instructions under "Backup Procedures," in Chapter 1.)

The following files must reside on the new system volume:

- SYSTEM.PASCAL
- SYSTEM.MISCINFO
- SYSTEM.PME.86 or SYSTEM.PME.87
- SYSTEM.CONFIG
- SYSTEM.LIBRARY
- SYSTEM.FONT (needed only for Turtlegraphics)

You should keep SYSTEM.FILER on the new system volume as well.

SYSTEM.PME.86 and SYSTEM.PME.87 are two different versions of the p–machine emulator (PME). The SYSTEM.PME.87 file is used by computers which have an 8087 floating point coprocessor. The SYSTEM.PME.86 file is used by computers that do not have an 8087. The advantage of using the 8087 is that it performs floating point calculations more quickly than software can.
NOTE: If a SYSTEM.PME.87 file is on the boot volume, an 8087 is assumed to be present.

The SYSTEM.PME.86 file on the bootable system volume contains 4-word floating point routines which do not use the 8087. If you have an 8087, but SYSTEM.PME.87 is not present on the boot volume, the system will look for SYSTEM.PME.86 at boot time and load it (if it is available).

If you have an 8087, you should use the 8087 REALOPS unit, REAL.8087.CODE. In order to do this, you should use the Library utility (described in the Operating System Reference Manual) to place the 8087 REALOPS unit into a new copy of SYSTEM.PASCAL. After you move the 8087 REALOPS unit into the new SYSTEM.PASCAL, you should move any units—except the old REALOPS—from the current SYSTEM.PASCAL over to the new SYSTEM.PASCAL. It is a good idea to save the old copy of SYSTEM.PASCAL before you do this.

Included with the PME's is a file SYSTEM.PME.86.2. This file contains 2-word floating point routines which also do not use the 8087. In order to make your system a 2-word system, you must first use the flier to transfer SYSTEM.PME.86.2 to the system volume. Then, while saving the old SYSTEM.PME.86 or .87, Chnage SYSTEM.PME.86.2 to SYSTEM.PME.86. You must then use the Library utility (described in the Operating System Reference Manual) to place the file REALOPS.2.CODE into SYSTEM.PASCAL (remember to save the old one). Also use the Library utility to install IBMSPECIAL.2.CODE; and, if you are using Turtlegraphics, TURTLE2.CODE into SYSTEM.LIBRARY.
CHAPTER 3
DOS FILER
GENERAL

The DOSFILER program allows you to:

1. Display a DOS directory.
2. Build (create) a new virtual volume.
3. Display information concerning virtual volumes.
4. Mount a virtual volume.
5. Dismount a virtual volume.
6. Transfer a file from a DOS disk to a p—System disk.
7. Transfer a file from a p—System disk to a DOS disk.
8. Set the default DOS disk and path name.
9. Remove a file from a DOS disk.
10. Change the name of a DOS file.

When DOSFILER is executed, the following menu appears:

DOS Filer: L(list dir), E(exe dir), C(change), R(refile), T(to DOS), ? [ ]

Pressing ? displays the remainder of the DOS filer's menu:

DOS Filer: F(from DOS), M(mount), D(dismount), V(olume), B(uild), H(e)lp, Q(uit) [ ]
FILER COMMANDS

The following subsections describe the DOS filer commands and give examples of their use. Commands are listed in alphabetical order (by command letter), with each new command beginning on a new page. All commands apply only to DOS files with the exception of transferring to/from a p-System file. The commands are selected by pressing the first letter of the command. The examples shown have the prompts displayed exactly as they appear on the screen: user response to the prompts is also shown and is underlined.

B  Build a virtual volume.

C  Change the name of a file.

D  Dismount a virtual volume.

E  Extended directory listing, including hidden and system files.

F  Transfer a DOS file to a p-System file.

H  Display Help information about T(o and F(from commands.

L  List a DOS directory.

M  Mount a virtual volume.

P  Set the default DOS disk and path.

Q  Quit the DOS filer program.

R  Remove a DOS file.

T  Transfer a p-System file to a DOS file.

V  Display virtual volume information.
B(uild)

On the menu: B(uild)

This command allows you to build (create) a new virtual volume.

The build command displays this prompt:

DOS File Name? _

To create a new virtual volume, enter an appropriate name. In this example, we'll name the virtual volume RPORT and enter it as follows:

A:\RPORT.VOL<enter>

This displays this prompt:

Volume name for virtual volume (<enter> for "RPORT")

If accepted at this point, by pressing <enter>, this is the p-System volume name that will be created. (The system will place a p-System volume inside RPORT.VOL.) However, before pressing <enter>, you may change RPORT to another name by entering some other volume name and then pressing <enter>. In this example, though, we'll leave the name as RPORT and press <enter>. The following typical prompt is then displayed. It shows the amount of unused space on the volume and prompts you for the number of blocks your new virtual volume requires.

Size of the virtual volume in blocks (maximum of 75)? _
DOS Filer

Enter the number of blocks required and press <enter>.

NOTE: The virtual volumes are not completely allocated by this command. Instead, only a small amount of space is allocated: and the file containing the volume is allowed to expand, as needed, up to the specified size of the volume. This means that a directory listing from the p—System filer shows the maximum amount of space entered here for the virtual volume, while the DOS filer and DOS directory (dir) commands only show the amount of space actually used. This has certain advantages for both p—System and DOS users.

A final display now appears which indicates that the virtual volume is created and asks—with a (y/n) prompt—if you want to mount it.
C(hange)

On the menu: C(hange

This command changes the name of a DOS file.

The change command can be used to change the name of a file, to move a file from one directory to another, or both.

Upon entering the change command, the prompt "Change what file?" appears. Any valid DOS file name may be entered and the name may include DOS wild cards ("*" and "). For each file that matches the name entered, you are asked, "New name for <old file name>?". At this time, you may enter the new name for the file, including a new path name if you wish to move the file to another directory. Entering only an <enter> will not change the name of the file and will go on to the next file. Entering an <esc><enter> will return you to the DOS Filer menu.

The following example shows how to change a file name:

Change what file? MYFILE.TXT<enter>
New name for MYFILE.TXT<enter>
New name for USERLIB.TXT? <enter>
B:\USERLIB.TXT --> user.txt

In this example, MYFILE.TXT was not changed; and the file USERLIB.TXT was changed to USER.TXT.
DOS Filer

D(ismount)

On the menu: D(ismount

This command allows you to (1) dismount a virtual volume you no longer need or (2) dismount a virtual volume to make room for mounting another one.

The following display shows the Dismount menu with four virtual volumes displayed:

Dismount: O..3, A(ll), N(ext), P(revious, Q uit

0 - #10: IV.13 A:\SYSTEM.VOL
1 - #11: UNITS A:\UNITS.VOL
2 - #12: A:\CONFIG.SYS
3 - #13: 

On the menu, 0..3 allows you to select the virtual volume which you want to dismount. A(ll) dismounts all the virtual volumes, except the boot device volume, and returns to the DOS Filer menu. N(ext) and P(revious are used when more virtual volumes are configured than can fit on one screen. N(ext) displays the next screen of virtual volumes, and P(revious displays the previous screen of virtual volumes.
E(xtended Directory)

On the menu: E[xt dir]

This command lists all the files, including hidden files, in more
detail than the L(ist dir) command. (See L(ist dir) for more
information.)

All files are listed with their last modification date, last
modification time, number of full blocks used (sectors),
number of bytes in the last block of the file, and flags. The
flags are coded by displaying a letter for each flag that is true.
For detailed information about the meaning of the flags, see
your Disk Operating System Manual. The letter codes
displayed are:

R    READ ONLY
H    HIDDEN FILE
S    SYSTEM FILE
D    DIRECTORY FILE
A    ARCHIVING REQUIRED
DOS Filer

The following is an extended directory listing of a DOS disk:

```
List directory of what volume? A:\ (enter)

Directory of A:\.

IBMBIOS.COM 08/Mar/83 12:00:00 9 812 RHSA
IBMDOS.COM 08/Mar/83 12:00:00 34 286 RHSA
COMMAND.COM 08/Mar/83 12:00:00 36 285 A
ANSI.SYS 08/Mar/83 12:00:00 4 128 A
FORMAT.COM 08/Mar/83 12:00:00 12 384 A
CMDSYS.COM 08/Mar/83 12:00:00 13 285 A
SYS.COM 08/Mar/83 12:00:00 3 384 A
DISKCOPY.COM 08/Mar/83 12:00:00 5 356 A
DISKCOMI.COM 08/Mar/83 12:00:00 5 25 A
EDLIN.COM 08/Mar/83 12:00:00 9 812 A
MODE.COM 08/Mar/83 12:00:00 7 67 A
PRINT.COM 08/Mar/83 12:00:00 9 812 A
RECOVER.COM 08/Mar/83 12:00:00 5 286 A
ASSIGN.COM 08/Mar/83 12:00:00 2 384 A
BASIC.COM 08/Mar/83 12:00:00 32 384 A
BASICA.COM 08/Mar/83 12:00:00 81 384 A
DEBUG.COM 08/Mar/83 12:00:00 24 128 A
```
Transfer F(from DOS)

On the menu: F(from DOS)

This command transfers files from DOS disks to p—System disks.

Upon entering the transfer command, the question, "Transfer what DOS file?" appears. Any valid DOS file name may be entered, and the name may include DOS wild cards. For each file that matches the name entered, you will be asked, "Transfer <dos name> to what p—System file ("$<enter>" for <p—System default>)?". The p—System default name is derived from the DOS file name. This is done by translating the three—character DOS suffix into a standard p—System suffix when possible. The following translations are used:

<table>
<thead>
<tr>
<th>DOS Suffix</th>
<th>p—System Suffix</th>
</tr>
</thead>
<tbody>
<tr>
<td>TXT</td>
<td>TEXT</td>
</tr>
<tr>
<td>CDE</td>
<td>CODE</td>
</tr>
<tr>
<td>VOL</td>
<td>SVOL</td>
</tr>
<tr>
<td>DTA</td>
<td>DATA</td>
</tr>
<tr>
<td>BAK</td>
<td>BACK</td>
</tr>
</tbody>
</table>

An <enter> response for the p—System file name will not transfer the file. An <esc><enter> response will return to the DOS Filer menu. Any other response is assumed to be a p—System file name. If this name contains a "$", then the "$" is replaced with the default p—System name that is indicated. In the case where the response is $<enter>, the file will be transferred to the p—System prefixed volume with the name indicated.

Files are transferred in one of two ways: block by block with no conversion or with text file conversions applied. The text file conversions involve removing line feed characters, expanding tab characters, and breaking up text into pages of 2 blocks. (See the Internal Architecture Reference Manual for details of p—System text file format.)
DOS Filer

By default, if the p-System file name ends in .TEXT, then all translations are performed; otherwise, none are. This may be overridden by adding an options specification to the end of the DOS file name. The option specification is a series of code letters placed between left and right brackettes ([*] and [*]). The code letters are:

D    DLE
T    TABS
E    END OF FILE
L    LINEFEEDS
C    CONTROL
A    ALL
N    NONE

DLE — When transferring to the p-System, leading blanks on a line will be converted to a DLE code in the p-System file to save space. When transferring from the p-System, DLE codes will be expanded into a sequence of blanks in the DOS file.

TABS — When transferring to the p-System, a TAB character will be converted into enough spaces to position the next character at the same column that the TAB character does in DOS. When transferring from the p-System, a sequence of blanks will be converted into enough TABs and spaces to position the next nonblank character at the same column that the blanks did in the p-System.

END OF FILE — When transferring to the p-System, a control-z character in the DOS file will be taken as the end of file. When transferring from the p-System, a control-z character will be inserted at the end of the file.

LINE FEED (LF) — When transferring to the p-System, all LF characters will be removed, since they are paired with a CR character; and a CR character implies an LF in the p-System. When transferring from the p-System, an LF will be added after each CR.
CONTROL — When transferring to the p—System, all control characters with ASCII codes from 0 to 31 (except 9, 10, 13, and 26) are removed from the text. This option does not apply when transferring from the p—System.

ALL — When selected, all the translations are performed on the file being transferred. It is equivalent to selecting all five of the previous options.

NONE — When selected, no translation is performed on the text file being transferred.

The following example shows how to transfer DOS files to p—System volumes:

```
Transfer what DOS file? MYFILE.TXT
Transfer MYFILE.TXT to what p—System file
("$<enter>" for MYFILE.TXT)? $<enter>
B:\MYFILE.TXT --> MYFILE.TXT
Transfer SCRATCH.VOL to what p—System file
("$<enter>" for SCRATCH.VOL)? $<enter>
Transfer DOS.X to what p—System file
("$<enter>" for DOS.X)? XXX.DOS$<enter>
B:\DOS.X --> XXX.DOS
```

In this example, SCRATCH.VOL was not transferred, MYFILE.TXT was transferred using the default name, and DOS.X was transferred using a specified name.
CONTROL – When transferring to the p–System, all control characters with ASCII codes from 0 to 31 (except 9, 10, 13, and 26) are removed from the text. This option does not apply when transferring from the p–System.

ALL – When selected, all the translations are performed on the file being transferred. It is equivalent to selecting all five of the previous options.

NONE – When selected, no translation is performed on the text file being transferred.

The following example shows how to transfer DOS files to p–System volumes:

Transfer what DOS file? *
Transfer MYFILE.TXT to what p–System file
("$<ENTER>" for MYFILE.TXT)? $<ENTER>
B:\MYFILE.TXT --> MYFILE.TEXT
Transfer SCRATCH.VOL to what p–System file
("$<ENTER>" for SCRATCH.VOL)? $<ENTER>
Transfer DOS.X to what p–System file
("$<ENTER>" for DOS.X)? XXX.DOS$<ENTER>
B:\DOS.X --> XXX.DOS

In this example, SCRATCH.VOL was not transferred, MYFILE.TXT was transferred using the default name, and DOS.X was transferred using a specified name.
DOS Filer

HELLP

On the menu: H(elp)

This command displays the following information about translation and wild card options:

When transferring T(o) or F(rom) DOS the following translation options may be selected by enclosing them in brackets at the end of the DOS file name:

- D = delete blank compression
- T = tab expansion
- E = crtl-z eof handling
- L = line feed handling
- C = control character removal
- A = all the above
- N = no special handling

For example, B:FILE.TXT[DELCE] specifies all the translations except tab expansion. By default, all the translations are performed if the p-System file name ends in "TEXT" otherwise none of the translations are performed.

Wild card characters may be used in specifying what files are to be transferred. p-System wild cards should be used when transferring T(o) DOS and DOS wild cards (" and ?) when transferring F(rom) DOS.

An empty response to the destination file name will skip the file. A "S" in the response will be replaced with the default name. An <esc><cr> response will terminate the transfer operation.
L(list Directory)

On the menu: L(list dir)

This command lists all the files in a directory, showing important information about each file.

The files are listed along with their last modification date, last modification time, number of blocks used (sectors), and number of bytes used in the last block of the file. Hidden files are not listed with this command. (See the E(xt dir command for more information.)

The following is a directory listing of a DOS disk:

```
List directory of what volume? A:

Directory of A:\e:

<table>
<thead>
<tr>
<th>File</th>
<th>Date/Time</th>
<th>Blocks</th>
<th>Sectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMAND.COM</td>
<td>08/Mar/83</td>
<td>35</td>
<td>256</td>
</tr>
<tr>
<td>ANSI.SYS</td>
<td>08/Mar/83</td>
<td>4</td>
<td>128</td>
</tr>
<tr>
<td>FORMAT.COM</td>
<td>08/Mar/83</td>
<td>12</td>
<td>384</td>
</tr>
<tr>
<td>CHKDSK.COM</td>
<td>08/Mar/83</td>
<td>13</td>
<td>256</td>
</tr>
<tr>
<td>SYS.COM</td>
<td>08/Mar/83</td>
<td>3</td>
<td>384</td>
</tr>
<tr>
<td>DISKCOPY.COM</td>
<td>08/Mar/83</td>
<td>5</td>
<td>306</td>
</tr>
<tr>
<td>DISKCOMP.COM</td>
<td>08/Mar/83</td>
<td>5</td>
<td>25</td>
</tr>
<tr>
<td>EDLIN.COM</td>
<td>08/Mar/83</td>
<td>9</td>
<td>512</td>
</tr>
<tr>
<td>MODE.COM</td>
<td>08/Mar/83</td>
<td>7</td>
<td>67</td>
</tr>
<tr>
<td>PRINT.COM</td>
<td>08/Mar/83</td>
<td>9</td>
<td>612</td>
</tr>
<tr>
<td>RECOVER.COM</td>
<td>08/Mar/83</td>
<td>5</td>
<td>256</td>
</tr>
<tr>
<td>ASSIGN.COM</td>
<td>08/Mar/83</td>
<td>2</td>
<td>384</td>
</tr>
<tr>
<td>BASIC.COM</td>
<td>08/Mar/83</td>
<td>32</td>
<td>384</td>
</tr>
<tr>
<td>BASICA.COM</td>
<td>08/Mar/83</td>
<td>51</td>
<td>384</td>
</tr>
<tr>
<td>DEBUG.COM</td>
<td>08/Mar/83</td>
<td>24</td>
<td>128</td>
</tr>
</tbody>
</table>
```
DOS Filer

M(ount)

On the menu: M(ount)

This command places a virtual volume on line.

The mount command displays the following prompt:

Mount what file? _

In response to this prompt, let's mount all files ending in .VOL. To do this, enter the DOS wild card _, followed by .VOL; then, press <enter>. This should result:

Mount what file? _ .VOL
A:\PSYSTEM.VOL mounted.
A:\UNITITS.VOL mounted.
P(refix

On the menu: P(refix

This command sets the DOS default drive and path name.

When a file name doesn’t specify a particular drive or path name, the prefix or default is used. This command allows this to be set. The drive may be set by entering only the drive specification, or both the drive and path may be set by entering the drive specification and path name.

The following two examples show how a default drive is set, and how a default drive and path are set.

To set a default drive, press P. The following prompt appears:

```
Current prefix is A:
New prefix? _
```

Now, enter B: and press <enter>. This appears:

```
New prefix is B: 
```

To set a default drive and path, press P. This prompt appears:

```
Current prefix is B:
New prefix? _
```

Then, enter adir and press <enter>. This appears:

```
New prefix is B:\ADIR
```

1700101:03
DOS Filer

QUIT

On the menu: QUIT

This command is used to end the DOS filer program and return to the p-System command menu.

When this command is selected, the DOS filer program is terminated, and the p-System command menu is displayed on the screen.
R(Emove File)

On the menu: R(Emove

This command removes a file from a DOS disk.

Upon entering the R(Emove command, the question "Remove what file?" appears. Any valid DOS file name may be entered and the name may include DOS wild cards. For each file that matches the name entered, you will be asked. "Remove <filename> (y/n)?". A response of y will remove the file; a response of n will not remove the file; while a response of <esc> will return to the DOS Filer menu.

The following example shows how to remove a DOS file:

Remove what file?  *cde*<enter>
Remove B:\DOFSFILE.CDE (y/n)?  n
Remove B:\AFILE.CDE (y/n)?  y
Removed.
Remove B:\SDIRINFO.CDE (y/n)?  n

In this example, three files matched the pattern **.cde*; but only the file, AFILE.CDE, was removed.
Transfer T(o DOS

On the menu: T(o DOS

This command transfers files from a p-System disk to a DOS disk.

Upon entering this command, the question "Transfer what p-System file?" appears. Any valid p-System file name may be entered and the name may include p-System wild cards. For each file that matches the name entered, you will be asked, "Transfer <p-System name> to what DOS file (*$<enter>" for <dos default>?)". The DOS default name is derived from the p-System file name.

An <enter> response for the DOS file name will not transfer the file. An <esc> <enter> response will return to the DOS Filer menu. Any other response is assumed to be a DOS file name. If this name contains a "$", then the "$" is replaced with the default DOS name that is indicated.

The following example shows how to transfer p-System files to DOS volumes:

Transfer what p-System file? m: text<enter>
Transfer SOME.TEXT to what DOS file
(*$<enter>" for SOME.TXT)? b:\<enter>
BETA:SOME.TEXT --> b:\SOME.TEXT
Transfer MYFILE.TEXT to what DOS file
(*$<enter>" for MYFILE.TXT)? f:\dos<enter>
B:\MYFILE.TXT already exists - remove it? (y/n)? n
RAMDISK:MYFILE.TEXT --> MYFILE.TXT[delc]

3-20 1700101:03
In this example, SOME.TEXT was transferred using the default name along with a specified path. The file MYFILE.TEXT is to be transferred to MYFILE.TXT with all text file translations except tab expansion. In this case, a file, MYFILE.TXT, already existed on the DOS disk; thus, the (y/n) prompt asking for approval to remove it.
DOS File

**V(olumes)**

On the menu: **V(olumes)**

This command displays the virtual volumes that are on line, along with associated information about these volumes.

The following example shows the type of information that is in a **V(olumes)** display:

```
    Virtual Volume Configuration

      Drive  Volume  Access  DOS File
          10    IV.13    R/W     A:\PSYSTEM.VOL
          11    UNITS    R/W     A:\UNITS.VOL
          12              R/W     A:\CONFIG.SYS
          13
```

In the above example, the Drive column lists the p-System unit number; the Volume column gives the p-System volume name; the Access column indicates the type of access—R/W for read/write and R/O for read only; and the DOS File column lists the DOS file that contains the p-System volume. The DOS file, CONFIG.SYS, is not a p-System volume; it's a regular DOS file that has no p-System information in it. Thus, it is listed without a p-System volume name.
CHAPTER 4
CONFIGURE
UTILITY
GENERAL

Configure is used to set various p—System options. Most of these options affect the way peripheral devices behave (such as the video display, printer, or disk drives). Also, Configure allows you to add drivers for new peripherals (or change drivers for existing peripherals).

This section describes the p—System drivers for the IBM Personal Computer family of machines. (Refer to Chapter 6 for detailed information about all the MS—DOS—hosted drivers.)

CAUTION: Before you use Configure, be sure you have backed up the disk you are reconfiguring. This is an important safety precaution since, if you make a mistake, your disk may no longer bootstrap correctly.

The Configure utility enables you to configure the p—System so that it supports your peripheral devices, allowing you to:

- Select the driver desired for each available device number.
- Select the characteristics for storage devices (such as sectors per track, tracks per side, bytes per sector, step rate, and so forth).
- Select the baud rate, parity, word length, number of stop bits, and protocol for REMIN:, REMOUT:, and asynchronous serial devices.
- Choose whether PRINTER: is to be treated as a serial or parallel device.
- Set the standard printer parameters (print type, lines per page, and so forth).
Configure Utility

- Set the horizontal synchronization point for a television set (if one is being used as the video display device).
- Set various parameters for user-defined drivers.
- Manage the device drivers library file SYSTEM.CONFIG that contains both device drivers and their parameters.

Note that the file SYSTEM.CONFIG, on the boot volume, is used to contain the drivers and parameters for the various peripheral devices. This file (which can be altered by the Configure utility) must be present on the boot volume.
Main Configure Menu

The Configure utility is found in CONFIGURE.CODE. When you execute this code file, the following main menu appears:

SYSTEM CONFIGURATION UTILITY [ ]

A) Change Driver Parameters
B) Change Operating System Parameters
C) Install Device Drivers
D) EPSON Matrix Printer Utility
Q) Quit. Exit Configuration Utility

Enter letter of option (A..Q): _

Option A, Change Driver Parameters, allows you to change the parameters of every device driver in memory or in a SYSTEM.CONFIG file.

Option B, Change Operating System Parameters, allows you to alter some of the operating system parameters in SYSTEM.MISCINFO, such as HAS EXTENDED MEMORY and MAX NUMBER OF SUBSIDIARY VOLS. (These parameters can be changed, as well, by the Setup utility. They are included in Configure for your convenience since they are the most commonly changed operating system parameters.)
Configure Utility

Option C. Install Device Drivers, allows you to install new device drivers into a SYSTEM.CONFIG file, or to build a new SYSTEM.CONFIG from scratch.

Option D. Matrix Printer Utility, allows you to configure the EPSON dot-matrix printer parameters, such as characters per line, lines per inch, tab stops, and so on.

Option Q. Quit, exits the Configure utility.

Changing Driver Parameters

The Change Drivers option of the main Configure menu allows you to access and change the parameters for your peripheral devices. These parameters are stored on disk in SYSTEM.CONFIG and are read into memory at boot time. You are first asked to select whether Configure should access parameters from memory or from disk:

Read driver parameters information from:

A) Memory
B) Disk

Q) Quit, return to main menu

Enter letter of option (A..Q): -
Configure Utility

If you select Memory by pressing A, the current configuration information will be accessed. Normally, this is the configuration that was loaded when the p—System was booted; but Configure may have changed it since booting.

If you select Disk by pressing B, you can access and alter the device parameters stored in a SYSTEM.CONFIG file on a disk. This does not necessarily have to be the SYSTEM.CONFIG file on the boot disk (from which the parameters in memory were obtained at boot time). In addition to updating the disk; if you wish, you can update memory with the altered parameters.

When you select Disk, the following prompt is displayed:

Read SYSTEM.CONFIG from which device (4, 5, 9..#) : 

Now, enter the appropriate number followed by <enter>; don't include the preceding number sign or trailing colon. (The # in the prompt indicates the highest device number allocated to storage volumes.) Your response should be the device number of the volume that contains the SYSTEM.CONFIG file you want to access. You could change the parameters for several different boot disks by selecting disks other than the current system disk.
Configure Utility

When you are reading parameters from disk, you are also prompted for a SYSTEM.MISCINFO file as follows:

SYSTEM.MISCINFO is required for device number information
Read SYSTEM.MISCINFO from which device (4, 5, 9..##)

Again, you should enter the appropriate number, followed by <enter>. Your response should indicate the volume that contains the SYSTEM.MISCINFO file you plan to use with the SYSTEM.CONFIG file you are altering. For example, if you selected device 4 as the disk that contains SYSTEM.CONFIG—and that disk is a boot disk containing the corresponding SYSTEM.MISCINFO—you should respond to this prompt by indicating device 4.

After the Configure utility has read in the information from either memory or disk, it displays a menu. The following menu is displayed when the information is read from disk (the memory menu is essentially identical):

SYSTEM CONFIGURATION UTILITY [ ]
Change Driver Parameters from Disk

A) Console Driver
B) Printer Driver
C) Disk Driver
D) Remote Driver
E) Serial Driver
F) User-Defined Driver
G) Quit: Return to main menu

Enter letter of option (A..G): _

The following paragraphs describe each of the options in this menu.
Configure Utility

Console Driver

Option A of the Change Driver Parameters menu allows you to change the parameters of the console driver. The menu below is a typical example of what this option displays. This menu indicates that parameters are being accessed from disk.

```
SYSTEM CONFIGURATION UTILITY [ ]
Change Driver Parameters from Disk

Console Driver Parameters - Device 1

A) Driver = CONSOLEDRV
B) Display Type = Graphics Adapter
C) 40 Column Mode = Sliding 80
D) Foreground Color = White
E) Background Color = Blue
F) Border Color = Blue
G) Video Page = 3
H) Video Mode = 7
I) Horizontal Sync = 0

Q) Quit. Return to main menu

Enter letter of option (A..Q): _
```

The Driver option allows you to view the name of the console driver in the current SYSTEM.CONFIG file that you are accessing. (This option does not appear if you are accessing parameters in memory.)
Configure Utility

The Display Type option allows you to select one of the following two types as the main display device:

Monochrome Adapter Display
Graphics Adapter Display

If both monochrome and graphics cards are installed, this option allows you to select either one of them. At boot time, an equipment check is made. If the configuration specified in SYSTEM.CONFIG is not consistent with the current hardware, this parameter is ignored, and the installed hardware is enabled.

The 40 Column Mode option allows you to select one of the following two types of display windows:

Sliding 80
Fixed 40

The 40 Column Mode is only useful if you are in video mode 0 or 1 (described ahead). In video mode 0 or 1, you can only view 40 characters per line. With option Fixed 40, these characters are always the 40 in the line. With option Sliding 80, you can view any sequence of 40 characters in the first 80.

The next three options are Foreground Color, Background Color, and Border Color. These options allow you to select the corresponding colors for the video display. The color specifications are ignored if the monochrome adapter card is used.
Configure Utility

If Foreground or Border color is selected, you can select from the following list of 16 colors:

A) Black
B) Blue
C) Green
D) Cyan
E) Red
F) Magenta
G) Light Gray
H) Dark Gray
I) Light Blue
J) Light Green
K) Light Cyan
L) Light Red
M) Light Magenta
N) White

If Background color is selected, you may select from only the first eight of the 16 colors.

The Video Page option allows you to set the video display page. Please note that choosing a video display page has no important effect with the current IBM PC implementation of the p-System. Depending upon the "video mode" (described next) which you are using, you may choose from among several video display pages. When you select the Video Page option, you are prompted for a number. If you are using video modes 0 or 1 (described ahead), then you may choose display pages 0 through 7. If you are using video modes 2 or 3 (also described ahead), you may choose display pages 0 through 3.
Configure Utility

The Video Mode Option allows you to set the video display mode. When you select this option, you are prompted to enter a number. This number should be from 0 to 7, corresponding to the following modes:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Columns</th>
<th>Color</th>
<th>Graphics</th>
<th>Pixels</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>40 x 26</td>
<td>No</td>
<td>No</td>
<td>--</td>
</tr>
<tr>
<td>1</td>
<td>40 x 26</td>
<td>Yes</td>
<td>No</td>
<td>--</td>
</tr>
<tr>
<td>2</td>
<td>80 x 25</td>
<td>No</td>
<td>No</td>
<td>--</td>
</tr>
<tr>
<td>3</td>
<td>80 x 25</td>
<td>Yes</td>
<td>No</td>
<td>--</td>
</tr>
<tr>
<td>4</td>
<td>40 x 25</td>
<td>Yes</td>
<td>Yes</td>
<td>320 x 200</td>
</tr>
<tr>
<td>5</td>
<td>80 x 25</td>
<td>No</td>
<td>Yes</td>
<td>320 x 200</td>
</tr>
<tr>
<td>6</td>
<td>80 x 25</td>
<td>No</td>
<td>Yes</td>
<td>640 x 200</td>
</tr>
<tr>
<td>7</td>
<td>Black and White Card</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The Horizontal Sync option allows you to adjust the position of the left column of your display. Choosing the Horizontal Sync option yields this message:

press Esc key to begin adjustment

Pressing `<esc>` yields the following display:

```
123456789012345678901234567890123456789012345678901234567890
```

Use arrow keys to align the ruler to the left margin of the screen
Current value in memory is: 0
Press Esc key when done

Typing the `<left-arrow>` or `<right-arrow>` keys causes the left margin of your display to be adjusted accordingly. The numbers on the top of the screen indicate what your screen will look like under the present changes.
Configure Utility

Printer Driver

Option B of the Change Driver Parameters menu allows you to designate the printer as being a serial or parallel device. When you select this option, the following menu is displayed:

```
SYSTEM CONFIGURATION UTILITY [ ]
Change Driver Parameters from Disk

Printer Parameters - Device 6
A) Driver       = SERPAR.DRV
B) Active Port  = Parallel 0
Q) Quit. Return to main menu

Enter letter of option (A..Q): _
```

The Driver option allows you to view the name of the printer driver in the current SYSTEM.CONFIG file which you are accessing. (This option doesn’t appear if you are accessing parameters in memory.)

The second option, Active Port, allows you to configure the software so it will interact with a printer connected to any of the three parallel ports or two serial ports. The p—System PRINTER: device, which is device 6, then takes on the characteristics of the specified port. The following options menu is displayed:

```
A) Parallel Port 0
B) Parallel Port 1
C) Parallel Port 2
D) Serial Port 0
E) Serial Port 1
```
Configure Utility

If a serial port is selected, the Printer Parameters menu will look like the Remote Parameters menu (described later). This means that in addition to the Driver and Active Port parameters, you can also change the Baud Rate, Parity, Word Length, Stop Bits, Protocol, and ETX Frequency parameters.

Disk Driver

Option C of the Change Driver Parameters menu allows you to alter disk driver parameters. This includes floppy disk drivers, fixed disk drivers, and RAM disk drivers. If you want to configure the fixed disk, you should use the Fdisk utility program.

The Disk Driver option first asks you to select the driver to be reconfigured:

Enter disk device number to configure (4, 5, 9..0F) : –

You should enter the appropriate device number followed by <enter>. Depending upon whether you have indicated a floppy disk, a fixed disk, or a RAM disk, you will receive the appropriate menu.
Configure Utility

Floppy Driver

If you select a floppy disk device, the following menu is displayed (it is assumed that 4 is the corresponding device number):

Floppy Disk Parameters

SYSTEM CONFIGURATION UTILITY [..]
Change Driver Parameters from Disk

Disk Driver Parameters - Device 4

A) Driver                = FLOPPY.DRV
B) Sectors per Track    = 8
C) Tracks per Side      = 40
D) Number of Sides      = 2
E) Bytes per Sector     = 512
F) Step Rate            = 6 ms
G) Write Verify          = NO
H) Cylinder Access      = NO
I) Interleave           = 1
J) Skew                 = 0
K) First p-System Track = 0
Q) Quit. Return to main menu

Enter letter of option (A..Q): _

This option allows you to configure the disk driver, which enables you to read/write p-System disks. The following information relates only to p-System disks and not to DOS disks containing p-System virtual volumes.
Configure Utility

The Driver option allows you to view the name of the disk driver, for the selected device number, in the current SYSTEM.CONFIG file you are accessing. (This option doesn't appear if you are accessing parameters in memory.)

The Sectors per Track option indicates the number of sectors on a diskette track. Normally, this value should be 8 or 9 sectors per track. A value of 89 selects a mode where either 8 or 9 sectors per track format is recognized. With the 8 sector mode, a single-sided diskette may hold 320 blocks; and a double-sided diskette may hold 640 blocks. With the 9 sector mode, a single-sided diskette may hold 380 blocks; and a double-sided diskette may hold 720 blocks. (These values assume that you are using 40 tracks per side and 512 bytes per sector.) Previous versions of the p-System on the IBM Personal Computer used the 8 sector per track format. The Universal Medium format uses the 8 sector per track format.

If the Sectors per Track value is other than 8, 9, 10, or 89, the system tries to read a diskette with the indicated number of sectors; but it may not succeed. Other values are usually used for Bytes per Sector values other than 512. For example, you can configure a diskette drive to read a 256 bytes per sector, 16 sectors per track diskette, but it may not work correctly.

The Tracks per Side option may be 35, 40, or 80. The Universal Medium disk format uses a value of 40 tracks per side. If you have a 96 TPI disk drive, you can use a value of 80.
Configure Utility

The Number of Sides option may be 1 or 2. You should select 2, if this is a double-sided disk drive; and if it is a single-sided disk drive, you should select 1.

The Bytes per Sector option may be 128, 256, or 512. The Universal Medium disk format uses 512 bytes per sector.

The Step Rate option may be 2ms, 4ms, 6ms, or 8ms. The suggested step rate for standard IBM Personal Computer disk drives is 6ms. A value of 8 is the slowest and most reliable. A value of 2 is the fastest and most error-prone.

The Write Verify option may be set to Yes or No. If Write Verify is used, the data on a disk will be checked after every write. (Using verified writes is slower, but safer.)

The Cylinder Access option may be Yes or No. This option allows a disk to be accessed in a cylinder mode. (In cylinder mode, data is recorded as follows: on the first track of the top side of a disk; on the first track on the bottom side; on the second track on the top side; and so on. When the non-cylinder mode is used, the entire top side of a disk is used, and then the entire bottom side is used.) Cylinder access is not consistent with the Universal Medium format or with the format used by earlier versions of the p-System on the IBM Personal Computer.
Configure Utility

The Interleave option allows you to set the ratio of sector interleave on a track. For the Universal Medium, this interleave is 1 (which means 1 to 1—no sectors are skipped).

The Skew option allows you to set the track-to-track sector offset. For the Universal Medium, this is set to 0.

The First p-System Track option allows you to set the track number where p-System volume begins. For the Universal Medium, this is set to 0.

RAM Disk Driver

If you select RAM disk (rather than a floppy disk), the following menu is displayed:

RAM Disk Parameters

SYSTEM CONFIGURATION UTILITY [ ]
Change Driver Parameters from Disk

-------------------------------
Disk Driver Parameters - Device 9

A) Driver = RAMDISK.DRV
B) Base Address (K Bytes) = 130
C) Size (in blocks) = 380

Q) Quit. Return to main menu

Note: Changes made to RAM Disk must be saved with Disk update and the p-System rebooted for the changes to take effect.

Enter letter of option (A..Q): _
Configure Utility

The Driver option allows you to view the name of the RAM disk driver in the current SYSTEM.CONFIG file that you are accessing. (This option doesn’t appear if you are accessing parameters in memory.)

The Base Address option allows you to specify the starting address of the RAM disk in 1024-byte (K-byte) quantities. A value of 0 or a value lower than the address of free memory gives the RAM disk base a default value of the first address of free memory.

The Size option allows you to specify the size of the RAM disk in blocks. (A block is 512 bytes.) If you enter a value of 0 or a value larger than the amount of memory available, all of the available free memory will be allocated to the RAM disk.

User—Defined Disk Drivers

If you are using a disk driver supplied by another manufacturer, you can use the User—Defined Driver option (rather than the Disk Driver option) from the Change Driver Parameters menu. The User—Defined Driver option is discussed later. It’s not as straightforward to use, since you need to know the parameter offsets and the proper hexadecimal values to be placed into those offsets. (This is because of the generality which is possible with user—defined driver formats.)
Remote Driver

Option D of the Change Driver Parameters menu allows you to select one of two serial ports and the parameters for each port. The following menu is displayed:

```
SYSTEM CONFIGURATION UTILITY [ ]
Change Driver Parameters from Disk

Remote Driver Parameters - Device 7,8
A) Driver = SERPAR.DRV
B) Active Port = Serial 0
C) Baud Rate = 9600
D) Parity = None
E) Word Length = 7
F) Stop Bits = 1
G) Protocol = DTR/CTS
H) ETX Frequency = 25

Q) Quit. Return to main menu

Enter letter of option (A..Q): -
```

The Driver option allows you to view the name of the remote device driver in the current SYSTEM.CONFIG file that you are accessing. (This option doesn’t appear if you are accessing parameters in memory.)
Configure Utility

The Active Port option allows you to configure the software so you can attach the remote device to either one of the 2 asynchronous serial ports. The p-System remote device (called REMIN: and REMOUT;—depending upon the direction of the data flow) then takes on the characteristics of the specified port. When you select Active Port, the value toggles between:

Serial 0
Serial 1

Once you have selected the active port, the other options allow you to set the parameters associated with it. These are described as follows.

The Baud Rate option allows the following selections:

A) 110  
B) 180  
C) 300  
D) 600  
E) 1200 
F) 2400 
G) 4800 
H) 9600

The Parity option allows the following selections:

A) None  
B) Odd parity  
C) Even parity

The Word Length option toggles between:

7 bits
8 bits
Configure Utility

The Stop Bits option toggles between:

1
2

The Flow Control Protocol allows these selections:

A) DTR/CTS
B) ETX/ACK
C) Xon/Xoff

If ETX/ACK protocol is selected, you are also allowed to select the parameter for ETX frequency. This parameter indicates the number of characters sent to the printer before an ETX is sent. The ETX frequency is only valid for ETX/ACK protocol. Its value is ignored for other protocols.

Serial Driver

If you have any user-defined serial devices, you can configure them using this option. It displays the same menus as the Remote Device option.

User-Defined Driver

The User-Defined Driver option allows you to set the parameters for user-written or non-standard devices. Because of the generality which is possible in non-standard devices, the Configure utility cannot interpret nor display the meaning of each parameter. This option merely provides a general mechanism with which a user-written device driver can be configured.
Configure Utility

You are first prompted for the device number of the device you want to configure:

Enter device number to configure (0..255) : _

The device number you enter is not checked for validity. The following menu is displayed.

SYSTEM CONFIGURATION UTILITY [ ]
Change Driver Parameters from Disk

User-Defined Driver Parameters - Device ##

A) Driver = CUSTOM.DRV
B) Parameter 1 = 0
C) Parameter 2 = 0
D) Parameter 3 = 0
E) Parameter 4 = 0
F) Parameter 5 = 0
G) Parameter 6 = 0
H) Parameter 7 = 0
I) Parameter 8 = 0
J) Parameter 9 = 0
K) Parameter 10 = 0
L) Parameter 11 = 0
M) Parameter 12 = 0

Q) Quit, Return to main menu

Enter letter of option (A..Q): _

The Driver option allows you to view the name of the user-defined driver, for the specific device number, in the current SYSTEM.CONFIG file that you are accessing. (This option doesn’t appear if you are accessing parameters in memory.)
Configure Utility

Other options allow you to enter a value for each of the 12 parameters used by the driver of the current device. Enter these numbers in decimal. (Their meaning depends upon the driver being used.)

Change OS Parameters

The Change Operating System Parameters option of the main Configure menu enables you to alter certain operating system values contained in SYSTEM.MISCINFO. (The Setup utility, described in the Operating System Reference Manual, allows you to alter all of the values in SYSTEM.MISCINFO, including the ones affected here. For convenience, the Configure utility offers you another way to change some of the more commonly altered operating system values.)

The Change Operating System Parameters option first asks you for the location of the desired SYSTEM.MISCINFO file, as follows:

Read SYSTEM.MISCINFO from which device (4, 5, 6...):  

You should enter a number followed by <enter>. This number should be the device number of a volume containing the SYSTEM.MISCINFO you want to alter.
The information is then read in; and the following menu—with typical values—is displayed:

**SYSTEM CONFIGURATION UTILITY [ ]**
Change Operating System Parameters

----------------------------------------
A) Has Extended Memory = YES
B) Code Pool Size (84K bytes max) = 0
C) Print Spooler Enabled = NO
D) First Subsidiary Vol Number = 20
E) Max Number of Subsidiary Vol = 10
F) Max Number of Serial Volumes = 1
G) Quit. Return to main menu

Note: Changes will take effect only after reboot

Enter letter of option (A..G): 

HAS EXTENDED MEMORY allows you to enable or disable the extended memory feature of the p-System. If the value is set to YES, the p-System automatically uses the extended memory feature, if there is enough memory to support it (generally at least 128K). The p-System won't use the extended memory feature, if the value is set to NO—regardless of how much memory you have in your machine.
Configure Utility

The CODE POOL SIZE parameter indicates the size of the external code pool. When this parameter is set to 0—and HAS EXTENDED MEMORY is set to YES—the p-System automatically calculates the optimum code pool size. The p-System allocates 64K for data (stack and heap space) and allocates as much available memory as possible (up to 64K) for the code pool. If there isn’t enough memory for a reasonable size code pool, the p-System configures itself without extended memory.

If you specify a non-zero value for CODE POOL SIZE, the p-System allocates that size for the code pool. On a system with more than 128K of memory, 64K is allocated for the data area (stack and heap). On a system with 128K or less, the memory allocated for the data area may be less than 64K to allow for the specified code pool size. (The p-System uses about 24K of memory in addition to the code pool and data areas.)

WARNING: Be careful when specifying a non-zero value for CODE POOL SIZE. If the value specified is too small (less than about 24K) or too large (leaving insufficient memory for a reasonable data area), the p-System may not boot. It’s best to set HAS EXTENDED MEMORY to YES and CODE POOL SIZE to zero, and let the p-System configure itself. (The code pool is explained further in the Internal Architecture Reference Manual.)

If HAS EXTENDED MEMORY is set to NO, the p-System ignores the CODE POOL SIZE parameter.
PRINT SPOOLER ENABLED allows you to turn the print spooling feature on or off. If you want to use the Print Spooler, be sure that the SPOOLOPS unit is placed in SYSTEM.PASCAL—using the Library utility—and that this field is to YES.

**WARNING:** If PRINT SPOOLER ENABLED is set to YES and the SPOOLOPS unit is not in SYSTEM.PASCAL, the p-System won’t boot.

FIRST SUBSIDIARY VOLUME number should be set to the lowest device number available for subsidiary volumes. This number must always be 9 or greater. If you have two floppy disks (4 and 5), six virtual volumes (9 through 14), and a RAM disk (15); then this number should be 16. If you also have a fixed disk (which requires four more device numbers, as described under the Fdisk utility), this number should be 20. There are other possibilities, depending upon your hardware configuration.

**WARNING:** If the FIRST SUBSIDIARY VOLUME number is set lower than the first virtual volume, the p-System won’t boot.

MAX NUMBER OF SUBSIDIARY VOL indicates the greatest number of subsidiary volumes that may be mounted at one time. You should set this to the smallest convenient number. For example, it might be set to 10.
Configure Utility

MAX NUMBER OF SERIAL VOLUMES indicates the maximum number of user-defined serial volumes allowed to be on line at one time. If you have any user-defined serial volumes, you should set this to the number of them you have. The largest value allowed is 26.

Install Device Drivers

The Install Device Drivers option from the main Configure menu allows you to install new device drivers into an existing SYSTEM.CONFIG or to build a new SYSTEM.CONFIG.

The Configure utility first prompts you for an output file name, an input file name, and the corresponding SYSTEM.MISCINFO file device number, as follows:

```
SYSTEM CONFIGURATION utility [ ]
Install Device Drivers to SYSTEM.CONFIG

---------------------------------------------------------------------

Configuration Output file name? FILE-NAME (enter)
Configuration Input file name? FILE-NAME (enter)
SYSTEM.MISCINFO is required for device number information
Read SYSTEM.MISCINFO from which device (4..##): # (enter)
```

As an example, you might respond with *SYSTEM.CONFIG for the input file, *NEW.CONFIG for the output file, and 4 for the device number of SYSTEM.MISCINFO.
Configure Utility

If no input file name is given (that is, if you simply press <enter> in response to the prompt), Configure creates the output file from scratch; and the following prompt appears:

**SYSTEM CONFIGURATION UTILITY [ ]**
*Install Device Drivers to SYSTEM.CONFIG*

```
Standard p-System device numbers are:
1       for Console device
4-6     for Disk devices
6       for Printer device
7-8     for Remote device
9-20    for additional Disks
31-33   for Serial devices

Enter device number to install driver (0-255): _
```

Here you insert the number for the device you wish to add. For a blocked device (such as a disk), you may install a driver at device number 4, 5, 9..X, where X is FIRST SUBSIDIARY VOLUME, minus 1 (20 in the example just shown). FIRST SUBSIDIARY VOLUME is defined in the SYSTEM.MISCINFO file and may be set either in the Configure utility or by using the Setup utility.
Configure Utility

To determine where you may install a serial device, you must find the FIRST SUBSIDARY VOL NUMBER and add to it the MAX NUMBER OF SUBSIDARY VOLUMES, as defined in SYSTEM.MISCINFO. This number is the device number where you may install the first serial device. In the previous example, this number is 31; and the value for MAX NUMBER OF SERIAL DEVICES in SYSTEM.MISCINFO is 3, so device 33 is your last serial device.

NOTE: The maximum number of devices that can be active at any one time is 32. This means that you can have 32 different drivers: or, for example, just 4 drivers, each of which may be used by one or more of 32 different devices.

The subsidiary volumes in the above configuration are located at devices 21 through 30. Remember that the above values are only examples, and that actual values depend on the values in the current SYSTEM.MISCINFO.
If there is an input file name given for the install device drivers option, the current contents of the input file are displayed as in the following example:

<table>
<thead>
<tr>
<th>Device</th>
<th>Driver</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td># 1 Console</td>
<td>CONSOL.DRV</td>
<td>1 2 16 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td># 5 Printer</td>
<td>SERPAR.DRV</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td># 4 Disk 0</td>
<td>FLOPPY.DRV</td>
<td>0 0 1 0 0 2 8 40 1 0 0 0</td>
</tr>
<tr>
<td># 8 Disk 1</td>
<td>FLOPPY.DRV</td>
<td>0 1 1 0 0 2 8 40 1 0 0 0</td>
</tr>
<tr>
<td># 9 Disk 2</td>
<td>RAMDISK.DRV</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>#10 Disk 3</td>
<td>FDISK.DRV</td>
<td>2 0 10 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>#11 Disk 4</td>
<td>FDISK.DRV</td>
<td>2 0 10 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>#12 Disk 5</td>
<td>FDISK.DRV</td>
<td>2 0 10 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>#13 Disk 6</td>
<td>FDISK.DRV</td>
<td>2 0 10 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>#7 Remote 0</td>
<td>SERPAR.DRV</td>
<td>1 129 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>#0 System 0</td>
<td>SYSTEM.DRV</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
</tbody>
</table>

In this display, the device numbers are associated with the device name and the "device index" in the first column. (In general, you won't have to concern yourself with the device index.) The second column shows the corresponding driver name. (The driver name was taken from the name of the driver code file at the time it was installed into SYSTEM.CONFIG.) The third column displays the current parameters for each device in decimal notation. (These parameters are displayed more meaningfully in the Change Driver Parameters option of Configure, as described earlier.)

Below this display, the following prompt appears:

Enter device number to install driver (0..255): ## (enter)
Configure Utility

If you want to install a device driver, enter the corresponding device number. You can press <esc> to exit from this prompt without installing a new driver. For example, if you enter 1 in response to this prompt, the following sample display appears:

```
SYSTEM CONFIGURATION UTILITY [ ]
Install Device Drivers
=====================================================================
Console Device 1

A) Driver = NEW_DRV
B) Parameter 1 = 0
C) Parameter 2 = 0
D) Parameter 3 = 0
E) Parameter 4 = 0
F) Parameter 5 = 0
G) Parameter 6 = 0
H) Parameter 7 = 0
I) Parameter 8 = 0
J) Parameter 9 = 0
K) Parameter 10 = 0
L) Parameter 11 = 0
M) Parameter 12 = 0

Q) Quit. Do not install device driver.
```

Enter letter of option (A...Q):

By selecting A, this menu allows you to enter the name of a driver code file to be inserted into the output file. (The example shows NEW_DRV as the code file name. Specify the entire file name, including the .CODE suffix if applicable.)
Configure Utility

This prompt also enables you to enter 12 bytes of general parameters. Select the appropriate option and enter the numbers in decimal. Alternatively, you can simply install the driver; and then set the parameters with the Change Driver Parameters option—making it much easier to understand what the parameters stand for in standard devices. A default value of 0 is given for any parameter you don’t specifically assign to a value. (The driver determines the meaning of the parameters.)

For the Change Driver Parameters option to recognize a specific device, the first byte of the general parameters must be set as described in Chapter 7, "Writing A Device Driver."

If you press Q to quit, the driver and the parameters are installed into the configuration output file. If, instead, you press <esc>, this level of the Configure utility is exited without installing any drivers. When you select Quit, you are asked if you want to install another driver:

```
Device Driver added to SYSTEM.CONFIG
Install another device driver (Y or N); Y or N
```

If you respond with Y, the previous menus are repeated. If you respond with N, the configuration output file is closed; and you are returned to the main Configure menu.
Configure Utility

The file name of the driver being installed determines how that driver is treated. If a driver's file name is the same as an existing driver's file name was when it was installed, Configure assumes they are the same driver. If the driver's file name is different, Configure assumes that it is different from any existing drivers. The implications of this are discussed in the following paragraphs.

When you install a driver with a new file name (that is, no driver in the output configuration file has this file name), and the corresponding device is not currently assigned to any driver, you are simply adding a new driver.

When you install a driver with a new file name, and the corresponding device is already assigned to a driver, the old driver is replaced with the new driver. However, if any other devices were also using the old driver, they will continue to use that old driver. The old driver will be thrown away, if the device you specify was the only device using it; otherwise, it is retained.

The existing copy of the driver in the output configuration file is assigned to the new device number. This is done when you install a driver with an existing file name (that is, a name that at least one device in the output configuration file is currently assigned to), but when the corresponding device is not currently assigned to any driver. This means that no additional code is actually placed into the output configuration file. In the following example you would want to Add a device to the existing driver.
Configure Utility

When you replace a driver that has the same file name as the existing driver for the device, the new driver replaces the existing driver. (It is assumed that you are adding a modified version of the same driver.) In this case, any other devices which had also used the old driver will now use the new driver. (The old driver is always thrown away.) This last possibility sometimes results in replacing drivers for devices other than the one you are specifically replacing. In this situation, you are asked to verify the action as shown in the following example:

```
Driver already exists for device(s) $10, $11, $12, $13
Replace driver, Add device to driver Quit (R or A or Q)? _
```

If you enter _R_, the new driver will replace the existing driver for all of the indicated devices. By entering _A_, the current device will now use the existing driver it is being assigned to. If you press _Q_, the new driver will not be installed at all; and the program will return to the next highest menu level. If you want to install the new driver for only one of the devices that currently uses the same driver (or for a subset of these devices), do the following. Exit the Configure utility and use the filer to change the name of the driver's code file to a unique name.

If the device you are installing is a disk device, you are also asked how many new devices are going to use this driver:

```
Enter number of devices using this driver (0..16): #<enter>
```
Configure Utility

If you are simply installing a driver for one disk device, enter 1. However, you might want to assign the new disk driver to several new disk devices. Using this feature, a maximum of 16 devices can be assigned to a driver at one time. As an example, you might be adding four p-System partitions from a fixed disk. In this case, enter 4. (This is just a convenience since, otherwise, you would have to specifically enter the driver four times.)

When you assign multiple device numbers to one driver, the driver is assigned to the current device number and to as many additional contiguous device numbers as necessary. For example, if you are installing device #10 and you want a total of 4 devices to use this driver, devices #10, #11, #12, and #13 will be assigned. (However, disk devices are not assigned to numbers 6, 7, and 8. For example, if you were starting at #5 with 4 devices, the devices would be: #5, #9, #10, and #11.)

**WARNING:** If you use this facility to install a disk driver for more than one device at a time, be sure that the additional device numbers are currently unallocated. Any existing drivers assigned to these device numbers will be replaced without a warning.
NOTE: Whenever you install a new disk driver (regardless of how many device numbers are being assigned to it), make sure that the resulting highest numbered disk device is less than the FIRST SUBSIDIARY VOL NUMBER field. This field indicates the first device number set aside for subsidiary volumes and can be set with either the Configure utility or the Setup utility. If there is a conflict between device numbers assigned to disk drivers and the device numbers assigned to subsidiary volumes, you can’t access the disk drivers until the FIRST SUBSIDIARY VOL NUMBER field is properly increased.

Matrix Printer

The Matrix Printer option from the main Configure menu allows you to set various options for the EPSON Matrix Printer. Note that this applies only to EPSON printers and not to printers from other manufacturers. The following menu is displayed:

```
SYSTEM CONFIGURATION UTILITY [ ]
EPSON Matrix Printer Configuration Utility

A) Chars per Line
B) Lines per Inch
C) Form Length
D) Horizontal Tabs
E) Vertical Tabs

Q) Quit. Return to main menu

Enter letter of option (A...Q): _
```

The Chars per Line option may be set to 80 or 132. This is the maximum number of characters that will be printed on a single line.
Configure Utility

The Lines per Inch option may be set to 6, 8, or 10. This determines the vertical spacing of the printed lines.

The Form Length option may be set to any value from 1 to 127. This determines the number of lines on a page.

The Horizontal Tabs option allows you to set the positions for tab stops within a line.

The Vertical Tabs option allows you to set the vertical tab stops for a printed page.

The following example explains how to install tab positions:

Enter Tab position list: 4,10,28,56,84,112

The above entry would place the first horizontal (vertical) stop at 4 spaces (lines) from the left (top), the next at 10 spaces (lines) from the left (top), and so on.
CHAPTER 5
M - S DOS ACCESS
AND CONFIGURATION
UNITS
GENERAL

The units described in this chapter provide access to DOS and to the p—System configuration.

The FILEIO, FILERUNIT, and DOSDIRINFO units offer the same facilities as those supplied in the "XenoFile for PC DOS" product. FILEIO provides file level access to DOS files and such features as setting the DOS default directory and path. FILERUNIT affords file transfer facilities between DOS and the p—System; and DOSDIRINFO provides information about DOS directories.

The CALLDOS unit gives Pascal programs access to the DOS functions, and the CLOCK unit provides access to the DOS time of day features.

The VIRTVOl unit accesses and changes the virtual volume configuration, which primarily involves controlling what DOS file is associated with each p—System virtual volume device.

The driver configuration unit, DRVCONFIG, changes the p—System device driver configuration file, SYSTEM.CONFIG.
Access/Configuration Units

FILEIO

The FILEIO unit provides seven procedures and three functions. Each of these routines is described in detail. Almost all of these routines require a file variable to be declared (type 'dos_file') and used for every access to a particular file. This variable must be used with a dos_reset or dos_rewrite before being used with any other routine; this allows the variable to be initialized.

Below is the interface section of the FILEIO unit, followed by descriptions of the routines available to the programmer:

UNIT FILEIO;
INTERFACE
TYPE
  dos_file = array[0..49] of integer;
  dos_name = string;
  dos_bik = packed array[0..811] of char;
  dos_e_modes = (dos_normal, dos_lock, dos_burge);

PROCEDURE dos_reset(var fideos_file; name:string);
PROCEDURE dos_rename(var fideos_file);
FUNCTION dos_rewrite(var fideos_file; name:string);
FUNCTION dos_bik_reed(var fideos_file; var block;dos_bik);
FUNCTION dos_bik_write(var fideos_file; var block;dos_bik); 
FUNCTION dos_close(var fideos_file; mode;dos_e_modes);
PROCEDURE dos_rename(var fideos_file; name:string);
PROCEDURE dos_flush;
PROCEDURE dos_prefix(name:string; var prefix:string);
FUNCTION dos_seen_title(name:string;
  var sel, path, title, option:char); bolean;

IMPLEMENTATION
**dos_reset**

This procedure opens an existing DOS file. A file variable is passed as the first parameter, and the DOS file name is the second parameter. The IORESULT should be checked after the call and will be 0 if the reset was successful; otherwise, it will indicate the cause of the error. (See Appendix A for IORESULT codes.)

**dos_rewind**

This procedure repositions back to the beginning of a DOS file. A block read or write after this call will start accessing with block 0. The file variable is passed as the only parameter. The IORESULT should be checked after the call and will be 0 if the rewind was successful; otherwise, it will indicate the cause of the error. (See Appendix A for IORESULT codes.)

**dos_rewrite**

This procedure creates a new DOS file. The file variable is passed as the first parameter, and the DOS file name is the second parameter. The IORESULT should be checked after the call and will be 0 if the rewrite was successful; otherwise, it will indicate the cause of the error. (See Appendix A for IORESULT codes.)
dos_blk_read

This function reads one or more blocks from a DOS file. A block consists of 512 bytes or one 512-byte disk sector.

The first parameter is the file variable, and the second parameter is the buffer which the block(s) will be read into. More than one block may be read into a buffer by declaring an array of blocks and by specifying more than one block to read. (See the next example.) The third parameter is the number of blocks to be read from the file. The fourth parameter is the block number of the starting block to be read.

The first block of a file is block number 0. If the starting block specified is −1, then the next block in the file will be read. The function result is the actual number of blocks read. The IORESULT should be checked after the function call and will be 0 if the read was successful; otherwise, it will indicate the cause of the error. (See Appendix A for IORESULT codes.)

The following Pascal program fragment shows how multiple blocks can be read:

```
VAR  
  f:dos_file;  
  n_read:integer;  
  {number of blocks read}  
  buffer:array[1..10] of dos_blk;  
BEGIN  
  { assume the file is already open }  
  n_read:=dos_blk_read( f, buffer[1], 10, 0 );  
  { up to ten blocks have been read into buffer }  
END;  
```
dos_blk_write

This function writes one or more blocks to a DOS file. A block consists of 512 bytes or one 512-byte disk sector.

The first parameter is the file variable. The second parameter is the buffer that the block(s) will be written from. More than one block may be written from a buffer by declaring an array of blocks and by specifying more than one block to write (see the previous example). The third parameter is the number of blocks to be written to the file. The fourth parameter is the block number of the starting block to be written.

The first block of a file is block number 0. If the starting block specified is -1, then the next block in the file will be written. The function result is the actual number of blocks written. The IORESULT should be checked after the function call and will be 0 if the write was successful; otherwise, it will indicate the cause of the error. (See Appendix A for IORESULT codes.)

dos_close

This procedure closes a file that has been opened with a dos_reset or dos_rewrite. The first parameter is the file variable. The second parameter is the type of close requested.

Close with dos_purge will remove the file from the directory. Close with dos_lock is used to close a new file and put it permanently in the directory. Close with dos_normal is used to keep a file opened with dos_reset and remove a file opened with dos_rewrite.

All files must be closed after you have finished using them. The IORESULT should be checked after the procedure call and will be 0 if the close was successful; otherwise, it will
Access/Configuration Units

indicate the cause of the error. (See Appendix A for IORESULT codes.)

dos rename

This procedure changes the name of a DOS file. The file must have been opened using reset. This may not be used after opening with dos rewrite.

The first parameter is the file variable. The second parameter is the name you want the file to have. The IORESULT should be checked after the procedure call and will be 0 if the rename was successful; otherwise, it will indicate the cause of the error. (See Appendix A for IORESULT codes.)

dos flush

This procedure causes all buffered data to be written out to the disk. It should be called if a disk is going to be removed from the drive. There are no parameters to this procedure. The IORESULT should be checked after the procedure call and will be 0 if the flush was successful; otherwise, it will indicate the cause of the error. (See Appendix A for IORESULT codes.)

dos prefix

This procedure sets the default drive and path. The path need not be given. The first parameter is the name of the default drive and path (optional) to become the default. The second parameter returns the default drive and path. The IORESULT is not set by this procedure and should not be checked.
dos_scan_title

This function breaks a file name into its individual parts. The first parameter is the file name. The second parameter returns the volume name, the third parameter returns the path (if any), the fourth parameter returns the title, and the fifth parameter returns the options (if any). The function result is true, if a valid file name was passed in the first parameter; and false, if the file name was invalid. The IORESULT is not set by this procedure and should not be checked.
Access/Configuration Units

DOSDIRINFO

The DOSDIRINFO unit provides programmatic access to DOS directories. By using this unit, you can determine what files are on a volume, the total space on the volume, and the amount of remaining free space.

This is the interface section of the DOSDIRINFO unit, followed by descriptions of routines available to the programmer:

UNIT DOS_DIR_INFO;
INTERFACE
TYPE
DD_Attr $ PACKED_RECORD
   NeedsBackup,
   SubDirectory,
   VolLabel,
   SystemFile,
   Hidden,
   ReadOnly : BOOLEAN;
END;

FUNCTION DD_StartSearch (Name : STRING
   Hidden,
   SystemFile,
   SubDirectory : BOOLEAN) : BOOLEAN;

FUNCTION DD_GetInfo (VAR Title : STRING;
   VAR Attr: DD_Attr;
   VAR Month,
   Day,
   Year,
   Hour,
   Min,
   Sec,
   Blocks,
   Bytes : INTEGER) : BOOLEAN;

PROCEDURE DD_FreeSpace (Drive : CHAR;
   VAR TotalSpace,
   FreeSpace : INTEGER);

IMPLEMENTATION
DD_StartSearch

This function is used to find all the files that match a given pattern. The DOS pattern-matching conventions must be used. A result of TRUE indicates that at least one file matched the pattern. The parameters Hidden, SystemFile, and SubDirectory cause hidden files, system files, and directory files, respectively, to be included in the search. (See Appendix A for IORESULT codes.)

DD_GetInfo

This function is used to retrieve the information concerning files which DD_StartSearch has found. DD_StartSearch must be called before calling this function. A value of TRUE indicates that information is being returned, and a value of FALSE indicates that no more files match the pattern. (See Appendix A for IORESULT codes.)

DD_FreeSpace

This procedure may be used to get information about the free space on a DOS volume. The parameter, Drive, is the single-character drive name. If Drive is a <space>, then the default drive is assumed. TotalSpace is the size of the volume in 512-byte blocks. FreeSpace is the number of blocks available. (See Appendix A for IORESULT codes.)
FILERUNIT

The FILERUNIT unit provides two procedures and a function. Each of these routines is described in detail. The FILEIO unit must be used if FILERUNIT is to be used; this is required since the FILERUNIT uses the FILEIO unit in its interface section.

This is the interface section of the FILERUNIT, followed by descriptions of the routines available to the programmer:

UNIT FILERUNIT;
INTERFACES

USES (Bu fileio.code) file_io;

TYPE des_special = set of (des_die, des_tab, des_mof, des_crif, des.ctrl);
    sys_file = file;

PROCEDURE des_show_dir(var out:text; des_name:string; full:boolean);
PROCEDURE des_transfer(var files:file; var pviews:file;
                        translate_phases:des_special; from:des:boolean);
FUNCTION des_scan_name(des_name:string;
                        var volume, path, title, options:string):boolean;

IMPLEMENTATION

Access/Configuration Units

**dos_show_dir**

This procedure will write a directory listing to a file. It may be used to display the directory listing of a DOS disk on the screen.

The first parameter is a text file which is already open; it may be the console, the printer, or any text file. The second parameter is the file name of the directory to be listed. The directory specified by the name given will be printed to the text file. The third parameter is a boolean (true or false) that specifies if a full (extended) directory listing is to be written.

The directory will be formatted as shown in the examples on page 3–10 (extended) and page 3–15.

**dos_transfer**

This procedure may be used to transfer a DOS file to a p–System file or a p–System file to a DOS file. It will also (optionally) convert text file format from DOS format to p–System format.

The first parameter is a DOS file variable, the second parameter is a p–System file variable, and the third parameter is a set which specifies what conversion is to be done if the file is a text file. The fourth parameter is a boolean specifying whether to transfer from DOS to the p–System or from the p–System to DOS.

Both the DOS file and the p–System files must be open before the procedure is called. The source file must have been opened with a reset or dos_reset, and the destination file must have been opened with a rewrite or dos_rewrite.
Access/Configuration Units

The set should contain the conversions you want to have done to the file during the transfer. The actions of the various conversions are described in the section "Transfer From DOS." in Chapter 3. To select all conversion options, the actual set parameter should be [dos_dle, dos_tab, dos_eof, dos_crlf, dos_ctrl].

The IORESULT should be checked after the procedure call and will be 0 if the transfer was successful; otherwise, it will indicate the cause of the error. (See Appendix A for IORESULT codes.)

dos_scan_name

This function is used to break a file name into its individual parts.

The first parameter is the file name. The second parameter returns the volume name, followed by a colon; the third parameter returns the path (if any), followed by a backslash; the fourth parameter returns the title; and the fifth parameter returns the options (if any) in square brackets.

The function result is true if a valid file name was passed in the first parameter and false if the file name was invalid. The IORESULT is not set by this procedure and should not be checked.
VIRTVOl

The virtual volume (VIRTVOl) unit provides programmatic access to the virtual volume configuration mechanism. The interface section of this unit is as follows:

UNIT VIRTVOl;
INTERFACE

PROCEDURE VVMout (Name : STRING; (DOS file name)
VAR UnitNo : INTEGER);

PROCEDURE VVDismount (UnitNo : INTEGER);

PROCEDURE VVCreate (DName : STRING; (DOS file name)
PName : STRING; (p-System vol name)
Size : INTEGER);

PROCEDURE VVInfo (UnitNo : INTEGER;
VAR Name : STRING; (DOS file name)
VAR PName : STRING; (p-System volume name)
VAR ReadOnly : BOOLEAN);

PROCEDURE VVDevices (VAR FlacctY,
NumVV : INTEGER);
IMPLEMENTATION

VVMout

This procedure mounts a virtual volume. The first parameter, Name, indicates the DOS file which is to be used as a virtual volume. The second parameter, UnitNo, indicates the device number to be associated with this virtual volume. If UnitNo is less than or equal to zero, the first available virtual volume device number will be used. If no virtual volume device number is available, then I/O error 2, bad device number, is reported. If UnitNo is greater than zero and specifies a device that is not a virtual volume device, then I/O error 2, bad device number, is reported. If UnitNo specifies a virtual volume device which is already in use, the previous virtual volume is dismounted and the new virtual volume is mounted. The function IORESULT should be checked to determine the result of this procedure.
Access/Configuration Units

VVDisMount

This procedure dismounts a virtual volume. The parameter, UnitNo, specifies the virtual volume which is to be dismounted. If this device is not a virtual volume, then I/O error 2, bad device number, is reported. If no virtual volume is mounted on that device, then I/O error 9, no such volume on line, is reported. The function IORESULT should be checked to determine the result of this procedure.

VVCreate

This procedure creates a DOS file and puts a p-System directory in this file. The first parameter, DName, indicates the name of the DOS file which is to be created. If a file already exists by this name, it is removed. The second parameter, PName, is the p-System volume name which is to be used with this volume. If PName is not a legal p-System volume name, then the I/O error 7, bad file name, is reported. The third parameter, Size, indicates the size in 512-byte blocks of this virtual volume. The virtual volume created by this procedure is not automatically mounted. The function IORESULT should be checked to determine the result of this procedure.

VVInfo

This procedure returns the DOS file name which is being used as a virtual volume. The first parameter, UnitNo, specifies the virtual volume device. The second parameter, Name, is the DOS file name. The name is an empty string if UnitNo is not a virtual volume or if no virtual-volume is mounted in UnitNo. The third parameter, PName, is the p-System name of the virtual volume. If there is no p-System directory on this volume, then the null string is returned. The fourth parameter, ReadOnly, indicates whether or not the virtual
volume is available for write access. IORESULT is not set by this procedure and should not be checked.

VVDevices

This procedure returns the unit number of the first virtual volume device configured (FirstVV) and the number of virtual volumes configured (NumVV). Unit numbers for blocked devices are 4, 5, 9, 10, ... 127. If the first virtual volume is reported to be 4—and 3 virtual volumes are configured—then the virtual volume unit numbers are 4, 5, and 9. IORESULT is not set by this procedure and should not be checked.
Access/Configuration Units

CALLDOS

The CALLDOS unit enables direct access to DOS functions. The interface of this unit is:

UNIT CallDOS;
INTERFACE
TYPE
  DOS_Byte = 0..255;
  DOS_Buf = PACKED ARRAY [0..255] OF CHAR;
  DOS_Flag = PACKED RECORD
              Overflow;
              Direction;
              IntEnable;
              Trap;
              Sign;
              Zero;
              Filler3,
              AuxCarry;
              Filler2;
              Parity;
              Filler1;
              Carry;
              RD;
  END;

PROCEDURE DOS_FUNCTION
  (VAR AH,
   AL : DOS_Byte;
   VAR BX, CX, DX : INTEGER;
   DSBX, DSBK, DSBE, EDXI : DOS_Buf;
   VAR Status : DOS_Flag);
  {in/out register params}
  {non all values override registers}
  {status flags returned}
PROCEDURE StrToAscii (VAR a : STRING;
                      VAR Ascilz : DOS_Buf);
PROCEDURE AscilzToStr (VAR a : STRING;
                      VAR Ascilz : DOS_Buf);
PROCEDURE ExitDOS (ErrorCode : INTEGER);
IMPLEMENTATION
Access/Configuration Units

DOS Function

This procedure invokes DOS functions. These functions are the services which are accessed—via interrupt hex 21—in the DOS environment. This is a very low level interface and should not be used when units, such as the FILEIO unit, can be used.

The parameters AH, AL, BX, CX, and DX are used for setting up the registers as required for each DOS call. The register AH is always used for the function number when calling DOS. The results of the function call are returned via these registers and the flag register. Following the call to the DOS function, the contents of the registers are passed back in the parameters AH, AL, BX, CX, and DX. The contents of the flag register are returned via the parameter Status.

Many of the DOS calls require a pair of registers (such as DS:DX) to point to a buffer. The parameters DSDX, DSBX, DSSI, and ESDI are provided for setting DS:DX, DS:BX, DS:SI, and ES:DI, respectively. If a value, other than nil, is passed in as one of these parameters, then the buffer this pointer points to is passed to DOS, overriding the value specified for BX (in the case of DSBX) or DX (in the case of DSDX).

Note that CS does not point to the Program Segment Prefix when DOS is called; therefore, functions that require CS to point to the Program Segment Prefix, such as Program Terminate (function 0), must not be used.
Access/Configuration Units

**StrToAsciiz**

This procedure translates a p-System string into a DOS ASCII string. The first parameter, `s`, is the p-System string to be translated. The second parameter, `Asciiz`, a 512-byte block used for communicating with DOS, will contain the resulting string. Note that all DOS functions that take strings as parameters require them to be ASCIIZ strings. An ASCIIZ string contains a NUL character to mark the end of the string. A p-System string contains a leading "length" byte that indicates the number of valid characters that follow.

**AsciizToStr**

This procedure translates a DOS ASCIIZ string into a p-System string. See StrToAsciiz for more information.

**ExitToDOS**

This procedure terminates the p-System and returns control to DOS. When the p-System is invoked, interrupt vectors 0 through 2FH are saved. Terminating the p-System restores these interrupt vectors prior to returning to DOS. The parameter `ErrorCode` is the p-System completion code returned to DOS.
CLOCK UNIT

The CLOCKUNIT unit, located in SYSTEM.LIBRARY, offers a convenient programmatic way to access the time of day and the date. The interface to this unit is as follows:

```pascal
unit clockunit;

interface
type
  weekday = (Sun, Mon, Tue, Wed, Thu, Fri, Sat);
end record
  clockinfo =
    thousands: 0..9;
    hundredths: 0..9;
    tenths: 0..9;
    seconds: 0..59;
    minutes: 0..59;
    hour: 0..23;
    day: 1..31;
    weekday: weekday;
    month: 1..12;
    year: 1980..2079;
end (clockinfo);

procedure timedate(var c: clockinfo);
procedure set_timedate(c: clockinfo);

implementation
```

(Note that since DOS maintains the time of day only to the hundredths of a second, the thousandth of a second that is returned is always 0.)
DRIVER CONFIGURATION UNIT

The driver configuration unit, DRVCONFIG, allows your programs to do the same sort of operations as the driver installation and configuration portions of the Configure utility. Here is the interface section for this unit:

UNIT DrsConfig;

INTERFACE

CONST
DrsDictSize = 5 (bytes); (DrsDictSize=100; (DrsDictSize+32 = end of dictionary segment address)
EndDictAddr = 100; (DrsDictSize+32 = end of dictionary segment address)

TYPE
Byte = 0..255;
LongAddress = ARRAY [0..1] OF INTEGER; (32-bit address)
DeviceType = (NaDevice, Console, Printer, Disk,
user, System, Remote, Serial);
Packet = Packed ARRAY [1..1024] of 8..256;
DrsDict = PACKED RECORD
  Loaded: LongAddress;
  Param : Packet;
END;
DrsEntry = PACKED RECORD
  Device: DeviceType;
  DName : Byte;
  Dlen : Byte;
  DType : DeviceType;
  DiaLen: String(15);
  CodedAddr : Integer;
  CLen : Integer;
  LeaAddr : LongAddress;
  DiaParam : Packed ARRAY [0..256] of Byte;
END;
DrsBuffer = Packed Array [0..8] of DrsEntry;

Function DrsInstall (ConfigFilename, DrsFilename: string;
  ParamBuf: Packet;
  DevType: DeviceType; DevIndex: Integer) : integer;

Function Get_NameParam ( DevType: DeviceType; DevIndex: Integer;
  Var ParmBuf: Packet ) : integer;

Function Put_NameParam ( DevType: DeviceType; DevIndex: Integer;
  ParmBuf: Packet ) : integer;

Function Get_NameParam ( ConfigFilename: string;
  Device: DeviceType; DevIndex: Integer;
  Var ParmBuf: Packet ) : integer;

Function Put_NameParam ( ConfigFilename: string;
  Device: DeviceType; DevIndex: Integer;
  ParmBuf: Packet ) : integer;

5-22 1700101:05
As described under the Configure utility, you can configure the p-System to run with a particular peripheral device by placing the proper driver code into a file called SYSTEM.CONFIG on the boot disk. There are several parameters associated with each driver. (For example, disk drivers require information about the number of sectors per track, bytes per sector, and so forth.) When these parameters are changed, the behavior of the device driver changes.

The routines in this unit allow you to install drivers into SYSTEM.CONFIG and to change their parameters. They are described next.

Description of the Routines

DRVINSTALL places a driver code file into a configuration file (such as SYSTEM.CONFIG). CONFIG_FILENAME is the name of the configuration file. DRV_FILENAME is the name of the driver code file. PARMBUF is a list of parameters (as defined by the general type PLIST). DEVICE indicates the type of the driver (console, disk, and so on); and DEVIINDEX indicates the index of this device driver. Device indexes are described as follows.

GET_MEMPARAM obtains the parameters for a given device from memory. (These parameters were obtained from *SYSTEM.CONFIG at boot time.) DEVICE is the device type. DEVIINDEX is the device index (described in the following paragraphs). PARMBUF returns the parameters. You can alter these parameters if you wish.

PUT_MEMPARAM complements GET_MEMPARAM. It places the parameters in PARMBUF into memory for the device driver indicated by DEVICE and DEVIINDEX.
Access/Configuration Units

GET_FILEPARAM works like GET_MEMPARAM, except that the parameters are taken from disk rather than from memory. CONFIG_FILENAME is the name of the configuration file. (It might be "*SYSTEM.CONFIG"). DEVICE and DEVINDEX specify the device, as before; and PARMBUF returns the parameters.

PUT_FILEPARAM complements GET_FILEPARAM. It places the parameters in PARMBUF into the file specified by CONFIG_FILENAME. DEVICE and DEVINDEX determine the driver being updated.

Installing and Replacing Drivers

The DRVINSTALL routine, described previously, can install new drivers as well replace existing drivers. The file name of the driver being added to SYSTEM.CONFIG determines how that driver is treated. If a driver's file name is the same as another driver's file name was when it was added to SYSTEM.CONFIG, DRVINSTALL assumes that it is the same driver (perhaps modified slightly). If the driver's file name is different, DRVINSTALL assumes that the driver is different.

When you install a driver with a new file name, and the corresponding device does not currently have a driver in SYSTEM.CONFIG, you are simply adding a new driver.
Access/Configuration Units

When you install a driver with a new file name, and the corresponding device already has a driver, the old driver is replaced with the new driver. However, if any other devices were also using the old driver, they continue using that driver. The old driver is thrown away if the device which you specify was the only device using it; otherwise, it is kept.

When you install a driver that has the same file name as an existing driver, the new driver overwrites the existing driver. (It is assumed that you are adding a modified version of the same driver.) In this case, any other devices which previously also used the old driver will now use the new driver. The old driver is discarded.

You can determine the names of the drivers currently in SYSTEM.CONFIG by reading the first few blocks (5 blocks maximum) of SYSTEM.CONFIG. This is the dictionary for SYSTEM.CONFIG and consists of DICTENTRY’s as defined in the DRVCONFIG interface section. The name for each driver is contained in the DTITLE portion of DICTENTRY. By looking at all of the current DTITLE entries, you can determine if the name of the driver you are about to install is unique. (If it is not, you might want to warn the user that the new driver is going to replace several device drivers.)

Before you make this comparison, however, make sure that the file name of the new driver doesn’t contain a volume name, device number, asterisk, or colon. (You don’t have to strip these off when sending the file name to DRVINSTALL in the DRV_FILENAME variable. But it is necessary if you want to compare the file name directly with DTITLE in one of the DICTENTRY entries.)
Device Numbers and Indexes

Many of these routines require a device type and device index. The device type can be console, disk, and so forth, as shown in the interface section under DEVICETYPE. (Note that the SERIAL item within DEVICETYPE stands for user-defined serial devices.)

The device index is a 0-based index into SYSTEM.CONFIG for each type of device. For example, the driver for disk device #4 is assigned 0, #5 is assigned 1, and so on. The console, which only has one driver, is assigned device index 0. Usually, device indexes are actually only useful for disk drivers and user-defined serial device drivers: but they must be included for all types of drivers.

When interacting with a user, your programs should not ask for a device index, since it can be difficult for the user to determine which index corresponds to which device number. (This is especially true if there are several devices of the same type—as with disk devices.) Your program should map device numbers into device indexes and vice versa. If you ask the user for the type of a device (as well as the device number), you should be able to easily determine the device number.
Access/Configuration Units

However, if the type is a user-defined serial device, the mapping between device number and device index becomes more complicated. This is because you must know the first device number associated with user-defined serial devices to determine the correct index for the device number entered by the user. You can obtain the first number for user-defined serial devices by adding together the following two fields from SYSTEM.MISCFIELD:

FIRST SUBSIDIARY VOL NUMBER
MAX NUMBER OF SUBSIDIARY VOLS

If you are configuring a user-defined serial device for the currently booted system disk, you can obtain these two values from memory. To do this, your program must use the operating system unit KERNEL with the following statement:

USES (IU KERNEL CODE) KERNEL;

You can then access the two fields above in this manner:

SYS.COM$SUBSIDSTART
SYS.COM$UNITDIVISION.SUBSIDMAX
Access/Configuration Units

If you are configuring a user-defined serial device in a SYSTEM.CONFIG file for a boot disk other than the currently booted system, you will have to obtain these two values from a SYSTEM.MISCINFO file on disk. (You may have to ask the user for the location of the SYSTEM.MISCINFO that corresponds to the SYSTEM.CONFIG file you are updating.) In order to obtain these values, you should read the first block of the SYSTEM.MISCINFO file and access the following bytes:

FIRST SUB VOL NUMBER  Bytes: 26, 27 (Integer)
MAX NUMBER SUB VOLS  Byte: 46 (Byte)

Once you have added the two numbers together, you should subtract the result from the device number the user entered. The result is the corresponding device index for SYSTEM.CONFIG—if the user entered a correct device number.
CHAPTER 6
DEVICE DRIVERS
AND MEMORY
ALLOCATION
GENERAL

This chapter describes the purpose and use of the MS-DOS device driver set. Memory allocation is also described, with related discussions concerning extended memory and RAM disk. After this, machine-specific drivers and low level I/O are covered.

MS-DOS DRIVER SET

This section describes each device driver in the MS-DOS device driver set. These drivers may be installed in the SYSTEM.CONFIG file and are independent of the underlying machine. They allow you to access MS-DOS-supported devices. In addition to these MS-DOS device drivers, there are also machine-specific device drivers available for your system. You may use these machine-specific drivers (presented later in this chapter) in conjunction with the machine-independent MS-DOS drivers described here.

Virtual Volume Driver

The virtual volume driver (DOSVV.DRV) is the only MS-DOS driver that must be used with the MS-DOS-Hosted p-System. This driver makes a DOS file appear to be a p-System volume. You may configure any number of devices to be a virtual volume. The lowest-numbered device configured to be a virtual volume is used as the boot volume. This volume must contain the files required for booting the p-System. These files include SYSTEM.PME.86, SYSTEM.CONFIG, SYSTEM.MISCINFO, and SYSTEM.PASCAL.

When a DOS file is associated with a virtual volume device, the virtual volume is said to be mounted. Virtual volumes may be mounted by using: (1) the bootstrap program
Device Drivers/Memory Allocation

(PSYSTEM.COM), (2) the VIRTVOl unit (programmatically), or (3) the DOSFILER program.

**WARNING:** When using virtual volumes, try to avoid using them with identical names. Swapping disks which have virtual volumes with the same name can confuse the system, causing unpredictable results.

When building a SYSTEM.CONFIG file, the driver must be named DOS\DRV. Parameter 1 should be set to 3: the remaining parameters are not used.

**Virtual Console Driver**

The virtual console driver (DOSVC.DRV) provides keyboard input and screen output facilities to the p-System. This driver does not access the DOS 2.0 standard I/O devices; thus, the p-System doesn't recognize DOS I/O redirection. Normally, devices 1 and 2 are configured to use this driver. Use of this driver requires that DOS be configured to support ANSI screen control functions.

Parameter 1 controls screen control translation options. If parameter 1 is 0, then no translation occurs. A value of 1 causes screen control codes—which the stand-alone p-System recognizes for the IBM PC—to be translated into ANSI screen control codes. This translation function is useful when DOS is configured with the ANSI terminal emulator.

Parameter 2 controls keyboard translation options. If parameter 2 is 0, then no translation occurs. A value of 1 causes keyboard control codes—returned by PC DOS on an IBM PC—to be translated into the control codes used by the stand-alone p-System on the IBM PC. Note that PC-DOS translates the break key into <Ctrl> C. Since <Ctrl> C is normally used for other things in the p-System, you should
configure (via SETUP) some other key for the break key.

By selecting between the above translation options, ANSI versus STD SCREENOPS and GOTOXY, and SETUP's terminal configuration, this driver can support almost any terminal.

In cases where a machine—specific console driver exists (such as the IBM PC), you can use either of the drivers. The machine—specific driver is likely to be somewhat faster and provide more machine—specific features than this machine—independent driver.

Virtual System Driver

The virtual system driver (DOSVS.DRV) should be configured as device 0. This driver provides the TIME function; and when the p—System is halted, this driver returns control back to DOS. You can use a machine—specific system driver with the MS—DOS—Hosted p—System, however, the H(alt command on the p—System command menu normally doesn't give control to DOS.

Stand—alone adaptations (such as the IBM PC and TI PRO)—that support the timer event—have this support in the system driver. However, the machine—independent system driver does not provide this. So, to supply these features, a special system driver (DOSVSPC.DRV) is furnished for the IBM PC. This driver provides all the support of the stand—alone system driver, plus it gives control back to DOS when the p—System is halted.
Virtual Printer/Remote Driver

This driver, the virtual printer and remote driver (DOS\PR.DRV), enables you to access the DOS printer and auxiliary devices. The driver is a serial device driver and is usually configured as devices 6, 7, and 8.

Parameter 1 specifies whether this driver uses the printer device or the auxiliary device. A value of 0 selects the printer, and a value of 1 selects the auxiliary device.

This driver doesn’t support the UNITSTATUS function. A UNITSTATUS request to this driver always reports that the device is on line, no characters are queued, and the device is ready to accept a character. You can’t use the p-System print spooler with this printer driver, because it requires that the UNITSTATUS function be fully supported.

Bridge Driver

The bridge driver (BRIDGE.DRV) enables you to access DOS files through normal UCSD Pascal\*1/O intrinsics. That is, when using this driver, a Pascal RESET can be used to open a DOS file, WRITE can write to a DOS file, and so on. Use of the driver doesn’t prevent access to p-System files. If this driver is not installed, the normal Pascal I/O intrinsics apply only to p-System files.

The bridge driver must be installed for device numbers 128 through 143, and no more than 15 DOS files can be open concurrently. (The number of DOS files open does not affect the number of p-System files that can be open concurrently.)

This driver doesn’t use the 12 bytes of parameters.
MEMORY ALLOCATION

The memory available to the p-System is used as follows:

```
HIGH
     ------------------------
     | 8K  -->  DEVICE       |
     |     | DRIVERS           |
     |     |                    |
     |     | EXTRA CODE POOLS/RA M DISK |
     |     |                    |
     |     | CODE POOL 64K      |
     |     |                    |
     |     | p-SYSTEM            |
     |     | STACK/HEAP 64K     |
     |     |                    |
     | 17K  -->  PUB         |
     |     | Saved interrupt vector |
     |     | code to return to DOS |
     |     |                    |
LOW
```

On systems with less than 128K available to the p-System, there is no extra code pool or RAM disk; and the code pool may be smaller than 64K or even nonexistent.

Systems that have enough memory to support an extra code pool/RAM disk area can take advantage of this extra memory.
EXTENDED MEMORY

The extended memory option allows the p-System to run more efficiently if you have 128K or more of memory. Since extended memory allows more space for program code, larger programs can run when extended memory is in use. Also, fewer disk reads are required because code is less frequently swapped out of memory. This means that programs may run more quickly—since disk reads are time-consuming. If you have at least 128K of memory, always use extended memory.

As you received it, the p-System is set up to automatically configure itself to use the extended memory feature, if the hardware you are using has enough memory—at least 128K—to support it. You may alter the default memory configuration by using the Configure utility.
RAM DISK

The RAM disk is an option that uses extra memory as a virtual disk device called RAMDISK:. RAM disk is just like any other p—System volume, except that it exists in memory—and because it exists in memory, it is a volatile storage device. That is, information stored there is lost if the system is rebooted or powered off. When the p—System is booted, the RAM disk is automatically created whenever there is enough extra memory.

By default, the RAM disk is located immediately after the 64K area for extended memory. If you are not using extended memory, the RAM disk is placed immediately after the first 64K p—System area by default. (It's a good idea to always use extended memory if you have more than 64K. However, you should use RAM disk only if you have more than 128K.) The Configure utility allows you to change the location of the RAM disk. (You can't place the RAM disk at a lower location than the default, because this would interfere with other uses of memory. So, the p—System simply places RAMDISK: at the default location.)

The number of blocks allocated to RAMDISK: corresponds, by default, to all of the available extra memory. With the Configure utility, you can specify that less should be used.

CAUTION: When rebooting, don't leave important files on the RAM disk; if you do, they'll be lost.
Device Drivers/Memory Allocation

MACHINE-SPECIFIC DEVICE DRIVERS

This section describes the machine-specific device drivers available for the IBM Personal Computer.

Console Driver

The console driver (CONSOLE.DRV) provides keyboard input and screen output facilities to the p-System. You can use it in place of the DOS virtual console driver (DOSVC.DRV). The console driver doesn’t use DOS for the console I/O and, thus, doesn’t require that the ANSI terminal emulator be used with DOS. Configure the console driver for use with p-System device 1, which is the console device.

When this driver is used, the DOS function calls used for the screen or keyboard can’t be accessed.

Parameter byte 1 is used as follows:

\[
\begin{align*}
\text{bit 0} & : \text{color card flag} \\
\text{bit 1} & : \text{absolute video flag} \\
\text{bit 2} & : \text{absolute keyscan flag} \\
\text{bit 3} & : \text{fixed 40 column flag} \\
\text{bits 4-7} & : \text{reserved}
\end{align*}
\]

Parameter byte 2 is the video mode.

Parameter byte 3 is used as follows:

\[
\begin{align*}
\text{bits 0-2} & : \text{foreground color} \\
\text{bit 3} & : \text{intensity flag} \\
\text{bits 4-5} & : \text{background color} \\
\text{bit 7} & : \text{blinking flag}
\end{align*}
\]
Parameter byte 4 is used as follows:

- bits 0-4 : border color
- bits 5-7 : video page

Parameter byte 5 is the horizontal sync offset.

Parameter bytes 8 through 12 are not currently used.

The UNITSTATUS procedure returns a 30-word record. The control parameter can be used to select the type of information that is to be returned. If bit 14 is 0, then the standard unit status record is returned. The first word of this record indicates the number of characters waiting for I/O. Bit 0 of the status word selects the output channel (=0) or the input channel (=1). If bit 14 is 1, then the following record is returned:

```
StatusRec = PACKED RECORD
  cursor_row : byte;
  cursor_col : byte;
  cursor_start : byte;
  cursor_end : byte;
  screen_width : byte;
  screen_mode : byte;
  display_page : byte;
  write_page : byte;
  background : byte;
  foreground : byte;
  unused : byte;
  shift_status : byte;
  unused : byte;
  border_color : byte;
END;
```

Fixed Disk Driver

The fixed disk driver (FDISK.DRV) provides access to the p-System partition on a hard disk. When using this driver, configure it for devices 9 and higher. The first fixed disk
Device Drivers/Memory Allocation

device must be device number 9.

Parameter 1 is the disk type parameter and should always be 2.

Parameter 2 is the fixed disk drive number (0, 1, or 2).

Parameter 3 is the device number of the first fixed disk device and should always be 9.

Parameter 4 is not currently used.

Parameter 5 is the foreign write access flag.

Parameters 6 through 12 are not currently used.

Floppy Disk Driver

The floppy disk driver (FLOPPY.DRV) is used to access p-System floppy disks.

Parameter 1 is the disk type parameter and should always be 0.

Parameter 2 is the disk drive number (0, 1, 2, or 3).

Parameter 3 is the sector interleaving. A value of 1 is normally used.

Parameter 4 is the track-to-track skew. A value of 0 is normally used.
Device Drivers/Memory Allocation

Parameter 5 is the first Pascal track. A value of 0 is normally used.

Parameter 6 is the encoded bytes per sector (0=128, 1=256, or 2=512).

Parameter 7 is the number of sectors per track.

Parameter 8 is the number of tracks per side (35, 40, or 80). A value of 40 is normally used.

Parameter 9 is the step rate. A value of 8 is normally used.

Parameter 10 is defined as follows:

- **bit 0**: verify writes
- **bit 1**: autorecognize 8/9 sectors per track
- **bit 2**: cylinder access (normally false)
- **bit 3**: double sided disk

Parameter 11 is the head settle time (0=15, 1=14, ...15=0).

Parameter 12 is not currently used.

**Serial/Parallel Driver**

The serial/parallel driver (SERPAR.DRV) is used to access the serial and parallel ports. It is normally used for the printer and remote devices (6, 7, and 8).

Parameter 1 identifies the device type (0=printer, 1=remote, or other character device).
Device Drivers/Memory Allocation

Parameter 2 is the port number (0, 1, 2 for parallel; $80H$, $81H$ for serial).

Parameter 3 is used for the serial ports only. Bits 0 through 4 must be 0. Bits 5 through 7 indicate the baud rate (0 = 110, 1 = 150, 2 = 300, 3 = 600, 4 = 1200, 5 = 2400, 6 = 4800, 7 = 9600).

Parameter 4 is used only for the serial ports. Bits 5 through 7 must be 0. Bits 0 and 1 indicate the word length (1 = 7 bits, 2 = 8 bits). Bit 2 indicates the number of stop bits (0 = 1 bit, 1 = 2 bits). Bits 3 and 4 indicate the parity (0 = none, 1 = odd, 2 = none, 3 = even).

Parameter 5 is used only for the serial ports. This indicates the protocol to be used (0 = none, 1 = etx/ack, 2 = xon/xoff).

Parameter 6 is used for serial ports which are using the etx/ack protocol. This is the number of characters to send before sending an etx.

Parameter 7 is used for serial ports which are using the etx/ack protocol. This parameter is the number of characters received since an etx.

Parameter 8 is used for serial ports which are using the xon/xoff protocol. Bits 0–3 will equal 15 when waiting for an XON from input. Bits 4–7 will equal 15 when an XOFF has been sent.

Parameter 9 indicates whether (=1) or not (=0) the async character ready event has been enabled.
LOW LEVEL I/O

The UNITREAD, UNITWRITE, and UNITSTATUS procedures have a control parameter associated with them. This is described in the Internal Architecture Reference Manual. The control parameter for UNITREAD and UNITWRITE is:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - Async</td>
<td>1 - Indicates asychronous request</td>
</tr>
<tr>
<td>0 - Physsect</td>
<td>1 - Indicates Physical Sector mode for disk I/O</td>
</tr>
<tr>
<td>2 - Nospct</td>
<td>1 - Indicates no special character handling</td>
</tr>
<tr>
<td>3 - NoCRLF</td>
<td>1 - Indicates no LF's are to be appended to CR's</td>
</tr>
</tbody>
</table>

The control parameter for UNITSTATUS determines the definition of the status record:

<table>
<thead>
<tr>
<th>Bits: 13 14</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>Normal unit status</td>
</tr>
<tr>
<td>bit 0 = 0</td>
<td>bit 0 = status of the output channel</td>
</tr>
<tr>
<td>bit 0 = 1</td>
<td>bit 1 = status of the input channel</td>
</tr>
<tr>
<td>0 1</td>
<td>rec01 status</td>
</tr>
<tr>
<td>1 0</td>
<td>rec10 status</td>
</tr>
</tbody>
</table>
Device Drivers/Memory Allocation

The UNITSTATUS command returns information on the status of each device for either the input or output channels. The information is returned in a 30-byte record called STATUSWORDS. Here is the standard record:

```
status record
  length: integer;  (# chars waiting for I/O)
  case device of
    disk: (bytespersector: integer;
           sectorspertrack: integer;
           tracksperdevice: integer;
    end;

Console Status Records

  rec 10 = packed record (read only)
    cursor_row: byte;
    cursor_column: byte;
    cursor_start,
    cursor_end: byte;
    screen_width,
    screen_height;
    display_page;  
    write_page: byte;
    background,  
    foreground: byte;
    unused,
    shift_status: byte;
    unused,
    border_color: byte;
  end;

Disk Status Records

  The standard disk status record is used here.
```
Printer Status Records

rec01 = record
  printerdevice: integer; (Parallel A..D, Serial A..D)
end;

rec10 = packed record
  timeout: boolean;
  unused: 0..3;
  ierror: boolean;
  selected: boolean;
  autofeed: boolean;
  acknowledge: boolean;
  not busy: boolean;
end;

When a serial printer is used, rec10 will return the same values as when doing a UNITSTATUS of REMIN or REMOUT.
CHAPTER 7
WRITING A
DEVICE DRIVER
GENERAL

To write a device driver, you should be familiar with the information contained in the Internal Architecture Reference Manual in the "Low-Level I/O" section. The following information will assist you in writing device drivers.

WRITING A DRIVER

A device driver is an assembly code file which the Compress utility must process and make into a nonrelocatable code file with a base address of 0. The code itself must begin with a word that points to the last byte of the driver. Following this, there must be a jump vector with entries for each of these five routines, as sequentially shown here:

    UnitRead
    UnitWrite
    UnitStatus
    UnitClear
    Initialize (one time initialization of the driver)
Writing A Device Driver

For example, a console driver has the following format:

```
    .PROC CONSOLE
    CODESIZE .WORD ENSADDR
    JMP CONSRD
    JMP CONSRD
    JMP CONST
    JMP CONSLCR
    JMP CONSRIN

    CONSRD ;code to read a char from console
    RETL

    CONSRD ;code to write a char to console
    RETL

    CONST ;code to check input/output status of console device
    RETL 4

    CONSLCR ;code to 'uninitialize' console device
    RETL 4

    CONSRIN ;code to initialize console
    RETL

    ENDSR .EQU %
    .END
```

Before control is passed to a device driver, the CS register is altered to reflect the beginning of the driver (intersegment call). This allows all relative jump instructions within the driver to be performed relative to the starting address of the driver. The return long (RETL) instruction at the end of each driver routine is used to restore the CS and IP registers of the BIOS calling routine.

Your driver should always return the correct p-System 1/O result (completion code) in the AH register.

Except for the parameters explained ahead, your drivers may use the following registers freely: AX, BX, CX, DX, BP, SI, and DI. Registers DS, SS, and ES must be returned unchanged.
All parameters which the BIOS passes on the stack must be removed (popped) in the driver routine before returning to the BIOS.

At every driver call, the BIOS passes a pointer (in registers DS:SI) to the "device parameter block" which is 12-bytes long. You may define a device parameter block as desired. The Configure utility allows you (or a user) to set device parameter block fields through the "User-Defined Driver" option of "Change Driver Parameters," as well as when installing a new driver.

Although the bytes in the device parameter block are open to any use you desire, we recommend that the first byte of a disk device parameter block be set as follows:

```
FIRST BYTE = 0  Floppy device
               1  RAM disk device
               2  Fixed disk device
               3  Other disk (blocked) device
```

When this first byte is a three, the Configure utility knows that it is a user-defined device and allows you to modify the device parameter block as such.
Writing A Device Driver

For the initialize routines (such as CONSOLEINIT), the BIOS passes an additional pointer (in registers DS:BX) to the "PME parameter block." The PME parameter block is always defined like this:

Word 0 - EVENT_HANDLER ; Address of event handler in the BIOS
Word 1 - IPETCH ; Address of IPETCH in the PME
Word 2 - RDPTASK ; Address of RDPTASK in the PME
Word 3 - CURTTASK ; Address of CURTTASK in the PME
Word 4 - SYSCOMP ; Address of SYSCOMP
Word 5 - FREE_HISEG ; Free memory highest segment address
Word 6 - BREAK_HANDLER ; Address of BREAK handler in the BIOS
Word 7 - SECTOR_BUFFER ; Address of $12 byte buffer for disk devices
Word 8 - TIMER_LOW ; Timer value lower 16 bits
Word 9 - TIMER_HIGH ; Timer value upper 16 bits
Word 10 - BREAK_KEY ; Break Hit, Reset at Console Clear

The following is a list of routines with the parameters the BIOS needs or uses. When the parameters refer to a device number, this device number is the relative zero-based device number. In other words, drive 4 is drive device number 0; and the first serial device will be serial device 0. Remember that at every call, the BIOS passes the device parameter block in the DS:SI.

<table>
<thead>
<tr>
<th>Routines for</th>
<th>PARAMETERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console Device</td>
<td>(device 0's 1-2)</td>
</tr>
<tr>
<td>READ</td>
<td>Return data byte in AL.</td>
</tr>
<tr>
<td>WRITE</td>
<td>Write data byte in AL.</td>
</tr>
<tr>
<td>CLEAN</td>
<td>BREAK vector at (SP)+4,(SP)+6.</td>
</tr>
<tr>
<td>STATUS</td>
<td>SYSCOMP pointer at (SP)+8,(SP)+7.</td>
</tr>
<tr>
<td>INIT</td>
<td>STATREC pointer at (SP)+4,(SP)+6.</td>
</tr>
<tr>
<td></td>
<td>CONTROL word at (SP)+6,(SP)+7.</td>
</tr>
<tr>
<td></td>
<td>PME parameter block passed in DS:BX.</td>
</tr>
</tbody>
</table>
Writing A Device Driver

Routine for
Printer Device
(device #0)

READ
Return data byte in AL.
WRITE
Write data byte in AL.
CLEAR
No parameters used.
STATUS
STATREC pointer at (SP)+4,(SP)+5.
CONTROL Word at (SP)+6,(SP)+7.
INIT
PWE parameter block passed
in DS:BX.

Routine for
Disk Device
(device #2's 4,5
9,Fire&Device - 1)

READ
Block number at (SP)+4,(SP)+5.
Byte count at (SP)+6,(SP)+7.
Data area address at (SP)+8,(SP)+9.
Drive number at (SP)+10,(SP)+11.
Control word at (SP)+12,(SP)+13.
Data area segment in ES.
Same as DISKREAD.
WRITE
Same as DISKWRITE.
CLEAR
Drive number in CL.
STATUS
STATREC pointer at (SP)+4,(SP)+5.
CONTROL Word at (SP)+6,(SP)+7.
INIT
PWE parameter block passed
in DS:BX.

Routine for
Remote Port Device
(device #7's 7,8)

READ
Return data in AL.
WRITE
Write data in AL.
CLEAR
No parameters used.
STATUS
STATREC pointer at (SP)+4,(SP)+5.
CONTROL Word at (SP)+6,(SP)+7.
INIT
PWE parameter block passed
in DS:BX.
Writing A Device Driver

Routines for
User Defined
Serial Device
(device #1's FirstSubDev+1 = SubDevMax ... - SerialMax)

READ
Return date byte in AL.
Device number in CL.

WRITE
Write date byte in AL.
Device number in CL.

CLEAR
Device number in CL.

STATUS
STATREC pointer at (SP)+4, (SP)+5.
CONTROL word at (SP)+6, (SP)+7.

INIT
PME parameter block passed
in DS:BX.

Routines for
User Defined
Device
(device # @ 128 .. 285)

READ
Black number at (SP)+4, (SP)+5.
Byte count at (SP)+6, (SP)+7.
Data area address at (SP)+8, (SP)+9.
Device number at (SP)+10, (SP)+11.
Control word at (SP)+12, (SP)+13.

WRITE
Same a USERREAD.
Device number in CL.

CLEAR
Device number in CL.

STATUS
STATREC pointer at (SP)+4, (SP)+5.
CONTROL word at (SP)+6, (SP)+7.

INIT
PME parameter block passed
in DS:BX.

Routines for
System Device
(device # 0)

READ
Black number at (SP)+4, (SP)+5.
Byte count at (SP)+6, (SP)+7.
Data area address at (SP)+8, (SP)+9.
Device number at (SP)+10, (SP)+11.
Control word at (SP)+12, (SP)+13.

WRITE
Same a SYSTEMREAD.
Device number in CL.

CLEAR
Device number in CL.

STATUS
STATREC pointer at (SP)+4, (SP)+5.
CONTROL word at (SP)+6, (SP)+7.

INIT
PME parameter block passed
in DS:BX.
Writing A Device Driver

A system driver (as in the previous example) is required for the p−System to boot. The STATREC for the system device is defined as follows:

```plaintext
Statrec packed record
  stack_heap_memory_size: integer;
  clock_low: integer;
  clock_maw: integer;
end;
{ Clock is a 32 bit (80 MHz). }
```
CHAPTER 8
SPECIAL CONSIDERATIONS
Special Considerations

GENERAL

This chapter discusses the special considerations that must be given to hardware requirements, special keys, and Turtlegraphics, when using the IBM Personal Computer.

HARDWARE REQUIREMENTS

The minimum IBM PC hardware configuration that can run the MS-DOS-Hosted p-System is 128K of memory and one disk drive.

The p-System uses the various switches on the IBM PC motherboard to tell it what equipment you have. You must set these correctly. For instance, you must select the type of display you're using and the number of disk drives.
Special Considerations

SPECIAL KEYS

The p-System uses several "special keys." Special keys perform various functions, such as accepting input or moving the cursor in a particular direction.

It's not possible to have one physical key for each special key, because there aren't enough extra physical keys. Therefore, combinations of two physical keys are used for some of the special keys. These multiple key combinations involve the special keys <Ctrl>, <Shift>, and <Alt>. (The <Shift> key is marked with a wide arrow pointing up at the bottom right and left corners of the main keyboard.) When using these key combinations, you should first depress the <Ctrl>, <Shift>, or <Alt> key, then, while it is depressed, press the other key.
Special Considerations

The following special keys are used with the p—System:

- `<Enter>`: Carriage return
- `<Backspace>`: Character delete
- `<Esc>`: Escape
- `<Ctrl>` `<Enter>`: Line feed
- `<Ctrl>` `<Backspace>`: Delete line
- `<Ctrl>` C: ETX, Editor Accept, End of File
- `<Ctrl>` Q: DC1. (must press twice)
- `<Ctrl>` S: Start/stop console output
- `<Ctrl>` F: Flush console output
- `<Ctrl>` `<Break>`: Break
- `<Ctrl>` `<Numlock>`: Suspend processor, any other key to restart
- `<Ctrl>` `<Left—Arrow>`: Move screen left
- `<Ctrl>` `<Right—Arrow>`: Move screen right
- `<Shift>` `<PrntSc>`: Print Screen
- `<Alt>` L: Switch to color adapter
- `<Alt>` M: Switch to monochrome adapter
- `<Alt>` 0..9: Any value entered
- `<Ins>`: Editor Exchange—Insert
- `<Del>`: Editor Exchange—Delete
- `<Caps Lock>`: Invert shift key for alphabetic characters.
- `<NumLock>`: Change cursor pad to numeric keypad
- `<Ctrl>` `<Alt>` `<Del>`: System Reset (Reboot)

The following paragraphs contain a few notes on some of these special keys. There is more information about the codes returned by various keys in the "Keyboard Codes" section of Chapter 10. "Programmer Information."

The `<enter>` key is the bent left arrow located just to the left of the Home key. Since this key is usually used as a carriage return, it is often denoted in prompts as `<cr>` or `<ret>`.
Special Considerations

The <Alt> L and <Alt> M key combinations switch between a color adapter card and a monochrome adapter card, if you have both cards installed.

The <Ctrl> <Left-Arrow> and <Ctrl> <Right-Arrow> key combinations are used for 40-character displays only. These keys shift the current display 20 columns to the left or right, respectively. This enables you to effectively use 80 columns even if your display is a TV screen (which can only clearly display 40 columns at a time). You can enable or disable these keys with the Configure utility, using the Console option of the Change Driver Parameters menu. (The Configure utility is described in Chapter 4.) The reason that you might want to disable these keys—even if you are using a 40-column display—is that there are some application programs that won't work as expected if these keys are enabled.

The <Alt> key can be used in combination with a sequence of digits entered from the keypad. The resulting ASCII value is entered.
Special Considerations

USING TURTLEGRAPHICS

The Turtlegraphics feature is used by certain programs to display graphics on your video display. If you are planning to use any Turtlegraphics programs, there are a few points you should be aware of.

There are two versions of Turtlegraphics available with the current release of the p-System. If you are going to use Turtlegraphics programs which are written for current release on a 1-word system, you won't have any problem. The TURTLEGRAPHICS unit in SYSTEM.LIBRARY is the 4-word version for the current release.

However, if you plan to use any Turtlegraphics programs which were written for a 2-word system, you must use a different TURTLEGRAPHICS unit. The 2-word TURTLEGRAPHICS unit for the current release is named TURTLE2.CODE. In order to use the 2-word version, it must be placed in SYSTEM.LIBRARY.

CAUTION: Version IV.0 Turtlegraphics is different than IV.1, so programs using IV.0 Turtlegraphics may not run.

You can use the Library utility (described in the Operating System Reference Manual) to place the new TURTLEGRAPHICS unit into a new copy of SYSTEM.LIBRARY. After you move the new TURTLEGRAPHICS unit into the new SYSTEM.LIBRARY, you should move any units—except TURTLEGRAPHICS—from the current SYSTEM.LIBRARY over to the new SYSTEM.LIBRARY. Before you do this, you should save the old copy of SYSTEM.LIBRARY. (In addition, SYSTEM.FONT must be placed on the root volume; if it isn't, the system won't run.)
CHAPTER 9

UTILITY PROGRAMS
Utility Programs

GENERAL

This chapter describes the p-System utility programs that are specific to the IBM Personal Computer. Other utility programs are described in the Operating System Reference Manual and the Program Development Reference Manual.

The utilities described here are:

- Diskformat
- Extrapools
- Fdisk
- NCICopy
- Startup

Diskformat is used to format a disk so you can begin to use it.

Extrapools allows the operating system to be moved to another code pool, which improves overall system performance.

Fdisk—used only if you have a fixed disk—enables you to partition, back up, and restore the fixed disk.

CAUTION: Before you use Fdisk, be sure to back up the disk you are reconfiguring. This is an important safety precaution. Since, if you make a mistake, your disk may no longer bootstrap correctly.

With NCICopy, you can copy disks (which were formatted as 10 sectors per track under the NCI p-System) onto any blocked p-System volume.
Utility Programs

Startup automatically performs certain functions normally desired after booting the system.
Utility Programs

DISKFORMAT

The Diskformat utility formats a p-System disk so it may be used on the IBM Personal Computer. To format DOS disks, use the DOS Format command as described in the DOS Operating System Manual. This utility first prompts for the disk device number of the disk to be formatted:

```
Diskette Formatter

Enter diskette device number to format (4,5,6,10): _
```

At this point, enter the appropriate number. (Do not enter a preceding number sign or trailing colon.) Once you have specified the device number, you are shown the current parameters of the selected drive. These parameters are used to format the disk. If you want to format the disk with a different set of parameters, you must exit the Diskformat utility. Then, use the Change Driver Parameters option of the Configure utility to set the drive parameters to the desired values. (The Configure utility is described in Chapter 4.)
Utility Programs

For example, if device 4 is selected, the following display may appear:

```
Diskette Formatter [ ]

Enter diskette device number to format (4,5,6,9,10): 4
```

**Parameters of Disk Device 4:**

- **Bytes per Sector** = 512
- **Sectors per Track** = 8
- **Tracks per Side** = 40
- **Number of Sides** = 2

Are these the correct format parameters (Y/N)? Y

(*N* will exit the formatter)

Note that if the bytes per sector parameter is not 512, Diskformat terminates after displaying this message:

```
Formatter only supports 512 bytes per sector.
Use the CONFIGURE utility to correctly configure device #
```

If this happens, you must run Configure and use the Change Driver Parameters option to set the bytes per sector parameter to 512 for the indicated drive.
There is a mode where the p-System automatically recognizes whether a disk is formatted in 8 or 9 sectors per track. (This mode is available from within the Configure utility and is explained under that utility in Chapter 4.) If you are using the 8/9 mode, Diskformat can't tell which of the two formats you would like to use for the new diskette. In this case, the Diskformat utility asks you as follows:

```
Diskette Formatter [ ]

Enter diskette device number to format (4,5,6,10): 4

Parameters of Disk Device 4:

  Bytes per Sector = 512
  Sectors per Track = 7
  Tracks per Side = 40
  Number of Sides = 2

Automatic recognition of 8 or 9 sectors / track is on.
Enter desired sector density (8/9) or "N" if incorrect format: _
```

If you respond by pressing N in response to either version of this display, the following message appears before Diskformat terminates:

```
Use the CONFIGURE utility to correctly configure device 4
```

If you respond by pressing Y for the first display, or if you respond by pressing 8 or 9 for the second menu, this prompt appears:

```
Insert diskette in drive. Press any key to continue: _
```
Utility Programs

You should place the disk to be formatted into the appropriate drive. If the disk you are about to format already contains a p-System directory, you are asked to verify that you want to destroy the old information. For example, if the disk name is VOLNAME:, this prompt appears:

\textbf{Destroy VOLNAME: (Y/N)?}

If you press \textbf{Y}, Diskformat returns to the prompt which asks you to enter a device number. If you press \textbf{Y} (or if this prompt did not appear at all), the format operation proceeds.

\textbf{NOTE:} Diskformat can't recognize a valid disk if the interleave, skew, or first p-System track is configured differently from the diskette format.
Diskformat displays a dot for every track which is successfully formatted. Upon conclusion of a successful format operation, you are asked for a p—System volume name. The following display shows this (after a double—sided 840—block disk has been formatted):

```
Diskette Formatter [ ]

Enter diskette device number to format (4,6,9,10): 4

Parameters of Disk Device 4:

Bytes per sector = 812
Sectors per track = 8
Tracks per side = 40
Number of sides = 2

Formatting side 0

Formatting side 1

840 block disk formatted
Enter volume name for formatted disk (up to 7 chars): _
```

The name that you enter will be given to the newly formatted disk. This name should follow the volume naming conventions in the Operating System Reference Manual. Briefly, this means that it must contain 7 or fewer characters; these characters may consist of letters, numbers, and certain special symbols.

After you enter the volume name, you are asked if you want to format another diskette:

Format another diskette (Y/N)? _
Utility Programs

If you press Y, you are returned to the first prompt. If you press N, Diskformat is exited.

If an error is detected during the format operation, Diskformat displays the track and sector number at which the error occurred. The current format operation is aborted and you are asked if you wish to format another diskette. The error message looks like this:

Format error ## at track ##, sector ## — Operation aborted
Utility Programs

EXTRAPOLLS

The Extrapol loops utility, as configured for the IBM PC, allows the operating system to be moved to another code pool. The utility can be used only on systems that have 4-word real numbers installed. The system disk that accompanies this utility is configured and shipped as a 4-word system.

The program EXTRAPOLLS.CODE moves the operating system from the main code pool, which is under control of the operating system, to another code pool, which is not under control of the operating system. This frees the main code pool space normally occupied by the operating system and greatly reduces the contention for code pool space between the operating system and other units. Thus, swapping is drastically reduced, which improves performance for most applications.

The following procedure provides optimum utilization of "extra" memory when using the p-System and systems having at least 192K of memory.

1. Using the Configure utility, set the RAM disk driver's size parameter to zero. so all available memory will be allocated to RAM disk.

2. Boot the system and check the RAMDISK: directory to determine its size.

3. Use the Configure utility again and set the RAM disk driver size to the number of blocks previously allocated, less the size in blocks of SYSTEM.PASCAL. For optimum utilization, the size of the RAM disk can be set approximately 30--blocks larger than that difference. This is because SYSTEM.PASCAL will have some code and other information that won't be needed after the system is loaded.
Utility Programs

4. Boot the "reconfigured" system and then execute the Extrapol utility. If you use the startup program, change LOADFILES to remove any transfer of SYSTEM.PASCAL, and add a statement to execute Extrapol (that is, \xj#4:extrapol).

If you don’t want to use the startup program to load your RAM disk, you can rename EXTRAPOLS.CODE to SYSTEM.STARTUP; or you can execute Extrapol directly from the command menu after you boot up.

5. If the memory allocated to RAM disk is too large, then a message "Not enough memory to load entire operating system" is displayed when Extrapol is executed.

NOTE: Extrapol will load as much of the operating system as will fit into available memory.
Utility Programs

FDISK

The Fdisk utility is used with the IBM PC XT fixed disk. It provides the following services:

- Manages multiple IBM—fixed disk drives.
- Creates or deletes p—System partitions on fixed disk drives.
- Backs up and restores files in a p—System partition.
- Designates a fixed disk partition as bootable.

**NOTE:** The Fdisk utility only runs in the extended memory configuration of the p—System (this requires at least 128K of RAM).

The fixed disk is briefly described in the following paragraphs. After that, the Fdisk utility is covered. If you are interested in writing programs that perform the same sort of fixed disk operations as this utility, you should refer to the description of the FDISK unit in Chapter 10, "Programmer Information," in this supplement.
Utility Programs

General Description

The p-System can support multiple IBM PC XT fixed disks. Each fixed disk has a capacity of 10 megabytes and contains two platters. Disk memory is measured in terms of "cylinders." A cylinder consists of four tracks: a track on the top of the first platter, the corresponding track on the bottom of the first platter, the corresponding track on the top of the second platter, and the corresponding track on the bottom of the second platter. Altogether, the fixed disk contains 305 cylinders, numbered 0 through 304.

The disk may be logically divided into as many as four separate areas called "partitions." You can specify the size (in cylinders) of each partition. Each partition starts on a cylinder boundary, except for the first partition, which starts on the second sector of the first cylinder. (A sector is 512 bytes.)

With the p-System, each partition corresponds to a volume and is treated like a completely separate physical disk. You may select one partition to be designated as bootable. This partition must be on fixed disk 0; that is, if you have more than one fixed disk, you can only boot from a partition on the first one. When you designate a partition as bootable, you can bootstrap and run the p-System directly from the fixed disk (rather than from floppy disk). This causes increased performance because of quicker disk access time. Also, it is convenient to have the p-System reside on the fixed disk, since the floppy disk drive is completely free for other purposes.
Utility Programs

Each partition must be designated for a particular operating system. However, any partition's bootable status can be changed by the p-System—even if that partition is assigned to another operating system.

With the p-System, you can support two or more floppy disks, RAMDISK; and as many as four fixed disk volumes per fixed disk (one volume for each p-System partition). You can further divide the fixed disk volumes into subsidiary volumes if you wish. (Subsidiary volumes are described in the Operating System Reference Manual.) Dividing the fixed disk into several volumes (either using partitions, subsidiary volumes, or both) allows you to store many more files on the fixed disk than you could otherwise. (The current version of the p-System with the Standard File System allows a maximum of 77 files per volume.)

The p-System device numbers are assigned to floppy disks, RAMDISK:, and fixed disk p-System partitions according to the following table:

<table>
<thead>
<tr>
<th>Systems With One Floppy</th>
<th>Systems With Two Floppies</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 floppy disk 1</td>
<td>4 floppy disk 1</td>
</tr>
<tr>
<td>5 offline</td>
<td>5 floppy disk 2</td>
</tr>
<tr>
<td>6 RAMDISK</td>
<td>6 RAMDISK</td>
</tr>
<tr>
<td>10 partition 1</td>
<td>10 partition 1</td>
</tr>
<tr>
<td>11 partition 2</td>
<td>11 partition 2</td>
</tr>
<tr>
<td>12 partition 3</td>
<td>12 partition 3</td>
</tr>
<tr>
<td>13 partition 4</td>
<td>13 partition 4</td>
</tr>
<tr>
<td>14 subsidiary volume 1</td>
<td>14 subsidiary volume 1</td>
</tr>
</tbody>
</table>

etc.
Utility Programs

This table assumes that you are using one fixed disk—although you can use more. If you use a second fixed disk, p-System partitions for that disk are assigned device numbers starting at #14 (even if you don't use all four partitions as p-System volumes on the first fixed disk). There are always four device numbers set aside for the four possible partitions for each fixed disk. (This simplifies matters if you add or delete p-System partitions; the first subsidiary volume number doesn't have to change each time.)

For example, if you have two fixed disks, each of which has three p-System partitions (and you have two floppy disks and RAMDISK), device numbers might be assigned as follows:

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#4</td>
<td>floppy disk 1</td>
</tr>
<tr>
<td>#6</td>
<td>floppy disk 2</td>
</tr>
<tr>
<td>#9</td>
<td>RAMDISK:</td>
</tr>
<tr>
<td>#10</td>
<td>fixed disk 1, p-System partition 0</td>
</tr>
<tr>
<td>#11</td>
<td>fixed disk 1, p-System partition 1</td>
</tr>
<tr>
<td>#12</td>
<td>fixed disk 1, p-System partition 2</td>
</tr>
<tr>
<td>#13</td>
<td>(not used)</td>
</tr>
<tr>
<td>#14</td>
<td>fixed disk 2, p-System partition 0</td>
</tr>
<tr>
<td>#15</td>
<td>fixed disk 2, p-System partition 1</td>
</tr>
<tr>
<td>#16</td>
<td>fixed disk 2, p-System partition 2</td>
</tr>
<tr>
<td>#17</td>
<td>(not used)</td>
</tr>
<tr>
<td>#18</td>
<td>subsidiary volume 1</td>
</tr>
<tr>
<td></td>
<td>etc.</td>
</tr>
</tbody>
</table>

If you plan to use subsidiary volumes, you should first decide how many of them you want. Then set MAX NUMBER OF SUBSIDIARY VOLS to that number using the Setup or Configure utility.
Also, you must determine the device number of the first subsidiary volume. This should be the first available device number after the highest-numbered device set aside for p—System partitions. (For example, if you have two floppy disks, RAMDISK..., and you are using one fixed disk and three virtual volumes, the first subsidiary volume would be assigned to 17.) You should then properly set the FIRST SUBSIDIARY VOL NUMBER field by using the Setup or Configure utility. Whenever you want to check to see what device numbers are assigned to your volumes, you can simply use the filer’s V(olumes command.
Utility Programs

Main Menu

When you execute Fdisk, the following main menu of options is displayed. To provide you with current information during most fixed disk operations, the status of the fixed disk partition is constantly displayed and updated on the lower half of the screen.

Fixed Disk Utility [ ]

A) Select Fixed Disk Drive
B) Set Bootable Partition
C) Create p-System Partition
D) Delete p-System Partition
E) Back-up p-System Partition
F) Restore p-System Partition
Q) Quit. Exit FDISK.

Enter letter of option (A..Q): _

Current Fixed Disk Drive = 1

<table>
<thead>
<tr>
<th>Partition Number</th>
<th>Boot</th>
<th>Operating System</th>
<th>Volume Name</th>
<th>Start cyl.</th>
<th>End cyl.</th>
<th>Size (blocks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NO</td>
<td>(1)</td>
<td>BOOTPSY</td>
<td>0</td>
<td>99</td>
<td>6799</td>
</tr>
<tr>
<td>2</td>
<td>YES</td>
<td>p-System</td>
<td>LISTING</td>
<td>100</td>
<td>199</td>
<td>6800</td>
</tr>
<tr>
<td>3</td>
<td>NO</td>
<td>p-System</td>
<td>UTILS</td>
<td>200</td>
<td>259</td>
<td>4080</td>
</tr>
<tr>
<td>4</td>
<td>NO</td>
<td>p-System</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The status report at the bottom first indicates the fixed disk drive being accessed and the size and location (in cylinders) of the largest available free space on that partition.
Utility Programs

For each partition, the status report indicates: the partition number; whether or not the partition is bootable; the operating system assigned to the partition; the p-System volume name, if applicable; and the partition’s location and size.

The Boot field shows which partition is bootable. There can only be one bootable partition per system.

The Operating System field shows the type of operating system currently assigned to the partition. Only the p-System name is displayed in full. Other types are indicated by a number code: (0) for unknown, (1) for DOS, and (2) for CP/M®.

The volume name field indicates the p-System volume name that corresponds to the partition number.

The Start cyl. and End cyl. fields show the starting and ending cylinder numbers of the partition. Unallocated space may exist between partitions. The largest unallocated space is listed.

The Size field contains the number of 512-byte blocks allocated to the partition. You can enter this number to answer the Filer prompt for disk size when you zero the p-System partition.

The options on the main menu are described in the following section.
Utility Programs

Select Fixed Disk Drive

This option allows you to select the fixed disk that the Fdisk utility is currently working with. You may have multiple fixed disks. But, Fdisk must configure the p-System for only one at a time.

When you select this option, if you have already made changes related to the current fixed disk, you are asked if you want to save those changes:

Make changes permanent (update partition table) (Y/N)?

Pressing Y saves the changes made for the current fixed disk, and pressing N discards them.

Next, the following prompt asks you to select the fixed disk drive that you now want to configure:

Enter Fixed Disk Drive number (1..3): _

Create p-System Partition

This option allows you to create a new p-System partition. While doing this, it doesn't disturb any partitions allocated to other operating systems.
Utility Programs

A new p-System partition can be created if: (1) there is unallocated space and (2) there are currently fewer than four partitions. When a new p-System partition is created, you are given the choice of zeroing or not zeroing the volume which corresponds to that partition. Zeroing the volume is equivalent to using the filer’s Z(ero) command on it—the directory is initialized. However, you should zero the partition with Fdisk instead of with the filer. Unless you zero the volume, it won’t be accessible.

There is an exception to this last rule: if a directory already exists where you are creating the new partition (because a p-System partition previously existed there), then that directory will remain if you don’t zero the new partition. However, it’s a good idea to always zero a new partition. If you rely on an old directory, the files it lists may have been overwritten and the directory size may not match the size of the current partition.

NOTE: When a new partition is created (or an existing partition is deleted), the assignment of partition numbers to existing partitions may be altered.

The following prompts are displayed by this option:

Enter cylinder size of the new partition (Esc to exit): XXX
Zero new partition directory (Y/N)? Y or N
Enter volume name: VOLUME; (enter)
Utility Programs

In response to the first prompt, you should enter the size (in cylinders) that you want the partition to be. (Part of the disk status display indicates the size, in cylinders, of the largest available free area.) The new cylinder is always created in the largest available area.

The second prompt asks you if you want to zero (initialize) the p–System directory for the volume that corresponds to the new partition. As mentioned previously, if you don’t zero the volume, it won’t be accessible to the p–System unless an existing p–System directory is already there.

If you do elect to zero the volume, the next prompt appears and asks what volume name you would like it to have. You should enter a valid p–System volume name (seven characters or less), followed by <enter>.

**NOTE:** When you use Fdisk to zero a volume, the volume is immediately zeroed. Remember, with all other Fdisk actions, you must confirm the changes that you make when you quit Fdisk. Once you have zeroed a volume, however, it isn’t possible to return it to its original state.

If you enter a cylinder size which is too large, you are informed:

**ERROR:** Size specified is larger than available space.

In this case, you have to try again with a smaller cylinder size.
Utility Programs

Delete p–System Partition

This option allows you to remove a p–System partition on the selected fixed disk. (The command doesn’t affect any existing partitions which are allocated to other operating systems.) The space the deleted p–System partition occupied becomes free and can be reused. When you select this option, the following prompt is displayed:

Enter p–System partition number to be deleted ( Oasis to exit): [Enter]

You should enter the partition number that corresponds to the p–System volume you want to remove.

CAUTION: Be sure that you enter the correct partition number. If you have more than one p–System partition, you could accidentally lose valuable data by removing the wrong partition. (You may be able to recover the information by using the Create p–System Partition command to create a new partition on top of the old location; but don’t rely on this method.)

If you attempt to delete a partition that isn’t allocated to the p–System, you are informed:

ERROR: Partition # is not a p–System partition
Utility Programs

Backup p—System Partition

This command is used to back up the files in a p—System partition onto one or more floppy disks. The backup operation is performed on a file—by—file basis. If the current floppy disk becomes full, you are prompted to place a new floppy disk in the drive. All floppy disks must already be formatted. (However, they don't have to be initialized with the file's Z ero command.)

You can specify that all files in the partition should be backed up, that selected files should be backed up, or that files created on or after a certain date should be backed up.

If a file is larger than the floppy disk capacity or larger than the remaining floppy disk space, the backup option splits the file into segments which are placed on as many floppy disks as required. (Each segment of the file is given the same name as the file, itself.) The floppy disks are given the volume name BCKUP01 through BCKUP99.

A small file called BACKUP...RECORD is also placed on each floppy disk. This file contains the number of the partition being backed up, the name of the corresponding p—System volume, the date that the backup is performed, the number of files in the partition, the name of the file split across floppies, and a marker to indicate the last floppy disk. The bootstrap code is backed up as a file BOOT...RECORD on the first floppy disk.
Utility Programs

The following prompts are displayed by this option:

Enter partition number to be backed up (0-3 or a-%): a (enter)
Enter name of file to be backed up (CR for all files): FILE SPEC
Enter date to be backed up from: DATE SPEC
Enter destination flexible disk device number (4,5): 5 (enter)
Put a formatted diskette in device 4, press Enter when ready: (enter)

The first prompt asks you for the number of the partition that you want to back up. Be sure to select the partition that corresponds to the p—System volume you want to back up.

The second prompt asks you to specify the file(s) to be backed up. If you simply press <enter>, all of the files are backed up. If you enter a single file name, only that file is backed up (if it exists on the volume). You may, however, include an equal sign in the file name. This works the same as the filer's equal sign wild card. The "==" matches any string of characters. For example:

- .TEXT matches all files that end with .TEXT
- .CODE matches all files that end with .CODE
- SYSTEM= matches all files that start with SYSTEM
- SYS=CODE matches all files that start with SYS and end with CODE
- = matches all file names

Thus, entering ==.TEXT, backs up all the text files on the volume. Pressing == followed by <enter> is equivalent to simply pressing <enter> (all files are backed up).
Utility Programs

The third prompt asks for a date specification. You can enter a date or just press <enter>. If you enter a date, only files dated on or after that date are copied. If you simply press <enter>, the dates of the files aren't taken into consideration. A date is entered in the same way that it is when using the filer. For example:

1-JAN-84
4-JUL-76

Note that you also can select files using a combination of name and date.

The fourth prompt asks you to designate the floppy disk drive (#4 or #5) to be used for backing up the fixed disk. Don't enter a preceding number sign (#).

Finally, you are asked to place a formatted disk into the appropriate drive and press <enter>. 

9-28

1700101:09
Utility Programs

You are informed of each file that is backed up. When needed, you are also asked to insert a new floppy disk. The following is an example of the messages that you will receive:

Zeroing BCKUP01:
Creating back up record ...
Copying file VOLNAME:SYSTEM.PASCAL to BCKUP01:SYSTEM.PASCAL
Copying file VOLNAME:SYSTEM.EDITOR to BCKUP01:SYSTEM.EDITOR
Copying file VOLNAME:SYSTEM.FILER to BCKUP01:SYSTEM.FILER

Copying part of VOLNAME:MISC.FILE to BCKUP01:MISC.FILE
Put a formatted diskette in device 4, press Enter when ready:

Zeroing BCKUP02:
Creating back up record ...
Copying file VOLNAME:ANOTHER.TEXT to BCKUP02:ANOTHER.TEXT

Be sure to properly label your floppy disks so that you can remember the order in which they go. This is important if you decide to restore the floppies to a fixed disk partition (using the following Restore option).

Restore p—System Partition

The restore option complements the backup option. It is used to copy the floppy disk files created in the backup operation back onto the fixed disk partition.
Utility Programs

You may select the files to be restored by date and name. If a file exists on the fixed disk with the same name as one of the files being restored, it is replaced with the file being restored. Restore doesn't disturb other files that may already exist on the fixed disk volume.

The first floppy disk submitted to the restore option must be BCKUP01. You are prompted to insert any succeeding disks (BCKUP02... and so forth). Segments of a file that have been split during the backup process are rejoined into the original file.

If only a few files require restoration—and no files are split across a floppy disk—you may use the filer’s T(transfer) command, rather than Fdisk's restoring facility.

The following prompts are displayed by this option:

Enter partition number to be restored (Esc to exit): [Enter]
Restore boot record (Y/N)? [Enter]
Enter name of file to be restored (CR for all files); FILE SPEC [Enter]
Enter date to be restored from: DATE SPEC [Enter]
Enter device number of back up flexible disk (4,8): [Enter]
Put disk volume BCKUP01 in disk device #4, press Enter when ready:

The first prompt asks you for the number of the partition you want to restore. Be sure to select the partition that corresponds to the correct p—System volume.
Utility Programs

The second prompt asks if you want to restore the bootstrap code that originally existed on the volume. The bootstrap code only needs to be present on a boot disk. In general, you don’t have to restore this code. If you are restoring a bootable partition, and you have reason to believe that the old bootstrap code has been corrupted, you can restore it in this manner.

The third prompt asks you to specify the file(s) to be restored. If you simply press <enter>, all of the files are restored. If you enter a single file name, only that file is restored (if it exists on the backup floppy disks). You may include an equal sign in the file name as described previously under the backup option.

The fourth prompt asks for a date specification. You can enter a date or just press <enter>. If you enter a date, only files dated on or after that date are restored. If you simply press <enter>, the dates of the files aren’t taken into consideration. A date is entered in the same way that it is for the backup option (described previously).

As with the backup option, you may select files using a combination of name and date.

The fourth prompt asks you to designate the floppy disk drive (#4 or #5) from which files are to be restored. Don’t enter a preceding number sign (#).

Finally, you are asked to place BCKUP01 into the appropriate drive and press <enter>.
Utility Programs

You are informed of each file that is restored: and, when
needed, you are asked to insert the next floppy disk. The
following is an example of the messages you will receive:

```
Put disk volume BACKUP1 in disk device #4, press Enter when ready:
Reading backup record ...
Restore file BACKUP1:SYSTEM.PASCAL to VOLUME1:SYSTEM.PASCAL
Restore file BACKUP1:SYSTEM.EDITOR to VOLUME1:SYSTEM.EDITOR
Restore file BACKUP1:SYSTEM.FILER to VOLUME1:SYSTEM.FILER

Restore file BACKUP2:MISC.FILE to part of VOLUME2:MISC.FILE
Put disk volume BACKUP2 in disk device #4, press Enter when ready:
Reading backup record ...
Restore file BACKUP2:ANOTHER.TEXT to VOLUME2:ANOTHER.TEXT
```

Quit.

The quit option allows you to terminate the Fdisk utility. If
you made any changes related to the currently selected fixed
disk, you are asked:

```
Make changes permanent (update partition table) (Y/N)? Y or N
```

If you press Y, the fixed disk configuration changes you have
made are saved on disk. If you press N, those changes are
discarded. (As mentioned before, if you elected to zero a
partition, you can't undo that operation here, nor can you
undo a restore operation.)
Utility Programs

NCICOPY

Users of the Network Consulting Inc. (NCI) p-System may have disks that were formatted with 10 sectors per track (800 blocks). The 10 sector format used by NCI and the 10 sector format supported by this release are not compatible. In other words, this release of the p-System does not recognize NCI 10 sector format disks.

The NCICopy utility allows you to copy an NCI 10 sector format disk onto a p-System blocked volume. This volume may be a disk, RAM disk, or perhaps a subsidiary volume. The volume must contain at least 800 blocks.

An ideal way to use this utility is to format a 10 sector disk, then use NCICopy to copy the NCI disk to the newly formatted 10 sector disk.

First, execute the Configure utility (refer to Chapter 4) and set the desired destination drive to 10 sectors per track. Then execute the Diskformat utility to format a diskette in the destination drive. You now have a blank 10 sector (800 block) disk on which to copy your NCI disk.
Utility Programs

To copy your NCI disk, X(ecute "NCICopy"). This displays the following prompt:

```
This program copies an NCI 10 sector format disk
to a standard 10 sector format disk.

Enter device number of source diskette (NCI format):  
```

Respond to this prompt by entering the number of the drive that contains the NCI disk and then press <enter>.

```
Enter device number of destination:  
```

Enter the number of the blocked device where you wish to copy the NCI disk. There must be at least 800 blocks on the destination volume. If you have a newly formatted 10 sector disk, enter the number of the device where this disk resides. Make sure that the drive is presently configured to 10 sectors per track (use the Configure utility, if necessary). The source drive may be configured in any way.

The NCICopy utility then copies the data from the NCI 10 sector format disk to the destination volume. An error message is displayed if: (1) the source disk is not an NCI 10 sector format disk, (2) there is any problem in transferring the data, or (3) the destination volume contains less than 800 blocks.
Utility Programs

STARTUP

The Startup utility automatically performs certain operations which are desirable after booting the system. The utility can be programmed to transfer files to a RAM disk or to set other system options. This frees you from having to prepare your system for use each time you boot it.

Program File

The startup program must be installed on the boot disk as SYSTEM.STARTUP to be automatically executed when you boot the system. It may have another name (usually STARTUP.CODE); in which case, it isn't automatically executed every time you boot, but only when desired.

Units Needed

The startup program uses the "clockunit" in SYSTEM.LIBRARY. This unit obtains the time and date from DOS.

Command File

The startup program reads the file *LOADFILES.TEXT and interprets the command lines in the file. If the file is not found, the program ends with no action taken. Each command is one line, consisting of a command character within left and right square brackets, followed by the command parameters. The left bracket, command letter, and right bracket must be the first three characters of the line (no leading or embedded blanks). Any line that doesn't start with a left bracket is ignored. Each command is performed as the line is read—except the X command, where the command(s) occurs after the program ends.
Utility Programs

Command Summary

The available commands and a description of their operation are presented here. Where a VOLNAME appears in the syntax, you must substitute the appropriate volume name, which may be any legal p-System volume name followed by a colon. Where FILENAME appears in the syntax, you must substitute the appropriate file name, which may be any legal p-System file name and may include volume name if desired. No wild cards may be used in file names. EXECUTION OPTIONS may contain any execution option allowed by the chain command (the string is passed to chain)—such as a file name to execute, a redirection command, or both.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Operation Performed</th>
</tr>
</thead>
<tbody>
<tr>
<td>[*] VOLNAME</td>
<td>Change root volume to VOLNAME</td>
</tr>
<tr>
<td>[*] VOLNAME</td>
<td>Change prefix name to VOLNAME</td>
</tr>
<tr>
<td>[D] VOLNAME</td>
<td>Set system date and time from clockunit</td>
</tr>
<tr>
<td>[T] FILENAME1, FILENAME2</td>
<td>Transfer FILENAME1 to FILENAME2</td>
</tr>
<tr>
<td>[W] FILENAME</td>
<td>Mount subsidiary volume FILENAME</td>
</tr>
<tr>
<td>[F] VOLNAME</td>
<td>Use the SYSTEM.PASCAL found on VOLNAME</td>
</tr>
<tr>
<td>[E] FILENAME</td>
<td>Use FILENAME for SYSTEM.EDITOR</td>
</tr>
<tr>
<td>[C] FILENAME</td>
<td>Use FILENAME for SYSTEM.COMPILER</td>
</tr>
<tr>
<td>[A] FILENAME</td>
<td>Use FILENAME for SYSTEM.ASSMBLER</td>
</tr>
<tr>
<td>[L] FILENAME</td>
<td>Use FILENAME for SYSTEM.LINKER</td>
</tr>
<tr>
<td>[X] EXECUTION OPTIONS</td>
<td>Load EXECUTION OPTIONS into the execution queue (refer to the chain procedure)</td>
</tr>
</tbody>
</table>

9-34 1700101:09
Operating System Note

The [P] command is used to tell the p-System operating system to use the copy of SYSTEM.PASCAL on the volume specified. The copy you are switching to must be an exact copy of the SYSTEM.PASCAL you booted the system with. It's purpose is to enable a copy of the operating system to be loaded onto a RAM disk and to tell the operating system to use the copy on the RAM disk. The root volume need not be the same volume as the volume containing SYSTEM.PASCAL. The SYSTEM.PASCAL you are changing to must also begin on the same block number as the SYSTEM.PASCAL you booted with. This is done most easily by not having a duplicate directory on the boot disk and by placing SYSTEM.PASCAL on the disk first, thus placing it at block 8. Then transfer SYSTEM.PASCAL to RAM disk first, so it will be at block 8 on RAM disk; this allows the [P] command to work. Rather than copy SYSTEM.PASCAL to RAM disk and use the [P] command, it's better to use the Extrapools utility.

System Program Names

The [A],[C],[E],[F], and [L] commands are used to set the name of system programs and their volume location. These are originally set to *SYSTEM.ASMBLER, *SYSTEM.COMPLILER, *SYSTEM.EDITOR, *SYSTEM.FILER, and *SYSTEM.LINKER.
Utility Programs

Example of Command Lines

An example of the contents (with associated comments) of "LOADFILES.TEXT" is as follows:

[D] RAMDISK:
[D]#4:
[F]RAMDISK:
[F]SYSTEM.PASCAL, SYSTEM.PASCAL
[F]WORK:UTILS.SYNOL
[F]WORK:SYSTEM.SYNOL
[F]SYSTEM:SYSTEM.FILER, SYSTEM.FILER
[F]RAMDISK:SYSTEM.FILER
[X]SYSTEM:SYSTEM.EDITOR
[X]#4:LOGO
[X]=RAMDISK:

read date/time from clockunit
store system date on RAMDISK:
store system date on disk in #4:
change prefix volume to RAMDISK:
transfer SYSTEM.PASCAL to RAMDISK:
use SYSTEM.PASCAL on RAMDISK:
mount subsidiary volume
mount subsidiary volume
move flier to RAMDISK:
set flier to load from RAMDISK:
set editor to load from SYSTEM:
execute logo after this program
change root volume to RAMDISK:
CHAPTER 10

PROGRAMMER

INFORMATION
PROGRAMMER INFORMATION

GENERAL

If you plan to develop programs on the IBM Personal Computer, using the p-System, this chapter provides the information you will need. (The information should complement the Program Development Reference Manual and other p-System programming documentation.)

First, several units are described. These units provide routines that your programs may use. For example, the IBM_SPECIAL unit interfaces with the IBM PC hardware, and the TURTLEGRAPHICS unit allows your programs to display graphic images on the video display. For each utility program described in Chapter 9, a corresponding unit is available that allows your programs to provide the same services as the utility program does. All of these units are also described.

After this, several miscellaneous programming topics are covered, including: keyboard codes, events and interrupts, and video control codes.
IBM SPECIAL UNIT

The IBM_SPECIAL unit is a UCSD Pascal unit which your programs may use to interact with the IBM PC hardware.

Here is a listing of its interface section:

```pascal
unit IBM_SPECIAL;
interface
  type
    pitch_range = 0..12;
    octave_range = 0..7;
    onebit = 0..1;
    twomax = 0..2;
    twobits = 0..3;
    threebits = 0..7;
    fourbits = 0..15;
    font_pattern = packed array [0..63] of boolean;
    font_table = array [128..255] of font_pattern;
    font_ptr = "font_table;"
    str_ptr = "string;"
    key_table = array [1..16] of str_ptr;
    key_ptr = "key_table;"

  function button (select: twobits): boolean;
  procedure paddle (select: twobits; var result: integer);
  procedure note (pitch: pitch_range;
                  octave: octave_range;
                  duration: integer);
  function lightenup (var charxpos: charxpos;
                    charypos: charypos;
                    pixelxpos: pixelxpos; pixelypos: integer): boolean;
  procedure setkeys (tableptr: key_ptr);
  procedure video_mode (mode: threebits);
  procedure setfont (table: font_ptr);
  procedure bgnd_color (color: fourbits);
  procedure palette (color: onebit);
  procedure settime (var hour: minute; integer);
  procedure gettime (var hour: minute; integer);
  procedure blink (ann: boolean);
  procedure highlight (ann: boolean);
  procedure foreground (color: threebits);
  procedure background (color: threebits);
  procedure color_page (pagenum: threebits);
  procedure tocolor;
  procedure tomono;
  procedure set_attribute (ch: char);
  procedure read_attribute (var ch: char);
  procedure reverse_video;
  procedure settimes (var hour: minute; second: integer);
  procedure gettimes (var hour: minute; second: integer);
  function random : real;
  procedure randomize;
```
Function Button returns a Boolean value of true, if the selected game button is pressed. The switches are numbered from 0 through 3.

Procedure PADDLE returns the position of the selected game paddle in the parameter result.

Procedure NOTE plays a note of the specified pitch and octave for the given duration, via the built-in speaker.

The pitch parameter is specified as follows:

<table>
<thead>
<tr>
<th>0 rest</th>
<th>1 - C</th>
<th>2 - C#</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 - D</td>
<td>4 - D#</td>
<td>5 - E</td>
</tr>
<tr>
<td>6 - F</td>
<td>7 - F#</td>
<td>8 - G</td>
</tr>
<tr>
<td>9 - G#</td>
<td>10 - A</td>
<td>11 - A#</td>
</tr>
<tr>
<td>12 - B</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The octave parameter is defined as:

| 0 - Lowest | 7 - Highest |

The duration parameter is defined in milliseconds as an unsigned integer.
Function Lightpen returns true, if the lightpen is activated. It returns the cursor position of the lightpen as follows:

charxpos is the X position, 0..79
charypos is the Y position 0..24
pixelpos is the pixel X position, 0..539
pixelypos is the pixel Y position, 0..199

Procedure SETKEYS defines an alternate way of using the 16 function keys described in this chapter. The parameter passed is a pointer to a table of 16 pointers to strings. When the function key is depressed, the string associated with that element of the table is returned via console read. This continues until the program is terminated.

Procedure VIDEOMODE sets the mode of the BIOS video driver. Video 0 through 3 are low resolution modes, while modes 4 through 5 are graphics modes. Video modes 0 through 5 are defined as follows:

0 - 40x25 BW  1 - 40x25 Color
2 - 80x25 BW  3 - 80x25 Color
4 - 220x200 Color  5 - 320x200 BW
6 - 640x200 BW
Programmer Information

Procedure SETFONT allows you to redefine the upper 128 ASCII characters (80..FF). SETFONT is passed a pointer to a table. The table contains 128 entries (128 through 255), each of which is an array of bits (0 through 63). This boolean array defines a character. The bits are arranged in the following pattern (each entry represents a single pixel):

```
7  ...  0
15 ...  8
23 ... 16
31 ... 24
39 ... 32
47 ... 40
55 ... 48
63 ... 56
```

This use of the font applies to the graphics mode, NOT the alphanumeric mode. The definitions disappear when the program that called SETFONT terminates.

Procedure BKGND_COLOR sets the background color in the graphic mode. It sets the border color in the alphanumeric mode. Parameter color may take the values:

```
0 = Black
1 = Blue
2 = Green
3 = Cyan
4 = Red
5 = Magenta
6 = Yellow
7 = White
8 = High Intensity Black
9 = High Intensity Blue
10 = High Intensity Green
11 = High Intensity Cyan
12 = High Intensity Red
13 = High Intensity Magenta
14 = High Intensity Yellow
15 = High Intensity White
```
Programmer Information

Procedure PALETTE sets the colors to be used in the graphics mode (mode 4). If parameter color is 0, the following colors are produced:

0 = Background Color
1 = Green
2 = Red
3 = Yellow

If parameter color is 1, the following colors are produced:

0 = Background Color
1 = Cyan
2 = Magenta
3 = White

These are the colors that will be indicated by a variable of type color within the Turtlegraphics unit (see "Turtlegraphics" in this chapter).

Procedure SETTIME sets the clock to correspond to the indicated hour and minute.

Procedure GETTIME gets the time of day. This procedure returns the time since booting, if SETTIME hasn't been called first. After rebooting, SETTIME must be called again, if GETTIME is to return valid information.

Procedure BLINK turns blinking on and off. When blinking is on, anything subsequently sent to the video blinks. When blinking is turned off, subsequent material displayed on the video appears normally.
Procedure HIGHLIGHT turns highlighting on and off. When highlighting is on, anything subsequently sent to the video appears brighter than normal.

Procedure FOREGND allows you to set the foreground color. Parameter color may take on the same set of values as listed for Procedure BKGND _COLOR.

Procedure BACKGND allows you to set the background color in alphanumeric mode. Parameter color may be set to the first eight values listed for Procedure BKGND _COLOR. High intensity—which is actually a foreground attribute—and border color are not affected.

Procedure VIDEO _PAGE allows you to switch between the eight video pages that exist in memory when low resolution (modes 0–3) is being used. A call to VIDEO _PAGE, with the specified page number, will switch between these video pages. Any writing done to the screen while in a specified page will write over previous characters and be stored as part of the new page.

When in the 80 column mode, there are only 4 video pages (0–3) available. All 8 pages can be accessed in only the 40 column mode.

NOTE: VIDEO _PAGE won’t work correctly with the sliding—80 configuration of the 40 column mode.

Procedures TOCOLOR and TOMONO can be used to switch between a color monitor card and a monochrome monitor card (if you have both cards installed). These are analogous to the special keys <Alt> L and <Alt> M.
Procedure SET_ATTRIBUTE changes the screen attribute character. The new attribute will remain in effect until changed by another call to SET_ATTRIBUTE. (The screen attribute character, which is described in the IBM Technical Reference Manual, sets the mode used—reverse video, highlighted, and so forth—when characters are written to the screen.) READ_ATTRIBUTE returns the current screen attribute character.

When a call is made to REVERSE_VIDEO, the current cursor position and its subsequent locations are put into reverse video mode. This mode continues until the next call is made to REVERSE_VIDEO.

Procedures GETTIMES and SETTIMES are identical to GETTIME and SETTIME, but they also support seconds.

Function RANDOM returns a random real value which is greater than or equal to 0.0 and less than 1.0.

Procedure RANDOMIZE gives RANDOM a new seed based upon the clock value. RANDOMIZE should be called once before you begin to make calls to RANDOM.
TURTLEGRAPHICS

The TURTLEGRAPHICS unit allows programs to produce graphics on the video display (if the display has graphics capability). Turtlegraphics is described in the Program Development Reference Manual. However, this section covers a few important points about the IBM PC adaptation of Turtlegraphics.

CAUTION: If your program is written to use version IV.0 of Turtlegraphics, it must be modified for IV.1 Turtlegraphics to run. Beginning with version IV.1, the ALIGN_CURSOR procedure in version IV.0 has been replaced with the WCHAR AND WSTRING procedures.

The SYSTEM.LIBRARY distributed with the current release of the p-System contains the 4-word TURTLEGRAPHICS unit for the current release. The 2-word TURTLEGRAPHICS unit is in a separate file. (The section, "Using Turtlegraphics," presented in Chapter 8, discusses this from a non-programmer user's point of view.)
The interface section for the IV.1 (and later) version of Turtlegraphics is shown here.

```plaintext
unit turtlegraphics;

interface

procedure display_scale( min_x, min_y, max_x, max_y : real );
function aspect_ratio : real;
function create_figure( x_size, y_size : real ) : integer;
procedure delete_figure( screen : integer );
procedure viewport( min_x, min_y, max_x, max_y : real );
procedure fillscreen( screen : integer; shade : integer );
procedure background( screen : integer; shade : integer );
function read_pixel( screen : integer; x, y : real ) : integer;
function set_pixel( screen : integer; x, y : real; shade : integer );
procedure getfigure( source_screen : integer;
    corner_x, corner_y : real;
    copymode : integer );
procedure putfigure( destination_screen : integer;
    corner_x, corner_y : real;
    copymode : integer );
function read_figure_file( title : string ) : integer;
function write_figure_file( title : string ) : integer;
function load_figure( index : integer ) : integer;
function store_figure( screen : integer ) : integer;
procedure activate_turtle( screen : integer );
function turtle_x : real;
function turtle_y : real;
function turtle_angle : real;
procedure move( distance : real );
procedure moveto( x, y : real );
procedure turn( rotation : real );
procedure turns( direction : real );
procedure pen_mode( state : integer );
procedure pen_color( shade : integer );
procedure pen_char( c : char; copymode, shade : integer );
procedure write_string( s : string; copymode, shade : integer );
```

10-12 1700101:10
PRNCONTROL UNIT

The PRNCONTROL unit is used by the Configure utility. This unit is provided so that any program may configure an EPSON Matrix Printer. (It is distributed as the file, EPSON.CODE.) Chapter 4, "Configure Utility," provides an example, under "Matrix Printer," explaining how the unit is used. Here is a listing of the interface section for the PRNCONTROL unit:

```
UNIT PRNCONTROL;
INTERFACE
  Procedure Get_Printer_Title ( var title: string );
  Procedure Prn_CPL ( chars_per_line: integer );
  Procedure Prn_LPI ( lines_per_inch: integer );
  Procedure Prn_FormLen ( formlength: integer );
  Procedure Prn_HorTab ( tabstring: string );
  Procedure Prn_VertTab ( tabstring: string );
  Procedure PrintConfig;
```

Procedure GET(printer_title) returns in the parameter TITLE the string "EPSON Matrix."

The procedure PRN_CPL sets the maximum number of characters that can be printed on a line (80 or 132). If the parameter CHAR_PER_LINE is 132, then the printer is set to 132 characters per line; if CHAR_PER_LINE is any other integer, the printer is set to 80.

PRN_LPI sets the lines per inch (vertical spacing) for the printer. The parameter LINES_PER_INCH may be 6, 8, or 10; if another integer is passed, the default will be 6.

The parameter FORMLENGTH in the procedure PRN_FORMLEN may be set to any value from 1 to 127. This determines the number of lines on a printed page.
Programmer Information

PRN_HORTAB and PRN_VERTTAB allow you to set the positions for tab stops within a line (horizontally) or for a printed page (vertically). The TABSTRING parameter contents are explained in the section on the matrix printer under the Configure utility program.

A call to procedure PRINTCONFIG sets up a menu that allows a user to change the printer to a desired configuration. The options for the menu are described in the previous six routines.
FORMATTER UNIT

The FORMATTER unit is used by the Diskformat utility. This unit is available so that any program can format disks. Here is a listing of the interface section for the FORMATTER unit:

```pascal
unit Formatter;

interface

procedure formatdisk( device_num, sec_den_8_9 : integer;
    var result, err_track, err_sector, error_side : integer;
    display : boolean);
```

The FORMATDISK procedure formats the disk in the drive with device number DEVICE_NUM.

The formatting is done using the parameters for that device (as set using the Configure utility). If the auto-recognition of 8 or 9 sectors per track is on, then the variable SEC_DEN_8_9 must indicate which sector density to use. This variable should be set to either 8 or 9.

The RESULT parameter is zero if the format was successful. Otherwise, it is set to the appropriate diskette status as defined in the IBM PC Technical Reference manual.

If there is an error, ERR_TRACK, ERR_SECTOR, and ERR_SIDE indicate the problem location on the disk.
Programmer Information

If DISPLAY is true, a dot for each track formatted is displayed, as follows:

Formatting side 0

Formatting side 1
FDISK UNIT

The FDISK unit allows programs to perform the same sort of operations as the Fdisk utility. Here is the interface section for this unit:

```pascal
unit FDISK_UNIT;

interface

type
FD_op_seq_type = (unknown, DSK.CPN, pdSystem);
FD_part_entry = record
  Boot: boolean;
  op_seq: FD_op_seq_type;
  cyl_name: string[7];
  st_cyl, end_cyl: integer;
end;
FD_part_status = record
  entry: array[1..4] of FD_part_entry;
  from_max_cyl, from_st_cyl, num_part, drive: integer;
end;
FD_error_type = (FD_ok, invalid_part, non_exam_drive, name_err, max_part,
  max_parts, (FD_create_part - cyl_size) max, (FD_create_part - 4 partitions,
  already exists), not_found, (FD_restore, FD_restore - File not found),
  FD_offline, (FD_backup, FD_restore - Volume is offline),
  name_err, (FD_backup, FD_restore - Error in file name),
  FD_other, (FD_backup, FD_restore - Unknown),
  s_dir, (FD_backup - Error in Zeroling directory of backup directory),
  boot_read, (FD_backup - Error in boot read),
  boot_write, (FD_backup - Error in boot write),
  open_source, (FD_backup, FD_restore - Cannot open source file),
  transfer, (FD_backup, FD_restore - Error during transfer of file),
  switch_file, (FD_backup - Error in transferring switch file),
  part_full); (FD_restore - no room on volume or partition)
```

1700101:10 10-17
Programmer Information

```plaintext
Var
  FD_Part_info : FD_Part_status;

Procedure FD_read_part_tab(FD_drive : integer);
  var result : FD_Error_type;

Procedure FD_save_part_tab(var result : FD_Error_type);

Function FD_check_Boot(part_num : integer): boolean;

Procedure FD_set_Boot_part(part_num : integer);

Function FD_Create_part(xyl_size : integer): integer;

Procedure FD_Zero_part_dir(part_num : integer);
  vol_name : string; var result : FD_Error_type;

Procedure FD_Delete_part(part_num : integer);

Procedure FD_Backup(part_num, floppy : integer);
  backup_file_name, file_data : string;
  var result : FD_Error_type;

Procedure FD_Restore(reset_boot: boolean);
  part_num, floppy : integer;
  res_file_name, file_data : string;
  var result : FD_Error_type;
```

The "partition table" is an area on the fixed disk that contains information about the partitions on that disk. (The information includes the number of partitions, their locations, and so forth.) This unit can read the partition table from a fixed disk, alter the local copy of that table, and write it back out. The local copy of the partition table is stored in the variable FD_PART_INFO.

**NOTE:** You may read the information in the FD_PART_INFO variable directly, but you should never write directly to that variable. All updating of the variable should be done through the routines provided in this unit. This unit won't function correctly if you update FD_PART_INFO directly.
Programmer Information

FD_READ_PART_TBL reads the partition table information from the fixed disk designated by FD_DRIVE. The partition table information is placed in the variable FD_PART_INFO. FD_DRIVE should be 1 for the first fixed disk, 2 for the second, and so on.

FD_SAVE_PART_TBL places the local copy of the partition table back out on the fixed disk. The partition table is automatically saved on the fixed disk from which it was read (by the FD_READ_PART_TBL procedure).

FD_CHECK_BOOT checks to see if a bootstrap resides on the partition designated by PART_NUM. PART_NUM should be set to 1 through 4; it designates the corresponding partition. If the partition is a p–System partition, FD_CHECK_BOOT verifies that the bootstrap is a p–System bootstrap. If the partition does contain a correct bootstrap, the function returns true; otherwise, it returns false.

FD_SET_BOOT_PART designates the partition PART_NUM as the bootable partition (in the local copy of the partition table). PART_NUM should be 1 through 4 to designate the corresponding partition as bootable. If PART_NUM is set to 0, all partitions are designated as non-bootable.
Programmer Information

FD_CREATE_Part creates a new p-System partition (in the local copy of the partition table). The new partition is given the size indicated by CYL_SIZE (in cylinders). The partition numbers assigned to the existing partitions may change depending upon where the new partition is created. The partition number assigned to the new partition is returned as the function value.

FD_ZERO_Part_Dir creates an empty p-System directory on the partition designated by PART_NUM. (This must be a p-System partition.) The partition is given the volume name VOL_NAME. This operation is done to the fixed disk itself (not to any local variable). Once you have initialized a directory, you cannot return the directory to its previous state.

FD_DELETE_Part removes the partition indicated by PART_NUM (in the local partition table). This must be a p-System partition.

FD_BACKUP backs up the partition indicated by PART_NUM. FLOPPY should be a device number such as 4 or 5; it indicates the drive where the fixed disk will be backed up. BKUP_FILE_NAME indicates the file(s) from the fixed disk to backup; it may contain the equal sign wild card; or it may be empty, which implies all files. FILE_DATE indicates that any files created on or after that date should be backed up: it may be empty which means to ignore the date. (FILE_DATE should have the form dd—mmm—yy; for example, "1—Jan—84." It does not matter whether uppercase or lowercase letters are used.) This routine displays the same information as the Fdisk utility when the backup operation is used. (That is, it shows which files are being backed up, prompts the user to place a new disk in the drive, and so forth.) For more information about how this routine works, refer to the description of the backup operation under the Fdisk utility.
FD_RESTORE complements the FD_BACKUP operation. All of the parameters have the same meaning as they do for the FD_BACKUP routine, except that the files are moved from the floppy disk to the fixed disk. There is an additional parameter, REST_BOOT. This indicates whether or not you want the bootstrap code that was taken from the partition to be restored to it.
Programmer Information

KEYBOARD CODES

This section covers the codes returned by some of the keys on the IBM PC keyboard.

The function keys return the following code sequences:

<table>
<thead>
<tr>
<th>Function Key</th>
<th>ASCII Code Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>DC1 a</td>
</tr>
<tr>
<td>F2</td>
<td>DC1 b</td>
</tr>
<tr>
<td>F3</td>
<td>DC1 c</td>
</tr>
<tr>
<td>F4</td>
<td>DC1 d</td>
</tr>
<tr>
<td>F5</td>
<td>DC1 e</td>
</tr>
<tr>
<td>F6</td>
<td>DC1 f</td>
</tr>
<tr>
<td>F7</td>
<td>DC1 g</td>
</tr>
<tr>
<td>F8</td>
<td>DC1 h</td>
</tr>
<tr>
<td>F9</td>
<td>DC1 i</td>
</tr>
<tr>
<td>F10</td>
<td>DC1 j</td>
</tr>
</tbody>
</table>

The indicated code sequences are sent to the console when one of these function keys is pressed. A program may assign any desired meaning to these keys. The function keys may be set to return string values instead of the indicated code sequences. This can be done using the SETKEYS procedure within the IBM _SPECIAL unit (described at the the beginning of this chapter). When this is done, pressing one of these keys echoes the corresponding string to the console.
Programmer Information

The following keys on the number pad represent function keys 11 through 16 and may also be defined by procedure SETKEYS:

<table>
<thead>
<tr>
<th>Function Key</th>
<th>ASCII Code Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>11 = Home</td>
<td>DC1 k</td>
</tr>
<tr>
<td>12 = PageUp</td>
<td>DC1 l</td>
</tr>
<tr>
<td>13 = End</td>
<td>DC1 m</td>
</tr>
<tr>
<td>14 = PageDown</td>
<td>DC1 n</td>
</tr>
<tr>
<td>15 = INS</td>
<td>DC1 o</td>
</tr>
<tr>
<td>16 = DEL</td>
<td>DC1 p</td>
</tr>
</tbody>
</table>

At the keyboard, you may enter any 1-byte value from 1 to 255. Do this by depressing the `<Alt>` key and entering the decimal value (base 10) with the numeric keypad, and then releasing `<Alt>`. If you enter a number larger than 255, it is used as that number modulo 255.

In normal operation, pressing some keys at the keyboard perform special functions and cannot be read by a program. For example, the key combinations `<Alt>` L and `<Alt>` M toggle between color and monochrome monitors. A few keys are simply "dead" and produce no action.

The p-System can be configured so that a program may have access to all key combinations on the keyboard. This is referred to here as the "absolute" keyboard mode.
Programmer Information

To set this mode, execute the Configure utility and select the Change Driver Parameters option. When the next menu appears, select the User-Defined Driver option to change. Enter 1 for the device number (console). Parameter 1 of the console driver parameters should be modified as follows.

To set the absolute keyboard mode, perform a boolean OR of the current value with 4 (that is, turn on the third bit). For example, if the current value for the first parameter byte is 1, set the new value to 5.

In absolute keyboard mode, a keystroke that would generate a normal ASCII character returns that character to the program. However, a special key (most combinations of Ctrl> and Alt> keys as well as the function keys) generate a two-character sequence. The first character returned is the keyboard prefix character (DC1). The second character returned is the special keyboard function. These special functions are the "extended keyboard functions" as described in the "Keyboard Encoding" section of the IBM Personal Computer Technical Reference manual.

The following are examples of using the absolute keyboard mode:

<table>
<thead>
<tr>
<th>Keyboard input</th>
<th>Keysecn code(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Alt&gt; 0</td>
<td>16,16</td>
</tr>
<tr>
<td>&lt;Backspace&gt;</td>
<td>8</td>
</tr>
<tr>
<td>&lt;Alt&gt; 2</td>
<td>16,121</td>
</tr>
<tr>
<td>A</td>
<td>81</td>
</tr>
<tr>
<td>&lt;Ctrl&gt;(PgDn)</td>
<td>16,118</td>
</tr>
</tbody>
</table>
VIDEO CONTROL CODES

The following set of video control codes is available:

<table>
<thead>
<tr>
<th>ASCII Code</th>
<th>Hexadecimal</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESC &lt;0..6&gt;</td>
<td>1B &lt;30..35&gt;</td>
<td>Set video mode 0..6</td>
</tr>
<tr>
<td>ESC A</td>
<td>1B 41</td>
<td>Move cursor up</td>
</tr>
<tr>
<td>ESC C</td>
<td>1B 43</td>
<td>Move cursor right</td>
</tr>
<tr>
<td>ESC E</td>
<td>1B 45</td>
<td>Erase screen</td>
</tr>
<tr>
<td>ESC M</td>
<td>1B 46</td>
<td>Move cursor home (which is row 0, column 0)</td>
</tr>
<tr>
<td>ESC J</td>
<td>1B 4A</td>
<td>Erase to end of screen from cursor position</td>
</tr>
<tr>
<td>ESC K</td>
<td>1B 4B</td>
<td>Erase to end of line from cursor position</td>
</tr>
<tr>
<td>ESC L</td>
<td>1B 4C</td>
<td>Erase cursor line</td>
</tr>
<tr>
<td>ESC S</td>
<td>1B 53</td>
<td>Erase cursor line, scroll rest of screen</td>
</tr>
<tr>
<td>ESC T &lt;byte&gt;</td>
<td>1B 64 &lt;byte&gt;</td>
<td>Set attribute byte to &lt;byte&gt;</td>
</tr>
<tr>
<td>ESC Y &lt;bl&gt; &lt;bl&gt;</td>
<td>1B 66 &lt;bl&gt; &lt;bl&gt;</td>
<td>Move cursor to row &lt;bl&gt; - 32 column &lt;bl&gt; - 32</td>
</tr>
<tr>
<td>FF</td>
<td>0C</td>
<td>Form feed - clear screen</td>
</tr>
<tr>
<td>NAK &lt;byte&gt;</td>
<td>1B &lt;byte&gt;</td>
<td>Set video page to &lt;byte&gt;</td>
</tr>
<tr>
<td>ETB &lt;byte&gt;</td>
<td>17 &lt;byte&gt;</td>
<td>Set palette to &lt;byte&gt;</td>
</tr>
<tr>
<td>CAM &lt;byte&gt;</td>
<td>18 &lt;byte&gt;</td>
<td>Set border color to &lt;byte&gt;</td>
</tr>
<tr>
<td>BM &lt;0..6&gt;</td>
<td>19 &lt;0..6&gt;</td>
<td>Set video mode to 0..6</td>
</tr>
<tr>
<td>FS &lt;byte&gt;</td>
<td>1C &lt;byte&gt;</td>
<td>Set foreground to &lt;byte&gt;</td>
</tr>
<tr>
<td>RS &lt;byte&gt;</td>
<td>1E &lt;byte&gt;</td>
<td>Set background to &lt;byte&gt;</td>
</tr>
</tbody>
</table>

The video modes, 0 through 6, are defined as follows:

0 - 40x25 BW
1 - 40x25 16 Colors
2 - 80x25 BW
3 - 80x25 16 Colors
4 - 320x200 4 Colors
5 - 320x200 BW
6 - 640x200 BW
Programmer Information

The attribute byte is defined as follows (0 is the least significant byte):

<table>
<thead>
<tr>
<th>Bits</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-2</td>
<td>Foreground color</td>
</tr>
<tr>
<td>3</td>
<td>Highlight</td>
</tr>
<tr>
<td>4-6</td>
<td>Background color</td>
</tr>
<tr>
<td>7</td>
<td>Blink</td>
</tr>
</tbody>
</table>

The p-System may be configured so that the special video control codes just described are disabled, and all characters written to the console device are displayed on the screen. This is referred to as the "absolute video mode." These "absolute" characters range from special graphic characters, such as musical notes and happy faces, to greek letters and special mathematical notations. A definition of the "absolute" screen characters can be found in the IBM Personal Computer Technical Reference manual.

To configure the p-System to the absolute video mode, X(ecute the Configure utility and select the Change Driver Parameters option. When the next menu appears, select the User-Defined Driver option and enter 1 for the device number (console). Parameter 1 of the console device parameters should be modified as follows.

To set the absolute video mode, perform a boolean OR of the current value with 2 (that is, turn on the second bit). For example, if the current value is 1, set the new value to 3.
APPENDIX
APPENDIX A
I/O RESULTS

0  No error, operation successful
1  Bad block, Parity error (CRC)
2  Bad device number
3  Illegal I/O request
4  Data communications timeout
5  Volume is no longer on-line
6  File is no longer in directory
7  Bad file name
8  No room, insufficient space on volume
9  No such volume on-line
10  No such file on volume
11  Duplicate directory entry
12  Not close: attempt to open an open file
13  Not open: attempt to access a closed file
14  Bad format: error in reading real or integer
15  Queue overflow
16  Volume is write-protected
17  Illegal block number
18  Illegal buffer address (odd or illegal address)
19  Illegal size
20  Past end of file
21  Data size error
22  Corrupt tree
23  Duplicate key
24  No current record
25  Key size error
26  Illegal operation
27  Not available
28  Illegal file type
29  No directory
30  Network transmission
31  Network time-out
128  Too many open files
129  Invalid function number
130  DOS file not found
131  Path not found
Appendix A

132 Too many open files
133 Access denied
134 Invalid handle
135 Memory control blocks destroyed
136 Insufficient memory
137 Invalid memory block address
138 Invalid environment
139 Invalid format
140 Invalid access code
141 Invalid data
142
143 Invalid drive specified
144 Attempted to remove the current directory
145 Not same device
146 No more files
INDEX

- A -
absolute keyboard mode ......................................... 10-23
absolute video mode ............................................ 10-26
active port, printer .............................................. 4-13
active port (serial), Config .................................... 4-21
AsciiToStr ....................................................... 5-20

- B -
backing up a partition ......................................... 9-24
backup procedures
fixed disk backup ............................................... 1-6
floppy disk backup ........................................... 1-4
general ......................................................... 1-4
blockread ....................................................... 5-6
blockwrite ....................................................... 5-7
bootable partition, fixed disk ................................ 9-14
boot disks, creating ........................................... 2-7
bridge driver .................................................... 6-6
B(uild) .......................................................... 3-5
bytes per sector, Config ....................................... 4-17

- C -
CALLDOS unit .................................................. 5-18
C(hange) .......................................................... 3-7
change name ..................................................... 5-8
changing driver parameters ................................... 4-6
40 character displays .......................................... 8-6
characters per line (printer), Config ....................... 4-37
CLOCK unit .................................................... 5-21
close ............................................................. 5-7
color
 graphics mode ................................................ 10-8
setting background ........................................... 10-7, 10-9

1700101:01A
Index

- setting foreground ........................................... 10-9
- color/monochrome switching ................................. 8-6, 10-9
- color selection, Configure .................................. 4-11
- command menu ............................................... 2-6
- configuration unit, device drivers ......................... 5-22
- Configure ....................................................... 4-3
- configuring peripherals ........................................ 4-3
- configuring the fixed disk ..................................... 9-13
- Console configuration ......................................... 4-9
- console driver ............................................... 6-10
- console status records ......................................... 6-16
- control characters .............................................. 3-12
- conversion ....................................................... 3-11, 5-13
- creating a boot volume ......................................... 2-7
- creating a partition, fixed disk ............................... 9-20
- cylinder access (disk), Config ............................... 4-17
- cylinders, fixed disk .......................................... 9-14

- D -

- DD_FreeSpace .................................................. 5-11
- DD_GetInfo ..................................................... 5-11
- DD_StartSearch .................................................. 5-11
- default directory ............................................... 3-17, 5-8
- deleting a partition, fixed disk ............................... 9-23
- device drivers
  - console driver ............................................... 6-10
  - fixed disk driver ............................................ 6-11
  - floppy disk driver .......................................... 6-12
  - machine specific ............................................ 6-10
  - serial/parallel driver ....................................... 6-13
- device driver, writing .......................................... 7-3
- device index ..................................................... 5-26
- device number allocation, fixed disk ....................... 9-15
- directories, DOS, access to .................................. 5-10
- directory listing ............................................... 3-9, 3-15, 5-13
- Diskformat ....................................................... 9-3, 9-5
- disk FORMATTER unit .......................................... 10-15
- disks, backing up .............................................. 1-4
Index

Dismount .......................... 3–8
DLE Character .......................... 3–12
dos_blk_read .......................... 5–6
dos_blk_write .......................... 5–7
dos_close .......................... 5–7
DOSDIRINFO unit .......................... 5–10
DOS Filer .......................... 3–3
dos_flush .......................... 5–8
DOS_Function .......................... 5–19
dos_prefix .......................... 5–8
dos_rename .......................... 5–8
dos_reset .......................... 5–5
dos_rollback .......................... 5–5
dos_rotate .......................... 5–5
dos_scan_name .......................... 5–14
dos_scan_title .......................... 5–9
driver parameters, SYSTEM.CONFIG .......................... 4–6
DRVCONFIG (driver configuration) unit .......................... 5–22

- E -

End of File .......................... 3–12
error messages, during execution .......................... 2–5
events .......................... 10–27
ExitToDOS .......................... 5–20
E(xt dir) .......................... 3–9
extended memory .......................... 8–8
Extrapools .......................... 9–3
Extrapools utility
purpose .......................... 9–11

- F -

Fdisk .......................... 9–3, 9–13
Fdisk unit .......................... 10–17
FILEIO unit .......................... 5–4
FILERUNIT unit .......................... 5–12
first p–System track, Config .......................... 4–18
FIRST SUBSIDIARY VOL NUMBER .......................... 4–37, 5–27
fixed disk configuration .......................... 9–13

1700101:01A 1–3
Index

fixed disk driver ........................................ 6-11
fixed disk unit ........................................... 10-17
flags ......................................................... 3-9
floating point PMEs ........................................ 2-7
floppy disk configuration ................................ 4-15
floppy disk driver ......................................... 6-12
flush ......................................................... 5-8
FORMATTER unit ........................................... 10-15
formatting disks .......................................... 9-5
form length (printer). Config ............................ 4-38
free space. on DOS volume ............................... 5-11
F(rom DOS) .................................................. 3-11
function keys ............................................. 10-6, 10-22

— G —
game button ............................................... 10-5
game paddle ............................................... 10-5
graphics adapter ......................................... 4-10

— H —
hardware requirements .................................... 8-3
H(elp) ......................................................... 3-14
highlight .................................................... 10-9
horizontal synchronization, TV .......................... 4-12
horizontal tabs (printer). Config ....................... 4-38

— I —
IBM SPECIAL unit ......................................... 10-4
installation procedures
  copying onto a fixed disk ............................... 1-9
  copying onto a p-System system disk ............... 1-9
  general .................................................... 1-7
  Universal Medium disks to DOS disks ............... 1-7
Interface section ......................................... 5-4
interleave (disk). Config ................................ 4-18
interrupts .................................................. 10-27
Index

- K -

keyboard codes ............................................. 10-22

- L -

lightpen .................................................... 10-6
line feed .................................................... 3-12
lines per inch (printer), Config ......................... 4-38
L(list dir) ................................................... 3-15
low level I/O ................................................ 6-15

- M -

making a boot volume ........................................ 2-7
matrix printer configuration ............................... 4-37
MAX NUMBER OF SUBSIDIARY VOLS ....................... 5-27
memory allocation .............................................. 6-7
menus and prompts .......................................... 2-8
monochrome adapter ........................................... 4-10
Mount ......................................................... 3-16
mounting virtual volumes .................................... 2-4
MS-DOS device drivers
    detailed description .................................... 6-3
MS-DOS—Hosted p-System
    components ............................................... 1-3
    executing ............................................... 2-3
    executing, mounting virtual volumes .................. 2-4
    executing, with prefixed directory ................... 2-4
    musical note production .................................. 10-5

- N -

NCICopy ..................................................... 9-3
NCICopy utility ............................................. 9-31
number of sides (disk), Config .......................... 4-17

1700101:0IA
Index

- O -
open .................................................. 5-5
options ............................................. 3-12, 5-14

- P -
partitions, fixed disk ................................ 9-14
partition table ....................................... 10-18
peripheral configuration ............................... 4-3
8087 FME ............................................. 2-7
port, printer ......................................... 4-13
P(refix ............................................... 3-17, 5-8
printer configuration ................................ 4-13
printer, port configuration .......................... 4-13
printer status records ................................ 6-17
prompts and menus ................................... 2-6
PSYSTEM.COM program ................................ 1-3
PSYSTEM.VOL ......................................... 1-4

- Q -
Q(uit ........................................................ 3-18

- R -
RAM disk ............................................... 6-9
RAM disk configuration ................................ 4-18
random number generation ............................ 10-10
real arithmetic size .................................... 2-7
Remote driver, Config ................................. 4-20
R(emove File .......................................... 3-19
removing a partition, fixed disk ...................... 9-23
rename .................................................. 3-7, 5-8
reset ...................................................... 5-5
restoring a partition .................................. 5-27
reverse video mode ..................................... 10-10
rewind ................................................... 5-5
rewrite ................................................... 5-5
###_Index_

---

<table>
<thead>
<tr>
<th>S</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>scan name</td>
<td>5–14</td>
</tr>
<tr>
<td>scan title</td>
<td>5–9</td>
</tr>
<tr>
<td>screen attribute character</td>
<td>10–10</td>
</tr>
<tr>
<td>sectors per track, Config</td>
<td>4–16</td>
</tr>
<tr>
<td>serial/parallel driver</td>
<td>6–13</td>
</tr>
<tr>
<td>skew (disk), Config</td>
<td>4–18</td>
</tr>
<tr>
<td>special keys</td>
<td>8–4</td>
</tr>
<tr>
<td>Startup</td>
<td>9–4</td>
</tr>
</tbody>
</table>

**Startup utility**

- CLOCK unit | 9–33 |
- command file | 9–33 |
- command lines | 9–33 |
- command lines, example | 9–36 |
- command summary | 9–34 |
- *LOADFILES.TEXT | 9–33 |
- program file | 9–33 |
- purpose | 9–33 |
- setting system program names | 9–35 |

**step rate, Config** | 4–17 |

**StrToAscii** | 5–20 |

**subsidiary volumes** | 9–16 |

**SYSTEM.CONFIG** | 2–7, 4–4, 4–6, 5–23 |

**system disks, creating** | 2–7 |

**SYSTEM.MISCINFO** | 2–7 |

**SYSTEM.PASCAL** | 2–7 |

**SYSTEM.PME.88** | 2–7 |

**SYSTEM.PME.87** | 2–7 |

---

<table>
<thead>
<tr>
<th>T</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Tab Character</td>
<td>3–12</td>
</tr>
<tr>
<td>time, IBM Special unit</td>
<td>10–8, 10–10</td>
</tr>
<tr>
<td>T/o DOS</td>
<td>3–20</td>
</tr>
<tr>
<td>tracks per side, Config</td>
<td>4–16</td>
</tr>
<tr>
<td>Transfer</td>
<td>3–11, 3–20, 5–13</td>
</tr>
<tr>
<td>Turtlegraphics</td>
<td>8–7</td>
</tr>
<tr>
<td>TURTLEGRAphics unit</td>
<td>10–11</td>
</tr>
<tr>
<td>TV adjustment</td>
<td>4–12</td>
</tr>
</tbody>
</table>

1700101:01A | 1–7 |
Index

- U -
UNITREAD ........................................ 6-15
units, access/configuration .................. 5-3
UNITSTATUS .................................. 8-11, 6-15
UNITWRITE ..................................... 6-15
user-defined disk drivers .................... 4-19
using the p-System .......................... 2-6
utility programs .............................. 9-3

- V -
vertical tabs (printer). Config ...................... 4-38
video characters ............................. 10-26
video control codes .......................... 10-25
video mode ................................... 10-25
video mode. Config ........................ 4-12
video page .................................. 10-9
video page. Config .......................... 4-11
virtual volumes
  associated information ..................... 3-22
  building .................................. 3-5
  dismounting ................................ 3-8
  mounting ................................... 2-4, 3-16
  virtual console driver ..................... 6-4
  virtual printer/remote driver ............. 6-6
  virtual system driver ..................... 6-5
  virtual volume driver ..................... 6-3
VIRTSEL unit .................................. 5-15
Volumes ....................................... 3-22
VVCreate .................................... 5-16
VVDevices ................................... 5-17
VVDisMount .................................. 5-16
VVInfo ....................................... 5-17
VVMount ..................................... 5-18
Index

-W-

write verify, Config.............................................4-17
writing a device driver.........................................7-3