IDENTIFICATION

Product Code:	DEC-08-MEXA-D
Product Name:	PS/8 Programming System Programmer's Reference Manual
Date Created:	July, 1970
Maintainer:	PDP-8 Software Development

This document contains advanced information on the PS/8 Programming System, and is subject to change without notice.

PREFACE

This document describes the PS/8 Programming system for the PDP-8/I or PDP-8/L with at least 8K of core, one DECtape and an ASR-33 Teletype.

Chapters 1 through 4 cover information required by the average PS/8 user. The discussion assumes familiarity with 8K PAL-D, 8K SABR, 8K FORTRAN, the Symbolic Editor, ODT, the Absolute Binary Loader, and the Linking Loader (all of which are covered in the PDP-8 handbook Introduction to Programming 1970).

After a general introduction in Chapter 1, the structure and operation of PS/8's filing system is described in Chapter 2 as necessary background to using PS/8. In Chapters 3 and 4, interaction with PS/8 via the Teletype is detailed and many examples of possible command strings are given.

User programs run while PS/8 is on DECtape or disk can call upon PS/8 to perform directory operations and to use the I/O device handlers for user input and output.

Information about the preliminary version of PS/8 is presented in Chapter 6, along with internal information of interest mainly to the advanced programmer.

At the end of this document, appendixes summarize, in easily referenced form, the commands and error messages. An index concludes the whole.

Table of Contents

Appendixes

CHAPTER 1

INTRODUCTION

The PS/8 Programming System is a keyboard-oriented program-development system, expandable to accommodate any amount of core memory from 8K up. In addition to the language processors (FORTRAN, 8K PAL-D, and SABR) the PS/8 system includes:

- Absolute and Relocatable Loaders
- A Symbolic Editor
- CONVRT (a program to provide file compatibility with the present Disk Monitor System)
- PIP (Peripheral Interchange Program)
- An invisible ODT (Octal Debugging Technique) which allows the programmer to debug programs without giving up core space.

An important aspect of PS/8 is that user programs may call upon the system for various services, including:

- Loading device handlers
- Searching file directories
- Creating and closing variable and fixed length output files
- Decoding a line of Teletype input identifying I/O files and options
- Program chaining, that is, the process of automatically starting a subsequent program upon completion of a preceding one, often using results from the preceding one in the execution of the subsequent one.

PS/8 provides true device-independence, i.e., user and system programs can be written without concern for specific I/O devices. Programs can then be run using the most effective I/O devices available at a given installation. Further, if the system configuration is altered, programs need not be rewritten to take advantage of the new configuration. The PS/8 system controls the copying of data from any medium to any other medium. Logical names can be assigned to devices within the system to enable symbolic referencing of devices, making for easy-to-follow programs.

With PS/8 it is possible for the user program to specify variable length buffers. Large buffers ensure efficient use of storage devices and a minimum of time spent in data transfer operations by minimizing disk and tape motion. PS/8 takes full advantage of the new RK8 disk pack for fast bulk storage, yet full system services are possible with a single DECtape.

This discussion of the PS/8 Programming System assumes that the reader is familiar with 8K PAL-D, 8K SABR, 8K FORTRAN, Symbolic Editor, ODT, and the operations of the Binary and Linking Loaders. The reader need not be familiar with monitor systems, the Peripheral Interchange Program (PIP), and the program to convert the 4K Disk Monitor ASCII files to PS/8-compatible files (CONVRT); these are thoroughly explained in this chapter.

1.1 HARDWARE

The PS/8 Programming System can operate using either disk or DECtape as the system device. To accommodate PS/8, the disk configuration must have 96K or more words of storage and either a DECtape or high-speed papertape reader/punch.

The minimum PS/8 configuration is a PDP-8/I or PDP-8/L with at least 8K of core, one DECtape used as the system device, and an ASR-33 Teletype. A multiple DECtape system performs appreciably faster than a single DECtape system. The multiple DECtape system reduces DECtape motion because it is possible to copy directly (i.e., without intermediate searching) from the system DECtape to another DECtape (or vice versa) when editing or assembling.

Up to fifteen devices may be interfaced to PS/8. The devices include:

- up to 8 DECtape units (TCØ1/TU55 or TCØ8/TU56)
- high-speed papertape reader and punch
- up to four RK8 disks
- up to four DF32 disks or RFØ8 disks
- a card reader
- a line printer

- Linctape (compatible with the PDP-12)
- industry-compatible magnetic tape
- any other device for which it is possible to write a device handler in one, or at most, two pages.

1.2 SYSTEM SOFTWARE COMPONENTS OF PS/8

The main software components of the PS/8 system are:

1. The Keyboard Monitor, which accepts commands from the keyboard to create logical names for devices, run system and user programs, save programs, and call in debugging aids. The Keyboard Monitor provides communication with the PS/8 executive routines.

2. A library of system programs:

   a. Peripheral Interchange Program (PIP),

   b. Editor,

   c. 8K PAL-D Assembler,

   d. Absolute Binary Loader,

   e. 8K SABR Assembler,

   f. Linking Loader and 8K FORTRAN Subroutine library,

   g. 8K FORTRAN Compiler,

   h. a conversion program (CONVERT) to provide file compatibility with the Disk Monitor System,

   and other standard system programs as they become available.

3. The Command Decoder accepts a command string from the keyboard, indicating input and output files and various options. Following a keyboard command to run a system program, one or more command lines indicate to the system program what files are to be used as input, what to call the output file(s) and how to perform the function. The Command Decoder communicates with the System Library Programs.

4. Device handlers, that is, subroutines which accept a standard calling sequence from all programs and perform input and/or output operations on a device.

5. The User Service Routine (USR) which controls directory operations for the PS/8 system. The user program may call upon the USR using standard subroutine calls. Some of the functions performed by the USR are:

a. loading device handlers,

b. searching file directories,

c. creating and closing variable and fixed length output files,

d. decoding command strings from the keyboard,

e. chaining from one program to another.

When PS/8 is operating, the Command Decoder, USR, and the Keyboard Monitor are swapped into core from the system device as needed and when their operation has been completed, the previous contents of core are restored.

CHAPTER 2

FILING SYSTEM

Before going on to a study of how to operate the PS/8 system, a discussion of the files and devices being manipulated is in order. The filing system is basic to an understanding of the keyboard commands and service calls. Names for I/O devices and files on these devices are used for symbolic reference in keyboard commands and service calls.

2.1 TYPES OF FILES

Files may be permanent, tentative, or free:

1. A permanent file is one which is no longer expandable. That is, it has a fixed size.

2. A tentative file is a file that has been opened for output and has not yet been closed. If a tentative file is no longer receiving output, it is said to be inactive. An inactive tentative file will be deleted by the USR (the User Service Routine) at the first opportunity. Such an inactive tentative file might have been created by a program which was interrupted before completion. When a tentative file is still receiving output, it is said to be active. A directory device may have only one active tentative file at a time. Upon closing, a tentative file becomes permanent.

3. A free file is a contiguous area of unused storage space which is available for output.

Two calls to USR (ENTER OUTPUT FILE and CLOSE OUTPUT FILE) operate together for handling output files. The ENTER operation may be used to create an active tentative file. This file may have the same name as a permanent file.

Upon closing, the active tentative file becomes the new permanent file and the old permanent file of the same name (if it existed) is deleted. If the user program fails or is interrupted before executing the CLOSE operation, the tentative file becomes inactive and is soon deleted automatically; the old permanent file of the same name (if one existed) is untouched.

2.2 DEVICE NUMBERS

At system-generation time, the I/O devices in a given configuration are each assigned a number between 1 and 17 (inclusive). The number is used to refer to the device in coding user programs which interface with the USR of PS/8. Each of the devices also has a permanent (system) name.

2.3 PERMANENT DEVICE NAMES

The correspondence between permanent names and the standard I/O devices follows.

Permanent Name	I/O Device
SYS:	Systems device (Disk if system haslarge disk (RK8 or RFØ8), otherwise DTAØ)
DTAn:	DECtape n (n is an integer in the range 0 to 7, inclusive)
DSK:	The default storage device for files. The assignment of DSK: is specified at system generation time. In the preliminary software the assignment is fixed as follows: For a single system DSK: = SYS: = the disk For a DECtape system DSK: = SYS: = DTAØ:
TTY:	Teletype keyboard and printer
PTP:	High-speed papertape punch
PTR:	High-speed papertape reader
CDR:	Card reader
LPT:	Line printer
MTAn:	Magnetic tape n
LTAn:	LINCtape n (formatted for the PDP-12)

2.4 USER DEVICE NAMES

Each device may be given a user name by the keyboard command ASSIGN. The user name is composed of up to four alphanumeric characters of which the first must be alphabetic. The user name takes precedence over the permanent name. For example, you may assign the user name IN to the device with the permanent system name DTA1, via the command

```
ASSIGN DTA1 IN
```

Future references to IN will refer to DTA1. Thus, device-independent programs are easily possible since a change in the user name of a device using the ASSIGN command can change the operation of a routine without changing the code. User names may be removed by the same ASSIGN command without a name argument. For example,

```
ASSIGN DTA1
```

would remove the user's name for DTA1. All user device names may be deleted at once by the keyboard command DEASSIGN.

A device name may consist of up to four alphanumeric characters. All one and two character names are unique. Due to a system limitation, not all user device names of three and four characters are uniquely represented in the system.

The device name is internally coded in one 12-bit word. If the name is one or two characters long, the encoding is the name itself. If the name is three or four characters long, the name is the sum of the two words in the name with the sign bit set. As a result, some three or four character names have common encodings. For instance, DTA3 and CSB4 have the same encoding.

It is, therefore, recommended that user device names be one or two characters only. A three or four character name may be tested for uniqueness by typing

```
ASSIGN name
```

If a

NOT AVAILABLE

message results (indicating the device name is not in the system tables), the name is unique.

Some devices (such as disks and DECtapes) are directory devices, that is, they are a collection of entities called files. The system keeps track of these files by means of a Directory, which is a list of file names and their corresponding storage space on the device.

2.5 FILE NAMES

Files are referred to symbolically by a name of up to six alphanumeric characters optionally followed by a period and an extension of two alphanumeric characters. The extension to a file name is generally used as an aid for remembering the format of the file.

2.6 DEFAULT AND ASSUMED EXTENSIONS

If an extension is not explicitly given for an input file, a default extension will be appended by the Command Decoder if the program which calls the Command Decoder so specifies. If the file with the default extension appended is not found, then a search for the original file (without any extension) is made. When using the Command Decoder, most system programs make use of this default extension feature. Default extensions apply only to input files. In addition, some system programs append extensions to their output files if the user does not specify an extension. These extensions are called assumed extensions.

Assumed extensions apply only to output files. The keyboard monitor uses an assumed extension of .SV when referencing core-image files.

In the table which follows, extensions and their usage are detailed.

Extension	Indicates	Usage
.SV	a core image (SAVE) file	The assumed extension for the R,RUN, SAVE, and GET Keyboard Monitor Commands.
.FT	a FORTRAN source file	Default extension for 8K FORTRAN.
.SB	SABR source file	Default extension for 8K SABR.
.PA	PAL8 source file	Default extension for 8K PAL-D.
.BN	absolute binary file	Default extension for Binary Loader. Assumed extension for 8K PAL-D binary output.
.RL	relocatable binary file	Default extension for Linking Loader. Assumed extension for 8K SABR.

NOTE

The 8K FORTRAN Library file LIB.8.RL is not (as a whole) relocatable, though each subroutine of the file is.

.MP	file contains a loading map	Assumed extension for Linking Loader map.
.LS	an assembly listing	Assumed extension for 8K PAL-D and SABR listing output.
.TM	a temporary file	Extension for FORTRAN- and SABR-generated files for system use.

For example, if the user types:

```
RUN DSK PROG
```

the extension .SV will be assumed and the file named PROG.SV (on device DSK) will be run if found. If the user types:

```
RUN DSK PROG.A
```

then PROG.A will be run.

2.7 DIMENSIONS

The terms block, record and page are defined thus:

1 block=1 record=2 pages=256_{10} words

In directory listings, calls to the USR, etc., reference file length in terms of blocks (or records).

CHAPTER 3

KEYBOARD COMMANDS

Commands typed on the Teletype keyboard may be directed to the Keyboard Monitor or to any of the system programs in the PS/8 library. The Keyboard Monitor indicates it is ready to receive a command by printing a dot. The System Programs use the Command Decoder to interpret user commands. The Command Decoder indicates it is ready to receive a command by printing an asterisk.

Each command to the Keyboard Monitor or Command Decoder is typed on the Teletype and corrected, if necessary, before ending the line and initiating execution. The RUBOUT key may be used to delete the last character typed. Pressing this key prints a backslash (\ ) character followed by the character which was deleted. Successive pressings of the RUBOUT key each cause one more character to be printed and deleted. The first non-RUBOUT character typed after the last rubout in a sequence causes a closing backslash to be printed. For example:

```
User types:       RUN DSK (RUBOUT) (RUBOUT) (RUBOUT) DTA1 FILE
Teletype shows:   RUN DSK \ KSD\ DTA1 FILE
Monitor sees:     RUN DTA1 FILE
```

The key combination CTRL/U (produced by holding down the CTRL key while typing U) can be typed anywhere in an input line. CTRL/U echoes on the Teletype as "↑U". The line on which it occurs will be ignored. To the Keyboard Monitor, typing CTRL/C anywhere in an input line causes "↑C" to be printed and the line to be ignored. To the Command Decoder, typing CTRL/C prints "↑C" and returns control to the Keyboard Monitor.

The RETURN key is typed to enter a command to the computer and to initiate its execution. The ALT-MODE (also called ESCape or PREFIX on some Teletypes) key may also be used to end a line. This key will print on the Teletype as a dollar sign ($).

If the LINE FEED key is pressed at any time while entering a line, the line is echoed as the Keyboard Monitor currently sees it. For example:

```
                          (RUBOUT)          (RUBOUT)

User types:  RUN DTA3 \3\ 2  PRG  \G\ OG  (LINE FEED
Monitor
echoes:      RUN DTA2  PROG
```

The line is not terminated. The user may terminate the line, append or change it.

3.1 CORE CONTROL BLOCK

A block of data internal to PS/8 can be directly changed by commands from the keyboard. This block, called the Core Control Block, contains information about the file such as its starting address, the areas of core occupied by the file, and a word called the Job Status Word.

The Job Status Word is saved and loaded with a file to indicate what locations in core the file uses and how, as follows:

Bit Number	Contents	Meaning
0	1	The file does not load over locations 0-1777.
1	1	The file does not load over locations 10000-11777.
2	1	The program is not restartable, i.e., the file must be reloaded before it can be started again.
10	1	Locations 0-1777 need not be saved if the Command Decoder is called. The CD overlays them.
11	1	Locations 10000-11777 need not be saved if the USR is called. The USR overlays them.

A core control block is set up for each core image file when the file is created by the Binary or Linking Loader or by the SAVE keyboard command. (A core image file is one which has been dumped from core onto some device by the SAVE command.)

3.2 COMMANDS TO THE KEYBOARD MONITOR

The Keyboard Monitor is ready to receive a command when a dot is printed on the Teletype. You may then type any of eight commands to manipulate files and names; execution occurs upon typing a carriage return. Only the first two characters of the command need be typed. In the description of the eight commands to the Keyboard Monitor which follows, optional arguments are enclosed in parentheses. The following abbreviations are used for indicating arguments:

Abbreviation	Meaning
dev	any device name
pdev	a permanent device name
udev	a user device name
file	a file name
ext	an extension to the file name

1. ASSIGN pdev (udev)

   The specified user device name is associated with the permanent device name in the system tables. Only one user name may be associated with a device at a time. For example,

   ```
   ASSIGN DTA1 IN
   ```

   causes all future references to IN to refer to DECtape unit 1.

   If the optional argument for the user name is missing, the current user name associated with the device will be cleared.

2. DEASSIGN

   All current user names are disassociated from the devices.

3. GET dev file (.ext)

   This command handles only core image files. The specified file is loaded into core and its core control block is moved to the system scratch area.

   If an extension is not specified, the extension SV (for core image files) will be added to the file name. For example,

```
GET DTA3  OH
```

will fetch the file OH.SV from DTA3.

4. SAVE dev file (.ext) (aaaaa-bbbbb, ccccc,...) (;sssss) (=nnnn)

where: aaaaa-bbbbb, ccccc,... are the addresses of the areas of core to be SAVE d

;sssss is the starting address

=nnnn is the Job Status Word

The program currently in core is saved on the specified device with the specified file name. As before, if an extension is not specified, the extension SV is added to the file name. If the other three optional arguments are not given, the information they convey is taken from the core control block.

If an error message is printed in response to the SAVE command, the program currently in core has not been changed as yet.

For example,

```
SAVE DSK OHLA.SV 55,10500-10577;10502
```

saves the program in core on the disk as a file named OHLA.SV. The areas of core saved will be 0 - 177 of field 0 and 400 - 577 of field 1. The starting address of the program is 502 in field 1. The Job Status Word will be taken from the core control block.

5. START (sssss)

The program currently in core is started at location sssss if it is specified. If the argument sssss is missing, the program is started at the starting address specified in its core control block.

6. ODT

The system ODT which is completely invisible to the user is loaded into core and started. ODT occupies locations 0-1777 in field 0. It accepts five digit core memory addresses. If the CTRL/0 combination is typed, ODT will terminate printing and return to command mode.

7. RUN dev file (.ext)

This clmmand handles only core image files. The specified file on the specified device is loaded into core, its core control block is moved to the system scratch area, and the program is started at its starting address.

If an extension is not specified, the extension SV is added to the file name. For example:

```
RUN DTA1 HMM
```

causes the file HMM.SV on DECtape 1 to be loaded into core, its core control block to be moved to the system scratch area, and the program to be started.

8. R file (.ext)

This command handles only core image files from the system device (SYS.). The specified file on the system device is loaded into core and started at its starting address. If an extension is not specified, the extension SV is added.

R differs from RUN in that a core control block is not moved with the file. In order to SAVE a program which does not have its core control block with it, all the optional arguments of the SAVE command must be explicitly stated. System programs are most often called using the R command, since they need not be SAVEd. To call a user program to be later SAVEd, use either the RUN or GET command.

3.2.1 Error Messages

If an error is made in typing a command to the Keyboard Monitor, one of the following messages is printed.

Message	Meaning
device NOT AVAILABLE	The device name specified (to a RUN, GET, SAVE or ASSIGN command) is not in any system table.
xxxx ?	"xxxx" is not a legal command; e.g., if the user typed "HELLO" the system would respond "HELLO?"
filename NOT FOUND	The file name specified (to a RUN, GET, or R command) does not exist on the device specified.
TOO FEW ARGS	You left out an argument to a command; e.g., "RUN DSK" would produce this message.
NO!!	You attempted to START a program which cannot be started for some reason. For example, the program may modify itself and hence must be reloaded before starting it again.
SAVE ERROR	An I/O error has occurred while SAVEing a program. The program is still intact in core after this message is given.
ILLEGAL ARG	Bad syntax was detected in the optional arguments of a SAVE command.
BAD ARGS	Two areas of core specified in the optional arguments of a SAVE command overlay each other.
USER ERROR Ø AT xxxxx	An I/O error was detected while loading a program (RUN, GET or R commands). "xxxxx" is a meaningless vestigial location.
MONITOR ERROR 2 AT	An attempt was made to output to a WRITE LOCKED device, usually DECtape.

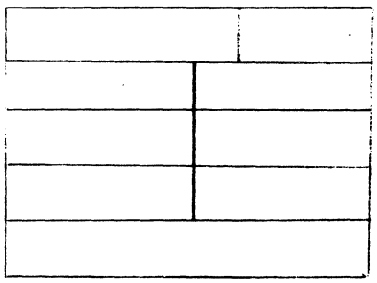
3.3 THE COMMAND DECODER

As soon as the system program is running, it uses the Command Decoder to receive a list of output devices and files, a list of input devices and files, and a list of options. These specifications you type on the line (or lines) following the R command. The Command Decoder prints an asterisk to ask for a command string.

The PS/8 Command Decoder is used to transmit information about specified inputs, outputs, and options to programs requiring that information. The Command Decoder builds up its information in Field 1 in certain specified locations.

3.3.1 Output Files

There is room for three entries in the output file list. The list begins at 17600 and appears as follows:



4 bit device number

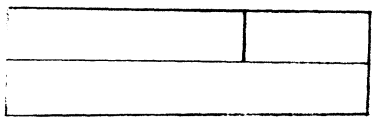
Filename in stripped ASCII (6 characters)

Extension (2 characters)

3.3.2 Input Files

There is room for 9 entries in the input file table. The list is built up from 17617, and each entry appears as follows:

8 bit file length



4 bit device no.

starting record (block) no. on specified device

NOTE

The file length is returned as a negative number, i.e., 377 in these bits is a length of 1 block, 376 is a 2 block file, etc.

A file length of 000 means the specified file has a length greater than or equal to 256 blocks.

3.3.3 Parameter Table

Options specified to the Command Decoder appear as bits in the parameter table. There are bits to correspond to A-Z, 0-9.

The list is three words long, beginning at 17643.

To clarify the above, consider an example. Assume the input to the Command Decoder appeared as:

*SYS:BILE.PA PTR:,DTA4:MASS/T

The Command Decoder tables would appear as follows:

Output File List:

8 bits	4 bits	
000	01	system device
02	11	BI
14	05	LE
00	00	
20	01	PA
10 words of zeros		

The input file list would appear as follows:

Meaningless for PTR (non-directory device)	0	17	PTR is device
	0		Meaningless for non-directory device
File length of MASS	367	11	DTA4 is device
	0050		Starts at record 50_8
	14 words of zeros		

Since there were two inputs, two entries were used.

The option list would appear as follows:

17643	0000	A-L not on
	0020	T bit on
	0000	Y,Z,0-9 not on

NOTE

The 4 bit device numbers are not sacred. They will depend on the configuration specified when the system is built.

3.3.4 I/O Specification

The expected string for I/O specification takes the form:

(output files)←(input files)

There may be 0 to 3 output files and 0 to 9 input files, depending upon the requirements of the system program as later discussed for each system program. For example, the assembler uses the first output file for binary, the second output file for the listing. The files are separated by commas. If no output files are given, the backarrow ← may be omitted.

File specifications for input and output files take the following forms:

Form	Meaning
device: filename	the specified file on the specified device
device:	the specified device used as a non-directory device, e.g., referencing DSK: will cause the whole disk to be used.
filename:	the specified file on an assumed device, depending upon context: 1. for output files and the first input file, the device is assumed to be DSK: 2. for input files after the first, the device is assumed to be that of the previous entry
(a null)	(By a null is meant the absence of an explicit specification.) The meaning depends upon context: 1. for output files, a null indicates no ouput file exists for this position 2. for input files, a null indicates the device of the most recent entry used as a non-directory device.

A null specification is indicated in the following three contexts:

1. A comma as the first character on a line, for example:

```
*,LPT:←SRC      1st output file = null (means no file)
                2nd output file = LPT:
                1st (only) input file = DSK:SRC
```

2. A comma following a comma or back arrow, for example:

```
*PTR:,,DTA1:X   1st input file = PTR:
                2nd input file = null (means PTR:)
                3rd input file = DTA1:X
```

3. A carriage RETURN following a comma, for example:

```
*A←TTY:,        1st output file = DSK:A
                1st input file = TTY:
                2nd input file = null (means TTY:)
```

Examples of command strings specifying I/O:

*DSK:BINARY, LPT:←SOURCE — The file named SOURCE is the input file on the DSK: The two output files are one named BINARY on DSK and another on the line printer (device LPT:)

*INPUT1, INPUT2, INPUT3, PTR:, — This string of input files and no output files might be typed to one of the loaders (which do not require output file specification). Three files are taken from device DSK then two are taken from the paper tape reader.

*DTA2:A,B←XYZ:C,D — The input files C and D are taken from device XYZ. The output files are a file on DECtape 2 named A and a file on device DSK named B.

The left arrow character (←) is used in this preliminary version of PS/8. In later versions the left angle bracket (<) will be interchangable with the left arrow.

3.3.5 Option Syntax

The Command Decoder accepts options typed on the same line as the previously described I/O specifications. Options may be octal numbers or single alphanumeric option characters. Option characters are set off from I/O specification by the slash / for single characters and by parentheses ( ) for a string of single characters. The equal sign = and square brackets [ ] are used to set off octal numbers. The following describes these constructions in detail, although the actual meanings ascribed to the options will be described later in connection with each system program.

1. The / construction consists of a slash followed by a single option character. This construction may occur anywhere in the input line, even in the middle of a name. For example,

   *TTY:/L←DSK:A/BC is equivalent to TTY:←DSK:AC/L/B

   and the options specified are "L" and "B".

2. The ( ) construction consists of an open parenthesis, followed by any number of option characters, followed by a close parenthesis. This construction is also valid anywhere in the input line. For example,

   *A:B←C:/Y(AXQ)Z(P) is equivalent to A:B←C:Z/X/A/Q/P/Y

   and the options specified are "X", "A", "Q", "P" and "Y".

3. The = construction consists of an equal sign followed by an octal number up to seven digits long. This can only occur once in a line and must be followed by a separator character (comma, backarrow, or end-of-line character) or by other options and a separator character. For example,

   *FILE1=1002(123),FILE2,

   (the 123 represents three separate options = "1", "2", and "3").

4. The [ ] construction can only occur immediately after an output file name and consists of an open bracket, an octal number between 1 and 377_8, and a close bracket. The open bracket is produced by holding down the SHIFT key while typing K (i.e., SHIFT/K). The close bracket is produced by holding down the SHIFT key while typing M (i.e., SHIFT/M). This construction is used to provide an upper limit on the number of records (256 words) of the output file so the system can optimize file storage.

This option need not be included in a command string although some users may find it to their advantage. For example,

```
*BINARY [12],LISTING[200]←SOURCE/8,
```

The output files, above, are a file named BINARY on device DSK: (maximum length 12_8 blocks) and a file named LISTIN (only six characters are significant) on device DSK: (maximum length 200_8 blocks). The input file is SOURCE on device DSK: and the option specified is "8".

3.3.6 Error Messages

If the command string to the Command Decoder has an error, one of the following messages may be printed on the Teletype:

Message	Meaning
ILLEGAL SYNTAX	This command line does not make sense.
TOO MANY FILES	More than three output files or nine input files were specified.
device DOES NOT EXIST	The device specified could not be found in the system tables.
filename NOT FOUND	The specified filename does not exist on the specified device.

CHAPTER 4

LIBRARY OF SYSTEM PROGRAMS

The R command to the Keyboard Monitor may be used to call the system programs using the following names as arguments. (Each system program is also briefly described and mutual interaction outlined):

Name	System Program Description
PIP	Peripheral Interchange Program - PIP is used to transfer files from one device to any other, to concatenate files, to list directories, and to pack files so that free space becomes a contiguous block.
EDIT	Symbolic Editor - the PS/8 Editor is a modified version of the Disk Monitor System Editor with four added commands. It is used to create, add to, or delete from source files before assembling or compiling (see Chapter 6).
ABSLDR	Absolute Binary Loader - used to load object programs in absolute binary format (assembled by 8K PAL-D) into core memory for execution.
PAL8	8K PAL-D Assembler - accepts source programs written in the 8K PAL-D Assembly language and outputs object programs in absolute binary format. Options to the Command Decoder used by PAL may specify calling the Absolute Binary Loader upon completion of the assembly - an example of program chaining. (For a complete description of PAL, see Programming Languages, Chapter 14).
CONVRT	DECtape conversion program - as input it accepts ASCII files on DECtapes compatible with the Disk Monitor System. It outputs the PS/8 version of the ASCII files onto DECtapes compatible with PS/8.

(The following three system programs work together as the 8K FORTRAN system. The first two also work separately as stand-alone programs.)

LOADER	Linking Loader - loads the output of the 8K SABR Assembler and 8K FORTRAN Compiler, automatically stepping through the available fields for each input file until it finds one with enough space to load the file. The loader features an automatic search of the system library, loading only

those programs which are referenced. (For a complete description, see Programming Languages, Chapters 14 and 15).

SABR — 8K SABR Assembler - may be used as a stand-alone assembler to produce relocatable binary code for the Linking Loader or to serve as pass 2 of an 8K FORTRAN compilation. (For a complete description, see Programming Languages, Chapter 14.)

FORT — 8K FORTRAN - used to compile the user's FORTRAN program and automatically call K SABR to assemble it. Options to the Command Decoder exist which will cause the Linking Loader to load the assembled program and any referenced programs of the 8K FORTRAN Library. This operation is another example of program chaining. (For a complete description, see Programming Languages, Chapter 14.)

On the line following the R command calling the system program, a command string is typed by which the Command Decoder indicates input/output specifications to the particular system program. The number and type of input and output files are dependent upon the system program which receives them. Much repetition of file names, extensions, and device names has been eliminated by the existence of default options. For example, system programs append default extensions to input names. If such a file is not found, a search is made for a file without an extension.

Command strings to the various system programs and additional features of the PS/8 versions of these programs are discussed in the following sections.

4.1 PERIPHERAL INTERCHANGE PROGRAM

After the keyboard command R PIP to call the program, the next line to the Command Decoder should specify one output file and zero to nine input files. The input files are transferred to the output file under control of the option switches:

Option	Effect
/A	Transfer in ASCII mode (default mode).
/B	Transfer in Binary Mode (used for files in absolute and relocatable binary format).
/I	Transfer in Image mode (the /I option is used for transferring core-image files and other files which do not fall into the ASCII or binary categories. For example, the FORTRAN library file SYS: LIB8.RL must be copied using the /I option. NOTE: Merging of core-image files is meaningless.
/Z	Zero the directory of the output device before beginning the transfer. Before you can use a DECtape with the PS/8 system, you must create a zero directory on it using the /Z option with no input files specified.
/D	Delete the old copy of the output file(s) before beginning the transfer. If this option were not specified, the old copy (copies) would not be deleted until the transfer was completed.
/C	Eliminate trailing blanks (ASCII mode only).
/T	Convert tabs to spaces (ASCII mode only).
=n	Save n extra words per file entry in the directory to contain descriptive information about the file (with /Z only).

If no input files are specified, no data transfer occurs. For example, the option /Z may be used to zero a directory and /D to delete a file if no input files are specified.

NOTE: It is recommended that you do not use the "=n" option in preliminary versions of this software.

Examples of command strings:

Example 1

```
.R PIP
*DTA1:FILE1←DTA2:FILE
```

transfers the file FILE on DTA2 to DTA1 in ASCII mode and calls it FILE1.

Example 2 *BIN.BN,C,SV/D←A.BN,B.BN/B

deletes the file BIN.BN on device DSK: and combines files A.BN and B.BN (both from DSK:) in binary mode to form a new BIN.BN. Also deletes file C.SV from DSK: The old copy of DSK:BIN.BN would have been deleted after completion of the transfer if /D had not been specified.

Example 3 *DTA1:/Z←

zeros the directory of DTA1:

Example 4 *PTP:←CDR:/C

moves information from the device CDR: to the device PTP: in ASCII, deleting trailing blanks.

Example 5 *DTA3:PAL8.SV←PAL8.SV/I

transfers the file PAL8.SV from DSK: to DTA3: in Image mode.

Example 6 *DTA4:LIB8.RL ←SYS:LIB40.RL/I

transfers the 8K FORTRAN Library to DECtape 4 from the system device. Loading the Library can be speeded up (by reducing DECtape motion) if the Library is transferred to a DECtape other than the system DECtape(on a DECtape system) before loading.

PIP also performs other operations for the options given below:

Option	Effect
/L	Lists the directories of the input devices onto the output file starting at the file specified. Note that in this case the input file itself is not transferred, only the directory.
/F	Lists directories in short form (file names only).
/E	Lists directories in Extended form (the lengths for the free files are also listed.
/S	Moves the contents of the input device(s) specified onto the output device, eliminating all free files. This combines all of the free space into one contiguous block instead of many fragmented blocks, and packs all the programs one after another.

More examples of command strings follow:

Example 7 .

```
.R PIP
*LPT:/L←DSK:,DTA2:BLAH
```

transfers the listing of the entire directory of device DSK: and the directory of device DTA2: starting with the file BLAH onto the device LPT:

Example 8

```
*DIRECT←DTA7:/E
```

transfers an extended listing of the directory of device DTA7: onto the file DIRECT of device DSK:.

A PIP directory listing might appear thus:

```
ABSLDR.SV     4
PIP   .SV    1Ø
LIB4Ø .RL    41
BLAH         17
<EMPTY>       6
X      Y      1
<EMPTY>     511

517 FREE BLOCKS
```

These octal numbers represent the length of the files in records and are listed for the "L" and "E" options only.

Free files like these are only listed if the "E" option is specified.

4.2 SYMBOLIC EDITOR

The Editor is called by the command R EDIT. Then the line to the Command Decoder should specify one output file and Ø to 9 input files. If the input is null, then the R Editor command is illegal. Instead, the A command is used for input from the Teletype. The command string may include the following options:

Option	Effect
/A	Upon closure of the output file by the E or Q Editor commands, return control to the Command Decoder and leave the Editor in core. Without the /A option, control returns to the Keyboard Monitor after an E or Q command.
/B	Convert two or more spaces to a tab.
/D	Delete the old copy of the output file (if one exists) before opening the edit output file.

For example,

```
*DTA2:FILE←DTA1:BLECH/D
```

This will delete FILE on DTA2 before opening the new file FILE.

The Editor is a modified version of Disk Monitor System Editor described in Chapter 6. The following commands and error messages are in addition to those described therein.

Command	Meaning
E	Close the presently open output file first copying the editor buffer and the rest of the input file into the output file.
nY	Delete the next (n-1) pages and read the nth page into the text buffer. For example, 5Y will kill (delete) the next four pages, then read and leave in the fifth page.
Q	End the file with the last page already processed by the P command. Compare this with the E command, which ends the file by passing the rest of the file to the output device.
B	Print the decimal number of unused core locations which remain in the text buffer (maximum capacity approximately 5000 source characters). To estimate the relationship between locations and characters, the value 1.7 characters per location may be used.
CTRL/R	(an alternative way to merge files) return control to the Command Decoder to receive more input and output file specifications and options while retaining the present contents of the text buffer. A carriage return sends control back to the Editor. Compare this with CTRL/C followed by the START command, which loses the contents of the buffer.

If control returns to the Command Decoder during output operations, the specified output device lacked enough space for the output. You may specify a different output device, and complete the output operation upon typing the carriage return which sends control back to the Editor. Otherwise, if an error occurs during data transfer, the Editor prints the message:

? n ↑C

where n refers to an error code in the following table.

n	Meaning
0	failed in reading the device
1	fatal write-error
2	output file close-error
3	output-file open-error

4.3 ABSOLUTE BINARY LOADER

Call this Loader with the keyboard command R ABSLDR. Then type a number of lines to the Command Decoder, each containing input files only (no output files) and options. A line ending in a carriage return indicates that there is more input to come. The last line of of input files is indicated by the /G option or ended with an ALTMODE character. If extensions to the input file names are not specified, the default extension .BN is added. The input files are loaded according to the options specified and a core control block is set up.

Option	Effect
/S	Load all the binary programs in the specified input files (instead of loading only the first program in each file).
/8	Set bit 10 of the Job Status Word (this implies that the program does not care if locations 0-1777 are destroyed when the Keyboard Monitor is called).
/9	Set bit 11 of the Job Status Word (this implies that the program does not care if locations 10000-11777 are destroyed when the Keyboard Monitor is called).
/O	Treat the input file (only one permitted) on this line as a core image to be loaded and overlaid with the input of subsequent lines. This option may be used only in the first command string (not yet implemented).
/G	Start program when loading is complete. The start address may be specified using the remaining options. Otherwise, the starting address will be location 200. If /G is not specified, control returns to the Keyboard Monitor so that the core loaded may be saved or started.

/n (where n is an integer). Force loading of all files specified on this line into field n.

=n Set the starting address to n, (where n is an octal number up to five digits octal). For example,

```
*BIN1,DTA1:BIN2.MY←(8G)=13ØØØ
```

Load the file BIN1.BN (or BIN1) from DSK: and BIN2.MY from DTA1:, setting bit 10 of the Job Status Word and start at location 13ØØØ.

Examples of Command Strings:

Example 1

```
.R ABSLDR
*DTA3:FOCAL /S
*ODT/1/G=17ØØØ
```

Assuming that you had placed the binary of DEC's FOCAL program on DTA3: in the file FOCAL.BN (or FOCAL) and you had a binary version of the DECUS program XOD in the file ODT.BN (or ODT) on device DSK: this sequence would load all the programs in the FOCAL file into core (FOCAL is actually two or three binary programs, hence the /S is necessary) then load XOD into field one and start the combined program at location 7000 in field 1.

Example 2

```
.R ABSLDR
*DTA1:8KFORT,PTR:$↑  (the $ stands for ALTMODE)
.SAVE SYS FORT
```

This would load the file 8KFORT.BN (or 8K FORT) from device DTA1: and also a tape from device PTR:. The up arrow ↑ asks if the tape is in the reader. In order to initiate the reading of the tape, type any character on the Keyboard in response to the up-arrow. The combined programs are left in core and control returns to the Keyboard Monitor. The resulting program has a starting address of 00200 and may be saved (as in this example) or started using the START command. When putting a binary tape into the reader for input to the ABSLDR, it is not necessary to have leader (200_8) code under the read head. The tape may be positioned so that the blank tape at the beginning is under the read head.

4.4 8K PAL/D ASSEMBLER

The keyboard command to call 8K PAL-D is R PAL8. The line to the Command Decoder consists of an optional binary output file followed by an optional listing output file, 0 to 9 input files, and various options. A null output file indicates no output of that type is to be generated. For example,

*,TTY: ←DSK: ISIT

indicates no binary is to be generated, only a listing on the Teletype.

If extensions to the file name are omitted, the following extensions are assigned:

File Type	Extension	
input	.PA	(default extension
binary output	.BN	(assumed extension)
listing output	.LS	(assumed extension)

The following options may be included anywhere in the command string.

Option	Effect
/S	Omit the symbol table normally generated with the listing (applicable only if a listing file is specified).
/N	Generate the symbol table but not the listing (also applicable only if a listing file is specified).
/L	Call the Absolute Loader at the end of assembly and load the binary file (only applicable if a binary file is specified).
/G	Equivalent to /L/G - i.e., call the Absolute Loader, load and go.
/D	Generate an ODT-compatible symbol table.
(Options to the Binary Loader)	When the /L or /G option is specified, any options to the Binary Loader may be included in the command string for PAL8.

Examples of Command Strings:

Example 1

```
.R PAL8
*PTP:,LPT:←SOURCE
```

assembles SOURCE.PA (or SOURCE), putting the binary output on the paper tape punch and the listing and symbol table (in that order) on the line printer.

Example 2

```
*,BLAH←ECCH/S
```

assembles ECCH.PA (or ECCH), putting the listing only into file BLAH.LS on device DSK: No binary output is generated.

Example 3

```
*BIN←INPUT.XY/G=6ØØ
```

assembles INPUT.XY putting the binary output into BIN.BN, and then calls the Binary Loader, which loads BIN.BN and starts it at 00600. ("=600" is an option to the Loader specifying the start address.)

4.5 CONVERSION PROGRAM

Call the conversion program with the command R CONVRT. The line to the Command Decoder consists of an optional output file, one input file, and up to two options. Only ASCII files are handled by CONVRT. The output file, if specified, will be an ASCII file compatible with PS/8. If an output file is not specified, the /L option alone must be used. The input device must be a DECtape compatible with the Disk Monitor System and be specified by the form DTAn: where n is an integer in the range 0 to 7 (0 is DECtape unit 8). If an error is made, a message explaining the fault is printed on the Teletype. After the completion of the task(s) or the occurrence of an error, CONVRT will recall the Command Decoder, accept another command string, and perform the specified task.

The following are the two available options:

Option	Effect
/D	Look up the input file using the directory and storage map already in core. If there is no directory in core when this option is given, the message INPUT FILE NOT FOUND will be printed. If this option is not specified, the directory of the input device will be read into core for each conversion.
/L	Generate a directory listing of the input ASCII files of the device on the Teletype.

Examples of Command Strings:

Example 1

```
*SYS:YES←DTA5:YES
```

converts the ASCII file YES on DECtape 5 to a PS/8-compatible file YES on the systems device.

Example 2

```
*DTA2: /L
```

prints on the Teletype a directory listing of the ASCII files on DECtape 2, then recalls the Command Decoder.

Example 3

```
*SYS:GOPS.AS←DTA3:GODM(DL)
```

Generates a directory listing of the ASCII files of DTA3 on the Teletype, then converts the Disk Monitor file GODM on DECtape 3 to a PS/8 file GOPS.AS on the system device using the directory already in core.

4.6 LINKING LOADER

This Loader is called by the command R LOADER. The next line to the Command Decoder consists of 0 to 1 output files, 0 to 9 input files and various options. Only one binary program per file is permitted. The default extension for input files is .RL. A carriage return initiates the specified tasks. Other tasks may be specified on the following lines. The last line is ended by an ALT-MODE character or contains a /G (start program or go) option.

The options are as follows:

Option	Effect
/G	Start the program after processing the rest of the command string. Execution starts at the symbol MAIN unless otherwise overridden by an option.
/M	Output a map of loaded programs onto the output file specified. If no output file is specified, the map is put onto the Teletype. The default extension for the map output file is .MP. The map is printed after the rest of the line is processed.
/U	Like /M but only outputs undefined symbols.
/P	Like /M but only outputs counts of free pages in each field.
/n	(where n is an integer in the range 0 to 7 inclusive). Search through the available fields starting at field n for space large enough to load each input file. Only one binary program may be in each input file. If not specified, the Loader starts looking at field 0.
/R	Restart loading process (forget all previously loaded programs).
=n	(where n is an octal number up to 5 digits long) Specifies the starting address of the program if it is other than the entry point MAIN.
/L	Load the specified file (only one file permitted on the line) as a library, i.e., the loader expects a directory at the front.

The Loader automatically searches the 8K FORTRAN Library and loads those programs which are referenced after processing the last command string (ending with ALT-MODE or containing /G). If a loading map is requested on the last line, it is generated after the Library is searched.

NOTE

The 8K FORTRAN Library is in the file LIB8.RL on the device SYS: Despite its extension RL, it is not as a whole relocatable since it begins with a directory. Instead each of its elements is relocatable. The Library as a whole can be specified as input to the Linking Loader only by using the /L (Library) option. For example,

```
*SYS:LIB8.RL/L
```

tells the Linking Loader to load the Library as a Library, i.e., load only the necessary elements of the Library into core.

Examples of command strings to the Loader:

```
*PROG,DTA2:SUBR1,SUBR2/G
```

loads DSK:PROG.RL,DTA2:SUB1.RL, DTA:SUB2.RL loads the Library from SYS:LIB8.RL and starts the program at the entry point "MAIN".

The same process could also have been done as follows:

```
*PROG                   Load DSK:PROG.RL

*TTY:←/U                Get a list of undefined symbols on
                        the Teletype
     .
     .
     .
(a list of unde-
fined symbols
goes here)
     .
     .
     .
*DTA2:SUBR1,SUBR2       Load DTA2:SUBR1.RL,SUBR2.RL

*LPT:/M←$               Put a loading map on the Line Printer
                        load the Library from SYS:LIB8.RL and
                        exit.

                        ($ stands for ALTMODE)

.SAVE DTA2 FORTPG       Save the core image on DTA2 as FORTPG.SV

.START                  Start the core image at its starting
                        address (entry point "MAIN")
```

The Linking Loader gives error messages in the form "ERROR nnnn". The meaning of the different values of nnnn are:

Error Code	Explanation
0001	Symbol table overflow (more than 64 subprogram names).
0002	Program will not fit.
0003	Program with largest common storage was not loaded first.
0004	Checksum error in input tape.

Error Code	Explanation
0005	Illegal relocation code.
0006	An output error has occurred while outputting a storage map
0007	An input error has occurred while reading a binary file.
0010	No starting address has been specified and there is no entry point named "MAIN".
greater than 0010	An internal error has occurred. Please send the typeout and history to Digital Equipment Corporation.

4.7 8K SABR ASSEMBLER

Call this Assembler using the R SABR command. The line to the Command Decoder consists of 0 to 3 output files: the first for binary, the second for the listing, and the third for Linking Loader output for the /M, /U, or /P options when chaining to the Linking Loader occurs. Next on the line are 1 to 9 input files and various options. A null binary file is assumed to be SYS: FORTRL.TM if the /L or /G option is specified. Otherwise, a null binary output file indicates no binary output is to be generated. Similarly, null listing file indicates no listing is to be generated.

The default extensions for SABR are as follows:

File Type	Extension	
input	.SB	(default extension)
binary output	.BN	(assumed extension)
listing output	.LS	(assumed extension)

The following options may be included in the command string:

Option	Effect
/S	Omit the symbol table (applicable only if a listing file is specified).
/N	Output the symbol table but not the rest of the listing (applicable only if a listing file is specified).
/L	Call the Linking Loader at the end of the assembly and load the specified binary file. If a binary output file is not specified, then the temporary file FORTRL.TM is loaded into core and deleted from the file device.
/G	Equivalent to /L/G, that is, call the Linking Loader, load and go. If a binary output file is not specified, then FORTRL.TM is loaded and deleted. If a starting address is not specified (using the options to the Linking Loader), control is sent to the location labeled "MAIN".
/F	Indicates that the input file is an 8K FORTRAN output file.
(Options to the Linking Loader	When the /L or /G option is specified, any options to the Linking Loader may be included in the command string for SABR except the /L (Library) option of the Linking Loader.

Examples of command strings:

Example 1

```
.R SABR
*FORTRN.TM/F/G
```

DSK:FORTRN.TM is assembled as a FORTRAN output file and the relocatable binary is loaded and started at MAIN.

Example 2

```
*SYS:TEERL,TTY:←TEE /S
```

The input file TEE.SB (or TEE) on DSK: is assembled. The relocatable binary goes to the output file TEERL.RL on SYS:, the listing without a symbol table goes to the Teleprinter.

The following error messages have been added to the PS/8 version of SABR:

Message	Meaning
S	The definition of the S error message has been extended to include any system error, such as overrunning an output file.
D	The D error signifies that a Device Handler returned a fatal error.
U	The U message is printed if no symbol table is being produced and there is at least one undefined symbol in the program.
L	The L error indicates that the /L or /G options were requested but LOADER.SV does not exist on device SYS:

See Chapter 14, Programming Languages, for other messages.

4.8 8K FORTRAN COMPILER

The line to the Command Decoder consists of 0 to 3 output files -- the first for binary, the second for the listing and the third for Linking Loader output for the /M, /U, or /P options when chaining to the Linking Loader occurs. Next on the line are 1 to 9 input files and various options.

The default extension for FORTRAN input file is .FT. The Compiler produces an output file named FORTRN.TM on the system device, for input to the 8K SABR Assembler. The Compiler automatically calls SABR after compiling. The FORTRN.TM file is usually deleted by SABR unless the following option is specified.

/K Keep the file FORTRN.TM as a permanent file.

In addition to this option, any options to SABR may be included in the command string to the Compiler. SABR outputs a file in relocatable binary format into the specified binary output file. If a binary output file is not specified and the /L or /G option (to load binary) is, then the binary output goes to a file FORTRL.TM on device SYS: Otherwise, a null binary output file indicates no binary output is to be generated. A SABR listing is not generated if a listing output is not specified.

FORTRAN will only assemble one main program or subroutine per call. A job with multiple programs in it must be run by compiling each routine separately and combining them with the Linking Loader.

Examples of command strings:

Example 1

```
.R FORT
*DTA1:TEST /G
```

The input file TEST.FT (or TEST) on DTA1: is compiled, output into FORTRN.TM on device SYS:, then SABR is called. SABR uses FORTRN.TM as input and outputs the assembled file into FORTRL.TM, deleting the old FORTRN.TM. The Linking Loader then loads FORTRL.TM, deletes FORTRL.TM upon loading, and sends control to the location labeled MAIN (the beginning of the main program).

Example 2

```
*,,LPT:←BLAH/L/M
```

compiles and assembles file DSK:BLAH.FT (or BLAH), putting binary in SYS:FORTRL.TM, calls the Linking Loader, loads and deletes SYS:FORTRL.TM, puts a full loading map on device LPT:. The Loader then asks for another command string. If the line had terminated with an ALT-MODE, control would have returned to the Keyboard Monitor after loading.

Example 3

```
*BINARY,LPT:/N←MATRIX.AB/K
```

The input file MATRIX.AB on DSK: is compiled and output into SYS:FORTRN.TM. SABR is called and it assembles SYS:FORTRN.TM, putting the relocatable binary output into DSK:BINARY.RL and the symbol table only (see SABR section for details) on device LPT:. The /K option will cause SYS:FORTRN.TM to be kept as a permanent file.

Example 4

```
*DTA5:SOURCE /L
```

In this line to FORTRAN, /L means load but do not go.

Example 5

```
*DTA5:LIB40/L/G
```

In this line to the Linking Loader, /L means load file DTA4:LIB40 as a Library. /G means go (start execution).

The following error messages have been added to the PS/8 version of FORTRAN (all error messages described in Chapter 15 of Programming Languages are also valid):

Message	Explanation
I/O	A device handler has signalled a fatal I/O error
NO ROOM FOR OUTPUT	The file FORTRN.TM cannot fit on device SYS:.
SABR.SV NOT FOUND	Self-explanatory. The compiler will not run without SABR.
NO END STATEMENT	The input to the compiler has been exhausted.
COMPILER MALFUNCTION	The meaning of this message has been extended to cover various "impossible" monitor errors.

CHAPTER 5

INTERFACING WITH THE USER SERVICE ROUTINE (USR)

The heart of the PS/8 system is the program called the User Service Routine (USR). USR handles all directory operations for the PS/8 system, such as calling device handlers into core, creating files,searching file directories,decoding command strings fromthe Teletype and chaining to SAVEd programs. An important aspect of PS/8 is that the user program may call upon USR for these services as well, using standard subroutine calls.

USR in turn calls upon the device handlers, that is, page-relocatable subroutines which accept a standard calling sequence and perform input and/or output operations on a device. Any device can be interfaced to the PS/8 system or a user program by means of a device handler. Thus, user and system program can be written without concern for specific input/output devices, since the details of device operation are coded in the device handlers.

In order to write a program which interfaces with USR, the standard calling sequences for the device handlers and the USR subroutines must be used.

5.1 THE DEVICE AND FILNAM PSEUDO-OPS

In specifying arguments to the USR, it is sometimes necessary to indicate device names or file names. It is for this purpose that two pseudo-ops, named DEVICE and FILNAM, have been added to the instruction repertoire of the PAL-8 Assembler. Both of these pseudo-ops, when used, must be followed by a single blank or tab character and a name. For the DEVICE pseudo-op, the name

should be a legal PS/8 device name; two words are generated by the pseudo-op. The FILNAM pseudo-op takes a legal PS/8 file name; four words are generated. The name should be followed by one of the characters blank, tab, or carriage-return. For example,

```
DEVICE  DTA7
  FILNAM  PIP.SV   /NAME OF SYSTEM PROGRAM "PIP"
```

5.2 CALLS TO THE USR

Location 7700 in field 1 is the entry point to USR. Upon execution of a JMS 7700 with the instruction field = 10, user locations 10000-11777 are written out onto the system device (if necessary) and USR is read into these locations. USR cannot be called from locations 10000-11777 because these locations will not be in core while USR is fetching arguments. USR fetches the request number fromthe location following the JMS 7700.

The monitor performs the requested task, fetching and storing additional arguments as necessary. It then (unless otherwise directed) reads the dumped user locations back and resumes execution of the user program.

5.2.1 Generalized Calling Sequence

The generalized calling sequence for requests to USR is as follows.

```
TAD argument      (Where argument is data to be passed to
                   USR - applicable in only some cases.)
CDF n             (Where n is the number of the current
                   instruction field.)
CIF 10
```

JMS I (n	(Where n is 200 if the USR has been locked in core and not yet dismissed and n=7700 otherwise. See the description of the USR request LOCK USR IN CORE. This word sends control to the USR.)
request number	(Where the request number, an integer from 1 to 17_8 indicates the function to be performed.)
argument . . . argument	(The next few words contain arguments whose number and type depend on the function to be performed
error return	(If an error return is possible for this request, then this location is an error return. Otherwise, it is the normal return.)
normal return	(If an error return is possible for this request, this location is the normal (non-error) return.)

5.2.2 Request Numbers

The request numbers and their USR functions are summarized below.

Request Number	Function
1	FETCH HANDLERS: Fetch handlers into core if not already in core and pass entry point address to the calling program.
2	LOOKUP PERMANENT FILE: Pass the starting block number and the length of the specified permanent file on the specifield device to the calling program.
3	ENTER OUTPUT FILE: Create a tentative file on a specified device for output.
4	CLOSE OUTPUT FILE: The currently open tentative output file is closed to further output and becomes a permanent file. Any previous permanent file of the same name on the device is deleted and unused blocks become free files.

Request Number	Function
5	COMMAND DECODE: The Command Decoder is called and passes I/O specifications and options typed on the Teletype to the calling program.
6	CHAIN TO PROGRAM: Load the specified program from the system device and start it.
7	SIGNAL USER ERROR: Print on the Teletype the error message indicated by the calling program.
10	LOCK USR IN CORE: USR is loaded into core and is not overlaid again until the USR DISMISS command is given.
11	USR DISMISS FROM CORE: USR is overlaid with the user's core (unless the Job Status Word specifies otherwise).
12	ASCERTAIN IF HANDLER IN CORE: Ascertains whether a given device exists, if so whether its handler is in core, and if so what the entry point is. If the handler is not in core, it is <u>not</u> loaded by this request.
13	RESET SYSTEM TABLES: The USR tables are set to indicate that all device handlers (except the system device handler and its co-resident handlers) are not in core. This request may also be used to delete all active tentative files.
14-17	These request numbers are reserved for future expansion.

The specific calling sequence for each USR request and the details of its function are next discussed. It is assumed that the instruction field and data field are already set up for the call (see "Generalized Calling Sequence").

5.2.3 Fetch Handler

This request may be made by two different calling sequences:

1. The number of the I/O device to be used is passed to USR in the accumulator (often abbreviated AC).

2. The device number is obtained by searching the User Device Name table and, if no match is found, the System Device Name table. The number of the device is the index number of its match in the first table in which it is found.

The first calling sequence:

TAD device number	(Where the number becomes the low-order four bits of the AC.)
CALL,JMS I (n	(Where n=200 if USR is locked in core and n=7700 otherwise.)
1	(Request number)
ENTRY, loading address	(The loading address of the handler has the following format:

Bit No.	Contents
0-4	page number in which to load handler (i.e., 1 here means start at location 200)
5-10	ignored
11	0 if space available for handler is one page, 1 if space available for handler is two pages.

On return, the USR overlays this location with the address of the entry point of the handler.)

error return	(If the device name is 0, if the device does not exist, or if the handler will not fit in the allotted space, control returns here.)
normal return	(At this point the handler and any co-resident handlers, which share the same page, are in core in field 0, and any handlers which were formerly resident in the same page (or pair of pages for a two-page handler) are marked as not being in core.)

The second calling sequence:

```
        CLA                 (Not necessary if AC already cleared)

CALL,   JMS I (n            (Where n=200 if the USR is locked
                             in core and n=7700 if otherwise.)

          1                 (Request Number)

DEV,    DEVICE device name  (Where either the user or perma-
        (takes two locations) nent device name is given as the
                             argument of the DEVICE pseudo-op.
                             The second word (DEV+1) will be
                             overlaid by the USR with the de-
                             vice number for future reference
                             by the calling program.)

loading address             (of the device handler.  Opera-
                             tion is the same as for the
                             first calling sequence.)

error return                (If the device name is not found
                             in either table, if the device
                             does not exist, if the device name
                             is 0, or if the handler will not
                             fit in the allotted space, con-
                             trol returns here.)

normal return               (The same as for the first se-
                             quence.)
```

5.2.4 Lookup Permanent File

(The device handler for the file's device must be in core when this request is made.)

```
TAD  device number          (The device number is in the
                             low-order bits of the AC.  If
                             bits 8-11 of the AC are zero,
                             a system error is signalled
                             and control returns to the
                             Keyboard Monitor.

CALL, JMS I (n              (Where n=200 if the USR is
                             locked in core and n=7700
                             otherwise.)

   2                        (Request number)
```

pointer to file name	(If the device is a directory device, the file named (by the FILNAM pseudo-op or equivalent) in the specified location is searched for among the permanent files and if found its starting block number overlays this location. For a non-directory device, this location is cleared to zero.
reserved for file length	(For a directory device this location is overlaid with the file length. For a non-directory device this location is cleared.)
error return	(If the file is not found for a directory device.)
normal return	(The starting block number and file length are now available to a calling program for a directory device. This return is also taken for a non-directory device.)

5.2.5 Enter Output File

(The device handler for the file's device must be in core when this request is made.)

TAD device number and length	(The device number is in bits 8-11 of the AC. For a directory device the length of the desired file is indicated in bits 0-7. If the length is zero, the largest free file on the device is used. If the length is non-zero, the smallest free file at least as large as the length is used.)
JMS I (n	(Where n=200 if the USR is locked in core and n=7700 otherwise.)
3	(Request number)
pointer to file name	(For a directory device this location is overlaid with starting block number of the specified file. For a non-directory device this location is cleared.)

reserved for file length	(For a directory device this location is overlaid with the length of the created tentative file. For a non-directory device this location is cleared.)
error return	(If the device is read-only, if an active tentative file already exists on the device, or if there is not enough room for the file on the device.)
normal return	(At this point for a directory device an active tentative file is open on the device and the file's starting block number and actual length is available to the calling program for reference. Control is sent here for a non-directory device as well.)

5.2.6 Close Output File

(The device handler for the file's device must be in core when this request is made.)

```
TAD  device number
JMS I (n
```

	(Where n=200 if the USR is locked in core and n=7700 otherwise.)
4	(Request Number)
pointer to the file name to be deleted	(For a directory device the file pointed to here is deleted)
length of new permanent file	(This location should be loaded with the number of records which were actually used for the file. If the file has exceeded the space allotted in the ENTER OUTPUT FILE operation an error message is printed and control returns to the Keyboard Monitor.)
error return	(If no active tentative file or permanent file of the specified name exists on the device, then control returns here.)

normal return	(At this point if the device is a directory device, the new permanent file has replaced the old permanent file (if one existed) and the directory has been updated to reflect this change.)

NOTE

The normal way to use the ENTER and CLOSE operations is to point to the same file name in both calling sequences. This results in a very protective mode of operation - i.e., if a file A.B exists on a device and you ENTER a file A.B, the old file is not deleted. If your program bombs out or is interrupted before it can execute the CLOSE operation, the old file A.B is still intact and the new one will disappear in time. If in the arguments of the CLOSE operation you specify the file name A.B, then the old file will be deleted at this point and the new one will take its place. When using the ENTER and CLOSE operations in any other manner you must exercise extreme caution, because it is possible to wind up with two permanent files in the same directory with the same name, a situation for which the Monitor operations are not defined.

5.2.7 Command Decode

JMS I (n	(Where n=200 if the USR is locked in core and n=7700 otherwise.)
5	(Request number - interpret command string typed on the Teletype.)
default extension	(If the user of the Teletype does not specify an extension for an input file the default extension is added to the filename. If the file is not found, a search for a file without an extension to the filename is made.)
return	

Example

A program calls the Command Decoder and specifies .XY as the default extension to be used for filenames. The user types:

```
BLAH A.BC,DTA1:HELLO,GOODBYE
```

The input files that the system program sees are:

1. DSK: A.BC
2. DTA1:HELLO.XY if it exists; otherwise DTA1:HELLO (if both do not exist an error is printed).
3. DTA1:GOODBY.XY if it exists; otherwise DTA1:GOODBY.

The output file is DSK:BLAH. (Default extensions do not affect the output side of the list.)

5.2.8 Chain to Program

JMS I (n	(Where n=200 if the USR is locked in core and n=7700 otherwise.)
6	(Request Number)
starting block number	(This starting block number of the program chained to can be obtained by a Lookup Permanent-File operation. The program on the system device which begins at the specified starting block is loaded into core. The program is started at an address one greater than the starting address specified in the core control block.)
(no return)	

5.2.9 Signal User Error

JMS I (n	(Where n=200 if the USR is locked in core and n=7700 otherwise.)
7	(Request Number)
error number	(The message "USER ERROR n AT LOCATION xxxxx" is printed on the Teletype. The address xxxxx points into the offending calling sequence, not to its beginning. The integer n has the following meanings:

n	Error
1	User specified CLOSE operation with a length larger than the length given to him- this may indicate that the user has destroyed some of his permanent files.
2	I/O error reading or writing a directory block.
3	LOOKUP, ENTER, or CLOSE attempted on device whose device handler was not in core.
4	Illegal operation specified to Monitor, or device number Ø specified in a LOOKUP, ENTER, or CLOSE operation.

5.2.10 Lock USR in Core

```
JMS I (7700          (Call USR).
    10               (Request Number)
return
```

This command causes USR to come into core and stay there unless it is dismissed by the Monitor dismiss command. When USR is in core, it should be called by a JMS to location 200 in field 1. All other aspects of calling USR are the same.

If USR is called by a JMS I (7700 while locked in core, the core which it saved upon coming in may be destroyed. While USR is in core, locations 20-37 in field 1 are available for user scratch area.

5.2.11 USR Dismiss from Core

```
JMS I (0200          (This command is a no-operation
                     if you mistakenly write JMS I
                     (7700.)
```

11 (Request Number.)

return

This command also restores the user's core (unless the Job Status Word specifies otherwise.) After USR has been dismissed, it is called by JMS I (7700 again instead of JMS I (200.

5.2.12 Ascertain if Handler in Core

This request may be made by two different calling sequences:

1. The number of the I/O device to be used is passed to USR in the AC.
2. The device number is obtained by searching the User Device Name table and, if no match is found, the System Device Name table. (The number of the device is the index number of its match in the first table in which it is found.)

The first calling sequence:

TAD device number	(Where the number is the low-order four-bits of the AC.)
CALL, JMS I (n	(Where n=200 if the USR is locked in core and n=7700 otherwise.)
12	(Request Number)
ENTRY, loading address	(The loading address of the handler has the following format:)

Bit No.	Contents
0-4	page number in which to load handler (i.e., 1 here means start at location 200)
5-10	ignored
11	0 if space available for handler is one page, 1 if space available for handler is two pages.

	On return, the USR overlays this location with the address of the entry point of the handler if the handler is in core. If the handler exists out of core, the USR clears this location.)
error return	(If the device name is 0, or if the handler does not exist, control returns here.)
normal return	(If the device exists but is not in core, control returns here.)

The second calling sequence:

CLA	(Not necessary if AC already cleared.)
CALL, JMS I (n	(Where n=200 if the USR is locked in and n=7700 otherwise.)
12	(Request Number).
DEV, DEVICE device name	(Where either the user or permanent device name is given as the argument of the DEVICE pseudo-op. The second word (DEV+1) will be overlaid by the USR with the device number, for future reference by the calling program.)
loading address	(of the device handler. Operation the same as for the first calling sequence.)
error return	(If the device name is not found in either table or if the handler does not exist, control returns here.)
normal return	(The same as for the first sequence.)

5.2.13 Reset System Tables

JMS I (n	(Where n=200 if the USR is locked in core and n=7700 otherwise.)
13	(Request Number)

The return, if all the active tentative files are to be deleted, otherwise Ø.	(If this location is non-zero, all files upon which output operations have not yet been terminated will be destroyed. In any case, all handlers other than for the system device, PTR, and PTP will not longer be in core.)
The return if previous word Ø.	(At this point, all the handlers other than for the system device, PTR, and PTP are no longer in core, though the active tentative files are intact.)

5.3 DEVICE HANDLERS

Once a device handler is in core (loaded by USR call 1), it may be called to perform read and/or write operations on the device using the following standard calling sequence.

CDF n	(Where n is the number of the current instruction field.)
CIF 0	
JMS I entry	(Where entry is a location containing the address of the entry point of the device handler.)
function word	(Where the function word has the following format)

Bit Number	Contents
0	0 if a read operation 1 if a write operation
1-5	the number of pages to be transferred
6-8	the field in which the buffer resides
9-11	device dependent function which need not be programmed

buffer address	(of the location at the beginning of the buffer)

starting block number	(Block number at which the operation will take place. For a call to the PTR, if this location is 0, the handler will print an up-arrow and wait for a character to be typed.)
error return	(Control returns here if an error occurs. The accumulator will be negative if the error is fatal, (e.g., parity error, impossible request) and positive if the error is non-fatal (e.g., paper tape reader ran out of tape.)
normal return	(Control returns here if the transfer was successfully completed.)

For example, the calling sequence to the high-speed reader device handler from instruction field 10 might be as follows:

```
CDF 10          /SET DATA FIELD TO 10
CIF 0           /INSTRUCTION FIELD NOW 0
JMS  I  INPUT   /SEND CONTROL TO DEVICE HANDLER
0200            /FUNCTION WORD INDICATES A READ
                /OPERATION OF TWO PAGES (256 WORDS)
                /INTO FIELD 0.
3000            /BUFFER BEGINS AT LOCATION 3000
                /OF FIELD 0.
30              /READING BEGINS AT RECORD 30 (MEANINGLESS
                /FOR PAPER TAPE READER).
HLT             /IF AN ERROR OCCURS, HALT
 .              /OTHERWISE, CONTINUE WITH PROGRAM
 .
 .
```

All device handlers are loaded into field 0. The device handler for the system device is always resident at location 7607 in field 0. Before using the USR requests to Lookup, Enter, or Close a file on a device, its device handler must be in core.

5.4 DEMONSTRATION PROGRAM

What follows is a user program which interfaces with the USR to handle I/O for the task of copying a tape read by the high-speed reader onto DECtape unit 1. The program first locks the USR in core and loads the device handlers for the high-speed reader and DECtape 1 into core. Next, it creates an output file named YES.NO on the DECtape and reads a record from the reader into core. Then the record is written on DECtape, counters are incremented and the next record is read and written, etc.

If a non-fatal error occurs (e.g., end of tape), the error handling routine will write the last record of output on the DECtape and close the output file. The program then rings the Teletype bell and returns to the Keyboard Monitor.

```
        /THIS PROGRAM TRANSFERS A TAPE FROM DEVICE "PTR"
           /(USUALLY THE HIGH SPEED PAPER TAPE READER)
           /TO A FILE NAMED "YES.NO" ON DEVICE DTA1
           /(USUALLY DECTAPE 1)
           /THIS PROGRAM IS NOT RESTARTABLE BECAUSE THE
           /MONITOR CALLS WHICH IT USES ARE ALTERED
           /BY THE MONITOR AND NOT RESTORED.

           *200
           CIF 10              /SET INST FIELD TO 1
           JMS I (7700         /CALL USR
           10                  /LOCK USR INTO CORE
           CIF 10
           JMS I (200          /CALL USR AT ITS LOCKED-IN
                               /ENTRY POINT
           1                   /FETCH HANDLER
           DEVICE PTR

/ASSEMBLER WILL GENERATE DEVICE NAME HERE
INPUT,     1000                /LOAD HANDLER IN PG 1000 IF NOT
                               /IN CORE OVERLAYD WITH HANDLER ENTRY
           HLT                 /ERROR RETURN - PTR NOT FOUND
                               /OR HANDLER IS 2 PAGES LONG
           CIF 10
           JMS I (200          /CALL USR AGAIN
           1
           DEVICE DTA1

DTA=.-1                        /ASSEMBLER WILL PUT DEVICE#
                               /OF "DTA1" IN LOC "DTA"
OUTPUT,    2001                /LOAD HANDLER INTO PAGES
                               /2000 AND 2200 (OVERLAYED WITH REAL
                               /ENTRY POINT)
           HLT                 /DTA1 DOES NOT EXIST??!!
           CIF 10
           TAD DTA             /DEVICE # OF "DTA1" INTO AC
           JMS I (200
           3                   /ENTER OUTPUT FILE
FSTRT,     POINTR              /POINTS TO FILE NAME - OVERLAYED WITH
                               /STARTING BLOCK # OF CREATED FILE
FLEN,      0                   /OVERLAYED WITH LENGTH OF CREATED FILE
           HLT                 /ERROR RETURN

/WE ARE NOW READY TO DO THE TRANSFERING OF DATA
LOOP,      JMS I  INPUT        /CALL "PTR" DEVICE HANDLER
           0200                /TRANSFER 2 PGS TO FIELD 0
           3000                /BUFFER IS AT LOC 3000
IREC,      0                   /INPUT BLOCK NUMBER
           JMP INERR           /INPUT ERROR DETECTED
           TAD FSTRT
           TAD CLENGT          /COMPUTE BLOCK TO WRITE INTO
           DCA ECCH
           JMS I OUTPUT        /CALL "DTA1" HANDLER
           4200                /WRITE 2 PAGES FROM FIELD 0
           3000                /BUFFER IS AT 3000
```

```
ECCH,   0               /BLOCK TO OUTPUT INTO
        HLT             /OUTPUT ERROR
                        /THERE ARE NO SOFT OUTPUT ERRORS
        ISZ IREC        /BUMP THE INPUT BLOCK #
/(WE SHOULD DO THIS DESPITE THE FACT THAT PTR
/IS NOT FILE STRUCTURED!)
        ISZ CLENGT      /BUMP COUNT OF RECORDS OUTPUT
        ISZ FLEN        /BUMP COUNT OF BLOCKS IN OUTPUT FILE
        JMP LOOP        /LOOP IF EVERYTHING IS OK
        HLT             /OUT OF ROOM FOR OUTPUT

INERR,  SPA CLA         /WHAT KIND OF ERROR WAS IT?
        HLT             /BAD KIND
        TAD FSTRT       /OUT OF TAPE, COMPUTE OUTPUT
        TAD CLENGT      /BLOCK FOR LAST RECORD
        DCA ECCH2
        JMS I OUTPUT
        4200
        3000            /WRITE LAST BUFFER OUT
ECCH2,  0
        HLT             /ERROR
        ISZ CLENGT      /BUMP COUNT OF BLOCKS OUTPUT
        CIF 10
        TAD DTA         /GET THE DEVICE # OF "DTA1" IN AC
        JMS I (200
        4               /CLOSE OUTPUT FILE
        POINTR          /POINTS TO FILE NAME - SHOULD BE
                        /SAME NAME AS THAT USED IN
                        /THE "ENTER" OPERATION
CLENGT, 0               /NUMBER OF BLOCDS OUTPUT
        HLT             /SOMETHING BAD HAPPENED

        TAD (207
        6046            /RING THE BELL
        6041
        JMP .-1         /WAIT FOR TELETYPE TO STOP
        CLA
        JMP I (7600     /RETURN TO KEYBOARD MONITOR

POINTR, FILNAM  YES.NO

/ASSEMBLER WILL GENERATE 4-WORD FILENAME HERE
```

CHAPTER 6

PRELIMINARY AND SUPPLEMENTARY INFORMATION

The information contained in the following sections is preliminary and is subject to change without notice. Much of what follows is not necessary for the average user. It is documented for use by the advanced programmer who may wish to alter the system internally.

6.1 PS/8 AND THE DISK MONITOR COMPARED

PS/8 differs from the 4K Disk Monitor in the following ways:

- PS/8 has 8K FORTRAN and a system ODT while the DM system has 4K FORTRAN and a system DDT.
- PS/8 makes more effective use of extended memory than the DM system. For example, core image files may include code in more than one memory field.
- PS/8 is truly device independent. The device handlers are a part of the system and are loaded by the User Service Routine.

  In the Disk Monitor (DM system) the handlers for devices other than the system device must be coded into the program.
- In PS/8 it is possible to specify variable length reads and writes to the device handlers. The DM system transfers one block at a time.
- PS/8 files occupy contiguous blocks and the file directory entries are listed simply in order of occurrence of files on the device. DM files may be segmented and the directory includes a storage allocation map.

- The USR performs directory operations on file structured devices (Directory Lookup, Enter, Close.) DM system programs must perform these operations themselves.
- PS/8 will RUN and SAVE programs on devices other than system device.
- PS/8 is capable of program chaining.

6.2 LIMITATIONS OF THE PRELIMINARY VERSION

The PS/8 system which is being made available now is still a preliminary version and does not have all the features of the completed version as yet. What follows is a list of the present limitations on PS/8:

1. As yet, building the system is simply a process of reading in all the tapes of the system since no configuration editor exists. Device handlers other than those provided as a part of PS/8 can not yet be integrated into the system. Since the system is not edited to fit the configuration, reference to a non-existent device will cause the computer to enter an infinite loop waiting for the device to respond.
2. Various option which the Command Decoder passes to the system programs are not yet implemented:

Option	Not Yet Implemented for
=n	PIP
/S	PIP
/G	PIP
/L	CONVRT
/O	ABSLDR

3. The Card Reader Handler (CDR) is not useable.
4. Putting source and binary output on the Teletype punch is not recommended as it may come out interspersed with error messages and system prompt characters. Listings may be put on the punch. If you wish to load absolute binary tapes from the Teletype with ABSLDR, you should use the /S option to avoid confusion. (Type CTRL/Z after all the tapes are read.) Relocatable binary tapes should not be read from Teletype readers since they may contain CTRL/Zs and CTRL/Cs.

5. When loading a multi-segment absolute binary tape from the high-speed reader using ABSLDR, use the /S option.

6. File directories as yet are one block instead of six blocks. As a result the maximum number of files which can exist on a directory device will be between 34 and 48, depending on the fragmentation of the storage area.

7. Avoid using files longer than 256 blocks.

6.3 DIRECTORY STRUCTURE

File directories list the files in their order of occurrence on the device, since each file is in one contiguous area on the storage device. The directory occupies up to six blocks starting at block one of the device. Each block has the same format: a directory-block header of five words followed by entries for each file giving the file name and length. Each directory block is linked to the following block (if one exists) by a word in the directory-block header.

The organization within the directory block header is as follows.

Word	Contents
0	The number of entries in this directory block expressed as a negative number.
1	The starting block number for storage of files listed in this block.
2	Link word: The block number of the next directory block if there is a next. If this is the last directory block in use, this word contains zero.
3	Flag word: The address of the file entry for the tentative file if one exists. (Only one tentative file can exist on a device at a time.)
4	The number of extra words per directory entry. The number is specified as an option to PIP using the construction "=n" where $n \leq 4$. The directory entries may then be altered to include information as to date and author of the file.

From words 5 to 377 are the file entries of the directory block. The three different types of files -- permanent, tentative, and free - have three different entry forms, all loosely a file name followed by the file length.

1. A permanent file entry consists of a four-word name (the name is six characters with an extension of two characters) followed by a one-word length expressed as a negative number.

2. A tentative file entry consists of a four-word name followed by a one-word length of zero. The zero indicates that the length is not yet set, because the file is not yet closed to further output. A tentative file is always followed by a free file, whose entry is described next.

3. A free file entry consists of one word of zero (indicating no name) followed by a one-word length expressed as a negative number. Such an entry indicates the space available for output.

6.4 BUILDING AND STARTING A PS/8 SYSTEM

1) The standard DEC Binary Loader should be loaded into field 1. Use the following Bootstrap to move the Loader to the proper memory locations. Starting at location 0002:

```
0002  1410
0003  3411
0004  2007
0005  5002
0006  7402
0007  7600
0010  7577
0011  1577
```

2) Place the Binary tape for the PS/8 system in the reader. The number of the RK8 system tape is DEC-P8-MSDA-PB. Load address 011777; depress START. The PS/8 system tape will be read at this point. (NOTE: The system tape is composed of two binary tapes. The computer will halt after each segment is read in.) After the first segment has been read, the computer will halt with a Ø AC. Depress CONTinue to read the second segment If the AC is non-zero at any halt, a checksum error has occurred, and it is necessary to restart the building procedure.

Load address with 4000 (data field and instruction field set to Ø), and depress START. Some system I/O will take place, after which the machine will halt with -1 (7777) in the AC.

3) At this point, place the Command Decoder binary tape in the reader. (Tape No. DEC-P8-SWXA.) Make certain any switch register switches are off, and depress CONT. The binary tape will be read in, and the machine will halt with AC=0. Again, load address 4000 and depress START. System I/O will take place, and the Monitor will respond with a . (dot). This signifies that the Kayboard Monitor is in core, and is ready for input. At this point, the system Absolute Binary Loader will be used to load the system program, according to the following specifications.

Program Name: PIP Tape No. DEC-P8-PWXA
Place the PIP binary tape into the reader. Load as follows:

```
.R ABSLDR(CR)              CR=carriage return
*PTR:=13000(89)$           $=ALT MODE
```

When ↑ appears, strike any character on the keyboard, and the tape will be read in. The Monitor responds with . (dot).

```
.SAVE SYS PIP(CR)
```

At this point, PIP is written onto the system device.

Program Name: FORTRAN Tape No. DEC-P8-KFXA
Place the FORTRAN binary tape into the reader, Load as follows:

```
.R ABSLDR(CR)
*PTR:/S$
```

When ↑ appears, strike a keyboard character to read in the tape. Monitor responds with . (dot).

```
.SAVE SYS FORT(CR)
```

At this point, FORT is written onto the system device.

Program name: SABR Tape No. DEC-P8-ARXA
Place the SABR binary tape into the reader. Load as follows:

```
.R ABSLDR(CR)
*PTR:/S$
```

When ↑ appears, strike a keyboard character to read in the tape. The tape is read, and Monitor responds with . (dot).

```
.SAVE SYS SABR(CR)
```

At this point SABR is written onto the system device.

Program Name: Linking Loader Tape No. DEC-P8-LLXA

Place the Loader binary tape in the reader. Load as follows:

```
.R ABSLDR(CR)
*PTR:(9)$
```

When ↑ appears, strike a keyboard character to load the tape. Monitor responds with a . after loading the tape.

```
.SAVE SYS LOADER(CR)
```

At this point, the Loader is written onto the system device.

Program Name: LIB8 Tape No. DEC-P8-SFXA

Place the LIB8.RL tape in the reader.

NOTE: It is necessary to place the tape in the reader so that the first non-zero character to be read is directly to the right of the read head. If this restriction is not followed, the library may not be loaded properly. Load as follows, using PIP and the LIB8 setup tape.

```
.R PIP(CR)
*SYS:LIB8.RL TTY:,PTR:/I(CR)
```

Notice that this expects a file from the TTY first. The machine will appear to hang up. At this point, type in ↑ Z (CTRL/Z). When ↑ appears, strike a keyboard character to read in the LIB8.RL tape. After reading, PIP responds with *

```
*↑C
```

Type CTRL/C in response to the *. Monitor responds with a . (dot).

At this point, place the LIB8 setup binary in the reader. Tape No. DEC-P8-SYXA. Load as follows:

```
.R ABSLDR(CR)
*PTR:/G$
```

When ↑ appears, strike a keyboard character to read the tape. At this point, LIB8.RL is written, the bell is rung to signify completion of the operation, and the Monitor returns with . (dot).

Program Name: EDITOR Tape No. DEC-P8-ESAA

Place the EDITOR binary tape in the reader. Load as follows:

```
.R ABSLDR(CR)
*PTR:(9)$
```

When ↑ appears, strike a keyboard character to read the tape. The EDITOR will be read in, and Monitor responds with . (dor).

```
.SAVE SYS EDIT(CR)
```

At this point, the EDITOR is written on the system device.

Program Name: PAL8 Tape No. DEC-P8-ASXA

Place the PAL8 binary tape in the reader. Load as follows:

```
.R ABSLDR(CR)
*PTR:(9)$
```

When ↑ appears, strike a keyboard character to load the tape. The tape will be read, and Monitor will respond with . (dot).

```
.SAVE SYS PAL8(CR)
```

PAL8 will now be written on the system device.

Program Name: CONVRT Tape No. DEC-P8-SUTA

Place the CONVRT binary in the reader. Load:

```
.R ABSLDR(CR)
*PTR:(9)$
```

When ↑ appears, strike a keyboard character to load the tape. After loading, Monitor responds with . (dot).

```
.SAVE SYS CONVRT(CR)
```

CONVRT is now written on the system device.

That completes the building of the present PS/8 system.

Starting PS/8

A PS/8 system using a DECtape as the system device may be started by toggling in the DECtape Bootstrap Loader:

Location	Contents
7613	6774
7614	1222
7615	6766
7616	6771
7617	5216
7620	1223
7621	5215
7622	0600
7623	0220
7754	7577
7755	7577

and then starting the Loader at location 7613.

A PS/8 system using an RK8 disk as the system device may be started by toggling in the Bootstrap Loader for the disk:

Location	Contents
0030	6733
0031	5031

Load address and start at location 30.

6.5 USING THE SYSTEM ODT

The ODT available under PS/8 is a debugging tool similar to the stand-alone ODT program (DEC-08-COCO). It has the following properties and restrictions: (The user should be familiar with stand-alone ODT before reading this part.)

1. It does not occupy any core outside of the two system pages (07600 and 17600).

2. It requires locations 4, 5, and 6 in any field in which a breakpoint will be used for breakpoint processing.

3. ODT wipes out the USER DEVICE NAME table. Therefore you should not use it to debug programs which require user device names.

4. ODT should not be used to debug programs which use interrupts.

5. Do not set breakpoints:

   in the Monitor

   in device handlers

   between a CIF and the following JMP instruction

6. Addresses can be five digits long on input and will always be printed as five digits.

7. The following commands are available:

Command	Meaning (if preceded by number N)	Meaning if not preceded by number
/	Open and examine location N.	Re-examine last examined location
CR	Deposit N in opened location, (if one exists); close location	Close location
LF	Deposit N in open location, close location, open and examine location + 1.	Open and examine location +1.
;	Deposit N in open location, close location, open location + 1	Close location, open location+1
← (SHIFT/O)	Open and examine location N (same as /)	Open location pointed to by last-examined location
↑ (SHIFT/N)	Open and examine location addressed by N	Open and examine location addressed by last examined location
B	Set breakpoint at location N	Clear breakpoint
C	Continue from breakpoint N times	Equivalent to 1C.
G	Start program at location N	?
A	N ignored	Display AC as of last breakpoint
L	N ignored	Display Link as of last breakpoint

Command	Meaning if preceded by number N	Meaning if not preceded by number N
M	N ignored	Display Search Mask. Next location is the lower limit of search, next after that is the upper limit.
D	N ignored	Display Data Field as of last breakpoint
F	N ignored	Display Data Field for indirect (↑ and ←) references and searches.
T	N ignored	Display TCF instruction on ODT. (This should be a NO-OP if the TTY flag is to be on.)
W	Search field F under Mask M for N from location (M+1) to location (M+2); print all matches.	Same as 0W.

6.6 THE SUPPLIED DEVICE HANDLERS

Device	Normal Operation	Special Operation for Record 0	Terminating Conditions	User Interaction via TTY
PTR:	Read characters from paper tape and pack them into a buffer.	Type an up-arrow on the TTY and wait for user to mount his tape and type a character.	End of paper tape causes CTRL/Z to be entered, the buffer is padded with zeros and the non-fatal error return is taken.	Typing CTRL/C while tape is moving returns to Monitor.
PTP:	Unpack characters from buffer and punch them on paper tape.	None	None	CTRL/C returns to Monitor.
LPT:	Unpack characters from buffer and print them on line printer. Simulate any format control characters not in the private hardware.	Print a form feed.	CTRL/Z in buffer terminates printing form feed.	CTRL/C returns to Monitor.

Device	Normal Operation	Special Operation for Record 0	Terminating Conditions	User Interaction Via TTY
CDR:	Not usable in present version.			
TTY:	Transfer characters between buffer and Teletype.	None	Input: Reading a CTRL/Z from the TTY causes CTRL/Z to be put in the buffer, the buffer padded with zeros and the non-fatal error return taken. Output: seeing a CTRL/Z in the buffer or seeing a CTRL/O from the keyboard causes a normal return to be taken.	CTRL/Z terminates input. CTRL/O terminates printing of current output buffer. CTRL/C returns to Monitor.
DTAn:	Transfers words via data break between buffer and DECtape.	None	None	None
SYS:	Transfers words via data break between buffer and systems device.	None	None	None

Device	Error Condition	Type: AC<0 for hard (fatal) error	Type: AC≥0 for soft (recoverable) error
PTR:	Ran out of tape		Soft
	Attempt to write	Hard	
PTP:	Attempt to read	Hard	
LPT:	Attempt to read	Hard	
	Printer error (power low)	Hard	
CDR:	Not yet available		

Device	Error Condition	Type: AC 0 for hard (fatal) error	Type: AC 0 for soft (recoverable) error
TTY:	CTRL/Z typed during input operation (tries three times first).		Soft
	Write lock error during output operation (tries three times).	Hard	
	Unit not selected - handler hangs until unit is selected.		Not an error
SYS:	Same as DTAn for DECtape system. For RK8 disk system:		
	Track address error (tries recalibration three times)	Hard	
	All other errors (tries three times)	Hard	

6.7 WRITING DEVICE HANDLERS

All device handlers must be page relocatable - i.e., they must be able to execute in any page (except page 7600 and possibly page Ø). This means that they must not contain any fullword references to locations in the handler itself. In certain programs this is a restriction, but it can usually be overcome. For instance if a table lookup has to be done it can be done by the following methods:

METHOD 1 Not Page Relocatable	METHOD 2 Page Relocatable	METHOD 3 Page Relocatable
TAD N	TAD N	JMS .+1
TAD BASE	TAD TBASE	BASE1, Ø
DCA TEMP	DCA	TAD N
TAD I TEMP	HLT	TAD BASE
.	.	
.	.	TAD BASE 1
.	.	
BASE, TABLE-1	TBASE,TAD TABLE-1	DCA TEMP
.	.	TAD I TEMP
.	.	.
TABLE,	TABLE,	.
.	.	.
.	.	.
5 locs + 1 temp	5 locs	BASE, TABLE-BASE1-1
		TABLE
		8 locs and 1 temp.

The third method shown is less efficient than the other two, but has the advantage that it can be used in a two-page handler when TABLE might be in a different page from the code.

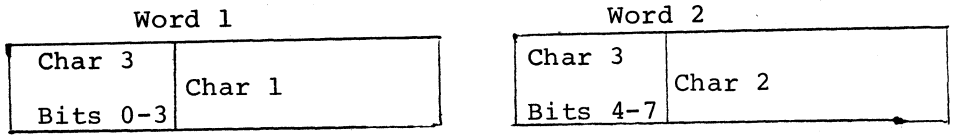
There are two restrictions on the entry point of a device handler. They are:

1. If the handler is a two-page handler, the entry point must be in the first page.

2. If the handler is for a random access device which will be used as a directory device, the entry point of the handler relative to the beginning of its page must not be zero or equal to the address of any other handler of a directory device (other than the system device) relative to its page. EXAMPLE: If you want to write a handler for magnetic tape, and you want the magnetic tape to be a directory device then you should make the entry point of the handler before location 10 of the page because the DECtape handlers have entry points of 10 through 17 in their pages.

The problems involved in writing a handler depend upon whether the device is character-oriented or uses data-break for information transfer. A character-oriented device requires a handler which packs and unpacks characters into and out of the buffer. Also the handler must be capable of interacting easily with the user, who must usually perform such manual functions as typing characters and loading tapes. By comparison, the handler for a data-break device is simple, since setting up the current address, the word count, and the actual transfer is all that is necessary.

The standard character-packing format used in PS/8 is a three-character into two-word format as follows:



This format was chosen because two out of the three characters are visible at a glance, and also because the packing and unpacking routines for this format are smaller than for any other format:

The following is a typical routine used for packing characters into an input buffer:

```
PACK, JMS RCHAR          /INPUT A CHARACTER
      DCA I POINTR
      TAD POINTR
      DCA TEM1
      ISZ POINTR
      JMS RCHAR
      DCA I POINTER
      JMS RCHAR
      RTL
      RTL
      DCA TEM2
      TAD TEM2
      AND (74ØØ
      TAD I TEM1
      DCA I TEM1
      TAD TEM2
      RTL
      RTL
      AND (7400
      TAD I POINTR
      DCA I POINTR
      ISZ POINTR
      ISZ COUNTR
      JMP PACK
        .
        .
        .
        .
```

The following is a typical routine used for unpacking characters from an output buffer:

```
UNPACK, TAD I POINTR
        JMS PUNCH              /OUTPUT A CHARACTER
        TAD I POINTR
        AND (7400
        DCA TEM1
        ISZ POINTR
        TAD I POINTR
        JMS PUNCH
        TAD I POINTR
        ISZ POINTR
        AND (7400
        CLL RTR
        RTR
        TAD TEM1
        RTR
        RTR
        JMS PUNCH
        ISZ COUNTR
        JMP UNPACK
          .
          .
          .
          .
          .
```

A final word of warning - try to make these handlers as foolproof as possible. For example, if a device is read-only the handler should give a hard error on write operations and vice-versa. Also try to guard against the possibility that the user will specify an operation which wraps around core (e.g., write three pages from location 7400) or will specify 0 as a page count (if you can, accept 0 as the same as 40_8 - a full core load).

What follows is a sample handler - it handles the paper tape reader and punch.

```
      7400              *7400
                        /HIGH SPEED PAPER TAPE HANDLER FOR PS/8 MONITOR
                        /PACKS 3 CHARACTERS IN 2 WORDS ON INPUT, UPACKS
                        /ON OUTPUT
                        /PAGE RELOCATABLE
7400  0000    PTP,      0                 /ENTRY FOR PAPER TAPE PUNCH
7401  7320              CLA CLL CML       /SET LINK TO INDICATE PUNCH
7402  4235              JMS PSETUP
7403  1712    PTPLP,    TAD I PTPCA
7404  4304              JMS PTPPCH        /FIRST CHAR IN LOW ORDER 8 BITS
                                          /OF WORD 1
7405  0342              AND PT7400
7406  3313              DCA PTR
7407  2312              ISZ PTPCA
7410  0232    PTP232,   232
7411  1712              TAD I PTPCA
7412  4304              JMS PTPPCH        /SECOND CHAR IN LOW ORDER 8 BITS
                                          /OF WORD 2
7413  0342              AND PT7400
7414  7112              CLL RTR
7415  7012              RTR
7416  1313              TAD PTR
7417  7012              RTR
7420  7012              RTR       /THIRD CHARACTER NOW IN AC
7421  4304              JMS PTPPCH
7422  7700    PT7700,   7700      /CLEARS AC AND NEVER SKIPS
7423  2312    PTPEND,   ISZ PTPCA
7424  0070    PT70,     70        /FOR WE'RE PUNCHING PG 7600 KEEP THIS
                                  /LITERAL
7425  6034              KRS
7426  1311              TAD PTM203
7427  7650              SNA CLA
7430  6031              KSF
7431  5272              JMP PTPISZ
7432  6203    PTPCIF,   CDF CIF 0
7433  5634              JMP I .+1
7434  7600              7600
7435  0000    PSETUP,   0
7436  6214              RDF       /GET FIELD OF CALLING PROC
7437  1232              TAD PTPCIF
7440  3301              DCA PTPXIT        /SET UP RETURN
7441  1600              TAD I PTP
7442  0224              AND PT70
7443  1372              TAD PCDF
7444  3271              DCA PTPCDF
7445  7010              RAR               /GET LINK(1=PTP,0=PTR)
7446  1600              TAD I PTP         /GET FUNCTION WORD
7447  2200              ISZ PTP
7450  7510              SPA               /CHECK CORRECT MODE
7451  5370              JMP PTPERR        /SIGNAL "UNRECOVERABLE DEVICE
                                          /ERROR"
7452  0222              AND PT7700
7453  7040              CMA               /SET UP -(WORD COUNT)/2-1
7454  3375              DCA PTPWC
7455  1600              TAD I PTP         /SET UP START ADDRESS
7456  2200              ISZ PTP
7457  3312              DCA PTPCA
7460  1600              TAD I PTP
7461  7650              SNA CLA
7462  7430              SZL
```

```
7463  5271              JMP PTPCDF
7464  1373              TAD PTP336
7465  6046              6046
7466  6031              6031
7467  5266              JMP .-1
7470  6032              6032
7471  0000      PTPCDF, 0
      7471              PTPEOF=PTPCDF
7472  2375      PTPISZ, ISZ PTPWC
7473  5635              JMP I PSETUP     /LOOP FOR BUFFER SIZE(128 WORDS)
7474  1271      PTPRTN, TAD PTPEOF
7475  7141              CLL CIA
7476  7620              SNL CLA
7477  2200              ISZ PTP
7500  2200              ISZ PTP
7501  7402      PTPXIT, HLT      /EXIT CDF HERE
7502  7004              RAL
7503  5600              JMP I PTP
7504  0000      PTPPCH, 0
7505  6026              PLS      /NOTICE LACK OF OVERLAP
7506  6021              PSF
7507  5306              JMP .-1
7510  5704              JMP I PTPPCH
7511  7575      PTM203, -203
7512  0000      PTPCA,  0
7513  0000      PTR,    0        /ENTRY FOR PAPER TAPE READER
7514  7300              CLA CLL
7515  1313              TAD PTR
7516  3200              DCA PTP
7517  4235              JMS PSETUP       /SET UP ADDRESS, COUNT, FIELDS
7520  4345      PTRLP,  JMS PTRGCH       /READ FIRST CHAR OF 3
7521  3313              DCA PTR
7522  4345              JMS PTRGCH
7523  3304              DCA PTPPCH
7524  4345              JMS PTRGCH
7525  7006              RTL
7526  7006              RTL
7527  3345              DCA PTRGCH       /THIRD CHAR SPLIT
7530  1345              TAD PTRGCH
7531  0342              AND PT7400
7532  1313              TAD PTR
7532  1313              TAD PTR
7533  3712              DCA I PTPCA      /HIGH ORDER 4 BITS TO WORD 1
7534  1345              TAD PTRGCH
7535  7006              RTL
7536  7006              RTL
7537  0342              AND PT7400
7540  1304              TAD PTPPCH
7541  2312              ISZ PTPCA
7542  7400      PT7400, 7400
7543  3712              DCA I PTPCA      /LOW ORDER 4 BITS TO WORD 2
7544  5223              JMP PTPEND
7545  0000      PTRGCH, 0
7546  1271              TAD PTPEOF
7547  7650              SNA CLA
7550  5745              JMP I PTRGCH
```

```
7551  6014              RFC
7552  3271              DCA PTPEOF
7553  2271    PTTIME,   ISZ PTPEOF      /LOOP WILL OVERFLOW IN
                                        /APPROX. 66 MS
7554  5360              JMP PGCHLP
7555  3374              DCA PTPTMP      /IF IT DOES, WE HAVE RUN OUT OF
                                        /TAPE
7556  1210              TAD PTP232      /ZERO CHAR AT END OF TAPE AND
                                        /SEND
7557  5745              JMP I PTRGCH    /↑Z
7560  1374    PGCHLP,   TAD PTPTMP
7561  3712              DCA I PTPCA
7562  6011              RSF
7563  5353              JMP PTTIME      /READER NOT READY - CHECK TIMING
7564  6012              RRB             /READER READY - READ CHAR
7565  3374              DCA PTPTMP      /BUFFER READER BY ONE CHAR TO
                                        /ELIMINATE
7566  1712              TAD I PTPCA     /GARBAGE CHAR AT END OF TAPE
7567  5745              JMP I PTRGCH    /AND RETURN
7570  7332    PTPERR,   CLA CLL CML RTR /ROTATED LEFT TO FORM CODE
7571  5277              JMP PTPXIT-2    /FOR "PERMANENT I/O ERROR" ON
                                        /THE DEV.
7572  6201    PCDF,     CDF Ø
7573  0336    PTP336,   336             /"↑"
7574  0000    PTPTMP,   Ø
7575  0000    PTPWC,    Ø
                EJECT
```

6.8 RK8 DISK HANDLER

```
      6731              DCLA=6731
      6733              DLDR=6733
      6732              DLDC=6732
      6741              DRDS=6741
      6752              DRWC=6752
      6742              DCLS=6742
      6745              DSKC=6745
      6747              DSKE=6747
      6753              DLWC=6753
      6755              DLCA=6755
                        *7607
7607  0000    SHNDLR,   Ø
7610  7346              CLA CLL CMA RTL
7611  3333              DCA SYSCNT      /SET # OF TRIES ON ERROR
7612  6214              RDF
7613  1340              TAD SCIF
7614  3326              DCA SFIELD      /SAVE CALLING FIELD FOR RETURN
7615  1607              TAD I SHNDLR
7616  3334              DCA SFUN        /GET FUNCTION WORD
7617  2207              ISZ SHNDLR
7620  7240              CLA CMA
7621  1607              TAD I SHNDLR
7622  3330              DCA SLOC        /GET BUFFER LOC  -1
7623  2207              ISZ SHNDLR
```

```
7624  1607      STRY,   TAD I SHNDLR
7625  1347              TAD SOFSET        /SOFSET=Ø
7626  3331              DCA SREC          /STORE RECORD #
7627  1334              TAD SFUN
7630  7004              RAL
7631  0250              AND S7600
7632  3332              DCA SBLKCT        /SBLKCT=TOTAL WORD COUNT
                                          /TO BE READ
7633  7006              RTL
7634  1341              TAD SDLDR
7635  3255              DCA SINST         /SINST=6733 (READ) OR
                                          /6735 (WRITE)
7636  1330      SLOOP,  TAD SLOC
7637  6755              DLCA              /LOAD CURRENT ADDRESS REG.
7640  1332              TAD SBLKCT
7641  1250              TAD S7600
7642  7640              SZA CLA           /HALF - OR FULL - RECORD READ?
7643  1250              TAD S7600         /FULL
7644  1250              TAD S7600         /HALF
7645  6753              DLWC              /LOAD WORD COUNT REG.
7646  7240              CLA CMA
7647  6742              DCLS              /CLEAR ALL FLAGS
7650  7600      S7600,  7600              /CLEAR AC
7651  1334              TAD SFUN
7652  0310              AND S70
7653  6732              DLDC              /LOAD DATA FIELD OF BUFF.
                                          /INTO COMD. REG.
7654  1331              TAD SREC
7655  0000      SINST,  Ø                 /READ OR WRITE
7656  6745              DSKC              /SKIP ON DONE
7657  7410              SKP
7660  5304              JMP SNEXT
7661  6747              DSKE              /SKIP ON ERROR
7662  5256              JMP SINST+1
7663  2333      SERROR, ISZ SYSCNT        /BUMP TRY COUNTER
7664  5267              JMP .+3
7665  7330              CLA CLL CML RAR   /THREE TRIES, YOU'RE OUT
7666  5325              JMP SRTRN+1
7667  6741              DRDS
7670  0342              AND S40           /LOOK AT "SECTOR NOT FOUND" BIT
7671  2335              ISZ SZERO
7672  5271              JMP .-1           /DELAY TO LET UNIT QUIET DOWN
                                          /(KLUDGE)
7673  7650              SNA CLA
7674  5236              JMP SLOOP         /ERROR OTHER THAN "SECT. NOT
                                          /FOUND" - RETRY
7675  7240              CLA CMA
7676  6742              DCLS
7677  7200              CLA
7700  6731              DCLA              /RECALIBRATE DISK (SEEK TRACK Ø)
7701  6745              DSKC
7702  5301              JMP .-1
7703  5236              JMP SLOOP         /RETRY
7704  6747      SNEXT,  DSKE              /DONE - IS ERROR FLAG UP?
7705  7410              SKP
7706  5263              JMP SERROR        /YES - ERROR
7707  2331              ISZ SREC          /BUMP RECORD #
```

```
7710  0070      S70,    70
7711  1330              TAD SLOC
7712  1337              TAD S400
7713  3330              DCA SLOC          /BUMP CURRENT ADR
7714  1332              TAD SBLKCT
7715  7120              CLL CML
7716  1336              TAD S7400
7717  7470              SZL SNA           /IS WORD COUNT EXHAUSTED?
7720  5323              JMP .+3
7721  3332              DCA SBLKCT        /BUMP WORD COUNT
7722  5236              JMP SLOOP         /DO NEXT RECORD
7723  7200              CLA
7724  2207      SRTRN,  ISZ SHNDLR
7725  2207              ISZ SHNDLR
7726  7402      SFIELD, HLT               /RESTORE CALLING INST. FIELD
7727  5607              JMP I SHNDLR
7730  0000      SLOC,   0
7731  0000      SREC,   0
7732  0000      SBLKCT, 0
7733  0000      SYSCNT, 0
7734  0000      SFUN,   0
7735  0000      SZERO,  0
7736  7400      S7400,  7400
7737  0400      S400,   400
7740  6202      SCIF,   CIF 0
7741  6733      SDLDR,  DLDR
7742  0040      S40,    40
                0
```

APPENDIXES

A SUMMARY OF KEYBOARD COMMANDS

Command	Short Form	Necessary Arguments	Optional Arguments	Operation Performed
ASSIGN	AS	permanent device name	user device name	Associate user name with specified device. If no user name specified, disassociate user name from specified device.
DEASSIGN	DE	device name, filename	extension (if not SV)	Disassociate all user names from all devices.
GET	GE	device name. filename	extension (if not SV)	Load specified core image file into core and move its core control block into the system scratch area.
SAVE	SA	device name, filename	extension (if not SV)	Dump program in core on specified device in a file with the specified name.
			addresses of areas of core to be saved (join by hyphen, separate by comma)	If the information given by optional arguments is not specified, it is taken from the core control block.
			starting address (preceded by ;)	
			Job Status Word (preceded by =)	
START	ST		start address	Start program in core. If a start address is not specified, use the on in the core control block.
ODT	OD			Load and start ODT.

RUN	RU	device name, filename	extension (if not SV)	Load the specified core image file and its core control block and start it.
R	R↑ (space)	filename	extension (if not SV)	Load and start the specified core image file.

B SPECIAL CHARACTER COMMANDS

(KM stands for the Keyboard Monitor; CD stands for the Command Decoder.)

Character	Where Used	Effect
CTRL/C	anywhere	returns to KM
CTRL/U	CD,KM	deletes current line of input
RUBOUT	CD, KM	deletes a character from the end of the input line
Carriage return	CD, KM	ends input line
ALT-MODE (or PREFIX or ESCape)	CD, KM	ends input line
Line Feed	CD, KM	prints a clean copy of current command line (free of RUBOUT corrections)
CTRL/O	TTY handler ODT	stops output
CTRL/Z	TTY handler	end of file

C ERROR MESSAGES

C.1 From the Keyboard Monitor

Message	Meaning
device NOT AVAILABLE	The device name specified (to a RUN, GET, SAVE or ASSIGN command) is not in any system tables.
xxxx ?	"xxxx" is not a legal command; e.g., if the user typed "HELLO" the system would respond "HELLO?"
filename NOT FOUND	The file name specified (to a RUN, GET or R command) does not exist on the device specified.
TOO FEW ARGS	You left out an argument to a command; e.g., "RUN DSK" would produce this message.
NO!!	You attempted to START a program which cannot be started for some reason. For example, the program may modify itself and hence must be reloaded before starting it again.
SAVE ERROR	An I/O error has occurred while SAVEing a program. The program is still intact in core after this message is given.
ILLEGAL ARG	Bad syntax was detected in the optional arguments of a SAVE command.
BAD ARGS	Two areas of core specified in the optional arguments of a SAVE command overlay each other.
USER ERROR Ø AT xxxxx	An I/O error was detected while loading a program (RUN, GET or R commands) xxxxx is a meaningless vestigial location.
MONITOR ERROR 2 AT xxxx	**An attempt has been made to output to a WRITE LOCKED device, usually DECtape.**

C.2 From the Command Decoder

Message	Meaning
ILLEGAL SYNTAX	This command line does not make sense.
TOO MANY FILES	More than three output files or nine input files were specified.
device DOES NOT EXIST	The device specified could not be found in the system tables.
filename NOT FOUND	The specified filename does not exist on the specified device.

C.3 From the Linking Loader

The Linking Loader gives error messages in the form "ERROR nnnn". The meaning of the different values of nnnn are:

Error Code	Explanation
0001	Symbol table overflow (more than 64 subprogram names)
0002	Program will not fit
0003	Program with largest common storage was not loaded first
0004	Checksum error in input tape
0005	Illegal relocation code
0006	An output error has occurred while outputting a storage map
0007	An input error has occurred while reading a binary file
0010	No starting address has been specified and there is no entry point named "MAIN"
0010	An internal error has occurred. Please send the typeout and history to Digital Equipment Corporation.

C.4 From the PS/8 Version of FORTRAN

The following error messages have been added to the PS/8 version of FORTRAN (all error messages described in Chapter 15 of Programming Languages are also valid):

Message	Explanation
I/O ERROR	A device handler has signalled a fatal I/O error.
NO ROOM FOR OUTPUT	The file FORTRN.TM cannot fit on device SYS:.
SABR.SV NOT FOUND	Self-explanatory. The compiler will not run without SABR.
NO END STATEMENT	The input to the compiler has been exhausted.
COMPILER MALFUNCTION	The meaning of this message has been extended to cover various "impossible" monitor errors.