DEC-10-MRDC-D

decsystem10 OPERATING SYSTEM COMMANDS

This manual reflects the software associated with the 5.05 monitor. For individual system program version numbers, refer to Page iii.

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

1st Printing May 1971
2nd Printing (Rev) December 1971
3rd Printing (Rev) June 1972

SOFTWARE VERSION NUMBERS

The following versions of the software are discussed in this manual.

ALCFIL	Version 7	INITIA	Version 3
BACKUP	Preliminary Information	KJOB	Version 47
BATCON	Version 6	LINED	Version 13
CDRSTK	Version 11	LOGIN	Version 53
COMPIL	Version 20	Monitor	5.05
COPY	Version 6	OMOUNT	Version 22
CREF	Version 46	OPSER	Version 4
DAEMON	Version 6	PIP	Version 32
DIRECT	Version 2	PLEASE	Version 11
DUMP	Version 4	QUEUE	Version 3
FILCOM	Version 16	QUOLST	Version 4
FILEX	Version 15	REATTA	Version 3
FUDGE2	Version 14	RESTORE	Preliminary Information
GLOB	Version 5	SETSRC	Version 11
GRIPE	Version 3	SYSTAT	Version 467
HELP	Version 3	TECO	Version 23
		UMOUNT	Version 20

CONTENTS

CONTENTS (Cont)

CONTENTS (Cont)

CONTENTS (Cont)

CONTENTS (Cont)

CONTENTS (Cont)

ILLUSTRATIONS

TABLES

FOREWORD

DECsystem-10 Operating System Commands is a complete reference document describing the commands available in the DECsystem-10 operating system. The information presented in this manual reflects the 5.05 release of the monitor and other related programs. Commands to both the monitor command language interpreter and the programs in the Batch system are grouped in alphabetical order for easy reference to the command repertoire.

DECsystem-10 Operating System Commands does not include reference material on assembly language programming. This information can be found in DECsystem-10 Monitor Calls (DEC-10-MRRC-D), which is intended for the experienced assembly language programmer. Included in DECsystem-10 Monitor Calls are discussions of the monitor programmed operators and the various I/O devices connected to the system. The two manuals, DECsystem-10 Operating System Commands and DECsystem-10 Monitor Calls, supersede the Timesharing Monitors Programmer's Reference Manual (DEC-T9-MTZD-D) and all of its updates.

A third manual, Introduction to DECsystem-10 Software (DEC-10-MZDA-D), is a general overview of the DECsystem-10. It is written for the person, not necessarily a programmer, who knows computers and computing concepts and who desires to know the relationship between the various components of the DECsystem-10. This manual is not intended to be a programmer's reference manual, and therefore, it is recommended that it be read at least once before reading the above-mentioned reference documents.

SYNOPSIS OF DECsystem-10 OPERATING SYSTEM COMMANDS

Chapter 1 presents all of the commands available to the user and introduces the various components of the operating system that interface with the user. Chapter 2 is a detailed description of the commands processed by the monitor command language interpreter. Presented in Chapter 3 are the commands to the Batch system and a discussion of the programs in this system. The DECsystem-10 system error messages and error codes are listed in Chapter 4 along with descriptive information on how to correct the errors. The appendices contain supplementary reference material and tables.

CONVENTIONS USED IN DECsystem-10 OPERATING SYSTEM COMMANDS

The following conventions have been used throughout this manual:

dev:	Any logical or physical device name. The colon must be included when a device is used as part of a file specification.
list	A single file specification or a string of file specifications. A file specification consists of a filename (with or without a filename extension), a device name if the file is not on disk, a project-programmer number, if the file is not in the user's disk area, and a protection code.
arg	A pair of file specifications or a string of pairs of file specifications.
jobn	A job number assigned by the monitor.
file.ext	Any legal filename and filename extension.
core	Decimal number of 1K blocks of core.
adr	An octal address.
C(adr)	The contents of an octal address.
[proj,prog]	Project-programmer numbers; the square brackets must be included in the command string.
fs	Any legal file structure name or abbreviation.
Ⓢ	The symbol used to indicate an altmode.
↑x	A control character obtained by depressing the CTRL key and then the character key x.
←	A back arrow used in command strings to separate the input and output file specifications.
*	The system program response to a command string.
.	The monitor response to a command string.
↲	The symbol used to indicate that the user should depress the RETURN key. This key must be used to terminate every command to the Monitor Command Language Interpreter.
___	Underscoring used to indicate computer typeout.
n	A decimal number.
=	An equal sign used in command strings to separate the input and output file specifications.

CHAPTER 1 INTRODUCTION

The DECsystem-10 Operating System is the interface between the user and the actual machine. The operating system, or monitor, has many functions, some of which are:

1. scheduling multiple and simultaneous use of the system,
2. protecting users of the system from one another,
3. allowing access to system resources including peripheral devices,
4. providing a comprehensive disk file system,
5. directing data flow between peripheral devices and the user's program,
6. controlling non-interactive jobs, and
7. overlapping input-output operations with computations for high system efficiency.

The user communicates with the operating system by means of the monitor command language. With the command language he may access all available resources of the computing system and obtain all the services provided by the operating system.

1.1 JOBS

The DECsystem-10 computing system is a multiprogramming system; that is, control is transferred rapidly among a number of jobs in such a way that all jobs appear to be running simultaneously. The term job refers to the entire sequence of steps, from beginning to end, that the user initiates from his interactive terminal or card deck or that the operator initiates from his operator's console. When a user initiates a job from his interactive terminal, the beginning of the job is designated by the LOGIN command and the end by the KJOB command. If a user initiates a job with a card deck, the beginning of the job is the $JOB card and the end is the end-of-file card. Operator jobs usually begin when the system is initialized and end when the system goes down.

Jobs, which may be timesharing, batch, or real-time in nature, may be initiated at the central computer site or at remote locations connected by the telephone system. Once a user initiates a job, it is possible for him to initiate another job without killing the first one. For example, a user can initiate a timesharing job and by using the SUBMIT monitor command submit a second job for batch processing (refer to Chapter 2). He may then wait for the results from this batch job, or have the results automatically output while he continues his timesharing job.

In configuring and loading the DECsystem-10, the system administrator sets the maximum number of jobs that his system can simultaneously handle. This number may be up to 127 jobs if the system has enough memory, disk storage, processor capacity, and terminals to handle this load.

1.2 MONITOR MODE AND USER MODE

From the timesharing user's point of view, his terminal is in either monitor mode or user mode. In monitor mode, each line the user types in is sent to the monitor command language interpreter. The execution of certain commands (as noted in the following examples) places the terminal in user mode. When the terminal is in user mode, it becomes simply an I/O device for that user. In addition, user programs use the terminal for two purposes. The user program will either accept user command strings from the terminal (user mode) or use the terminal as a direct I/O device (data mode).

Example (terminal dialogue):

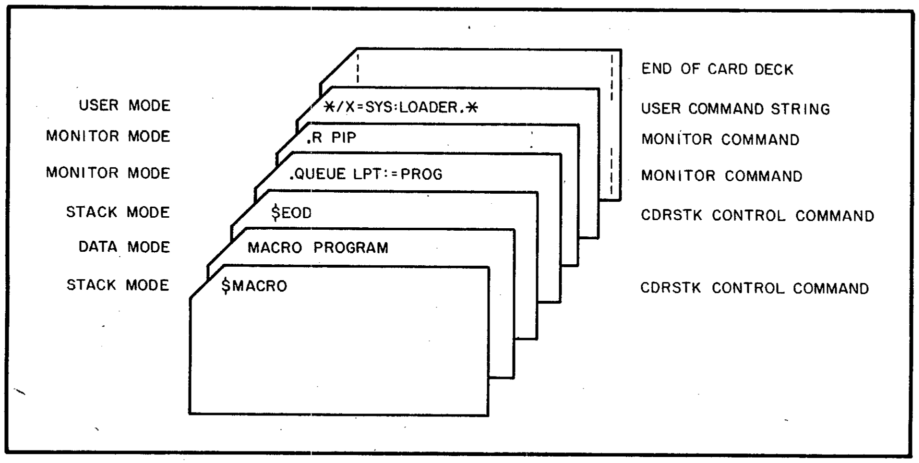
```
monitor mode    .R PIP↵                  monitor command
user mode       *DSK:PROG1.MAC←TTY:↵     user command string
data mode       THIS IS FILE 1 ↑Z        user program using terminal
                                         as input device
                *↑C
monitor mode    .R MACRO↵                monitor command

user mode       *,TTY:←DSK:PROG1↵        user command string
data mode                 .              user program using terminal
                          .              as an output device
                 assembly listing
                          .
                          .
```

The special character ↑C (produced by typing C with the CONTROL key depressed) is used by a timesharing user to stop a user program and return the terminal to monitor mode. If the user program is waiting for input from the terminal, the user needs to type only one ↑C to return the terminal to monitor mode; otherwise, he must type two ↑C's. Because of this procedure, the user knows that his program is not waiting for input if there is no response from the monitor after one ↑C. Certain commands cause the user program to start running or to continue (as noted in the following chapter) but leave the terminal in monitor mode.

When the system is started, each terminal is in monitor mode ready for users to log in. However, if the system becomes fully loaded (i.e., the maximum number of jobs that the system is set to handle has been initiated), then any unused terminals from which access is requested will receive the message JOB CAPACITY EXCEEDED.

The card-oriented Batch user can think of his cards as being in stack mode, monitor mode, or user mode. When the card is in stack mode, it contains a control command beginning with a $ (refer to Chapter 3) and is sent to the Stacker, CDRSTK. CDRSTK interprets these commands and performs various actions to create a control file for the Batch Controller. When the card is in monitor mode, it contains a monitor command preceded by a period and is copied by CDRSTK into the control file. When the card is in user mode, it contains a user-level program command preceded by an asterisk or an equal sign and is also copied by CDRSTK into the control file. As each line in the control file is executed, the Batch Controller passes the monitor-level line to the monitor command language interpreter and the user-level line to the user program.

Example (sample card deck):



10-0897

1.3 COMMAND INTERPRETERS

1.3.1 Monitor Command Language Interpreter

When the terminal is in monitor mode, the user communicates with the monitor command language interpreter. By means of commands to this interpreter, the user may initialize jobs, allocate facilities, prepare source files, manipulate files, prepare, control, and examine object programs, control job sequences and multiple jobs, terminate jobs, send messages, and obtain job and system information. The commands described in Chapter 2 are processed by this interpreter.

Most commands are processed without delay. However, a command may be momentarily delayed if a job is swapped out to the disk and the command requires that the job be resident in core; the command is executed when the job is swapped into core. The completion of each command is signaled by the output of a carriage return, line feed sequence. If the terminal is left in monitor mode, a period follows the carriage return, line feed. If the terminal is left in user mode, any response other than a carriage return, line feed comes from the user's program. For example, most standard system programs immediately send an asterisk(*) to the user's terminal to indicate their readiness to accept user command strings.

The type-ahead technique may be employed by the experienced timesharing user at a terminal. This means that the user does not have to wait for the completion of one command before he can begin another. For example, if two operations are desired from the monitor, the request for the second operation can be typed before receiving the period after completion of the first.

The command interpreter makes several checks before processing commands from users. On disk systems, if a user who has not logged in types a command that requires him to be logged in, the system responds with

```
?LOGIN PLEASE
```

and the user's command is not executed. The commands discussed in Chapter 2 all require login except where explicitly stated otherwise. When a command is recognized that requires the job to have core and the job has no core allocated, the command interpreter responds with

```
?NO CORE ASSIGNED
```

and the user's command is not executed.

1.3.1.1 Special Characters - There are several special characters recognized by the monitor command language interpreter that cause specific functions to be performed. As noted previously, control-C (↑C) interrupts the program that is currently running and returns the terminal to monitor mode. This character causes the input line back to the last break character (e.g., carriage return, line feed) to be deleted (equivalent to the action of a ↑U). Two control-C's are necessary if the user program is not requesting input from the terminal (i.e., the program is in the middle of execution).

The RUBOUT key on the terminal generates a character that causes the last character typed to be deleted. This permits correction of typing errors. Depressing the RUBOUT key n times causes the last n characters typed to be deleted. The deleted characters are echoed on the terminal enclosed in backslashes (\). Characters beyond the last break character or characters already processed by the user program are not deleted.

Control-U (↑U) causes the deletion of the current line, back to the last break character. The system responds with a carriage return, line feed so that the line may be typed again. Once a break character has been typed, line-editing features (↑U and RUBOUT) can no longer be used on that line, except when running TECO.

Control-O (↑O) suppresses output to the terminal. The system responds with a carriage return, line feed sequence. A subsequent control-O re-enables output to the terminal. At remote stations, the effect of the ↑O may be somewhat delayed.

1.3.2 Batch Command Interpreter

The monitor command language interpreter is used for all monitor commands submitted via the Batch system. In addition, the Batch user issues commands that are only used by the Batch programs, Stacker (CDRSTK) and Batch Controller (BATCON). Control commands, discussed in Chapter 3, are processed by the Stacker and, by means of these commands, the user can create a control file, a log file, and data files; can enter jobs into the Batch input queue; and can insert monitor commands into the control file. An additional interpretation is done by the Batch Controller. When the job is executed, the Batch Controller processes the control file to pass monitor commands to the monitor command language interpreter and user-level commands to the appropriate programs.

1.4 COMMAND FORMATS

Each command is a line of ASCII characters in upper and/or lower case. Spaces and TABs preceding the command name are ignored. Comments may be typed on the same line as the command by preceding the comment with a semicolon. The monitor and batch command language interpreters do not interpret or execute a line of comments. Every command to the monitor command interpreter should be terminated by pressing the RETURN key on the console. In examples in this manual, the symbol ↵ is used to indicate that the user should depress the RETURN key. If the command is in error, the command up to the error is typed out by the monitor preceded and followed by a ?, and the terminal remains in monitor mode.

1.4.1 Command Names

Commands to the monitor command interpreter are alphabetic strings of one to six characters; characters after the sixth are ignored. Only enough characters to uniquely identify the command need be typed. It is recommended that a Batch job use the full command name since the abbreviations may change with the addition of new commands.

Installations choosing to implement additional commands should take care to preserve the uniqueness of the first three letters of existing commands.

Control commands to the Stacker in the multiprogramming batch system must have a dollar sign ($) in the first column of the card or the line and an alphabetic character in the second column. Only the first part of the command name need be specified; as long as the specified command name is unique, it is accepted. The first three characters of the command name are generally sufficient to ensure uniqueness.

1.4.2 Command Arguments

Arguments follow the command name and are separated from it by a space or TAB. If the monitor command interpreter recognizes a command name, but a necessary argument is missing, the monitor responds with

```
?TOO FEW ARGUMENTS
```

Extra arguments are ignored.

1.4.2.1 Project-Programmer Numbers and Passwords - Access to the DECsystem-10 is limited to authorized users. The system administrator provides each authorized user with a project number, a programmer number, and a password. The project numbers range from 1 to 377777 octal (numbers 1 to 10 are reserved for DEC) and the programmer numbers range from 1 to 777777 octal (numbers 1 to 7 are reserved for DEC and numbers 400000 to 777777 are reserved for special purposes)[1]. These numbers identify the user and his file storage area on a file structure. In a command string, the project and programmer numbers are separated with a comma and must be enclosed in square brackets, e.g., [10,7].

The password is from one to six SIXBIT characters and is only used when logging on the computing system. To maintain password security, the monitor does not echo the password. On terminals with local copy (refer to DECsystem-10 Monitor Calls), a mask is typed to make the password unreadable.

1.4.2.2 Device Names - Associated with each system device controlled by the computing system is a physical device name. This name consists of three letters, zero to three numerals specifying the unit number, and a colon. Table 1-1 lists the generic physical device names associated with the various system devices.

[1]When the programmer number is from 1 to 7, all project numbers are reserved for DEC.

Table 1-1
System Devices

Device	Generic Physical Device Name
All Disks	ALL:
Card Punch	CDP:
Card Reader	CDR:
Console TTY	CTY:
DECtape	DTx:†
Disk	DSK:
Packs	DPx:†
Fixed-Head	FHx:†
Display	DIS:
Experimental System Library	NEW:
Help Library	HLP:
Line Printer	LPT:
Magnetic Tape	MTA:
Operator Terminal	OPR:
Paper-tape Punch	PTP:
Paper-tape Reader	PTR:
Plotter	PLT:
Pseudo-TTY	PTY:
System Library	SYS:
Terminal	TTY:

†X represents A, B, . . ., indicating the first controller, second controller, etc.

The user may also assign a logical device name to a physical device. The logical name is from one to six alphanumeric characters of the user's choice, followed by a colon, and is used synonymously with a physical device name in all references to the device. Logical device names allow the user, when writing his program, to use arbitrarily selected device names, which he assigns to the most convenient physical devices at run time. However, care should be exercised when assigning logical device names because these names have priority over physical device names. For example, if a DECtape is assigned the logical name DSK, then all of the user's programs attempting to use the disk via the device name DSK will use the DECtape instead.

Except for disk devices, only one logical device name can be associated at any one time with a physical device. The same logical name can be used for a second physical device by disassociating it from the first device and associating it with the second device via the ASSIGN command. Logical device names are disassociated from all devices with the DEASSIGN command (refer to Chapter 2). Subsequent ASSIGN commands (refer to Chapter 2) to all devices except disk devices replace the old logical name with the new one.

The following is an example of the use of physical and logical device names. Underscoring is used to indicate computer typeout.

`.ASSIGN DTA: ABC:↵`	User requests a DECtape drive with the logical name ABC.
`DEVICE DTA6 ASSIGNED`	Monitor has given the user drive DTA6. The user mounts a DECtape on drive DTA6.
`.ASSIGN PTP: ABC:↵`	User requests the paper-tape punch with the logical name ABC.
`% LOGICAL NAME WAS IN USE, PTP ASSIGNED`	Paper-tape punch is reserved, and ABC now refers to the PTP.
`.R PIP↵`	User requests the system program PIP (Peripheral Interchange Program).
`*ABC:←DTA6:FILEA↵`	User issues a command string to PIP asking that file FILEA be transferred from device DTA6 to logical device ABC (physical device PTP: which is assigned to the user).
`*↑C`	User returns to monitor mode.
`.ASSIGN DTA: DEF:↵`	User requests another DECtape drive with logical name DEF.
`.ASSIGNED TO JOB N1,N2,...`	All drives are in use by the specified jobs. No DECtape drive is assigned, and no logical assignment is made.
`.ASSIGN DTA6: DEF:↵`	User requests drive DTA6 (which he already has) with logical name DEF. The copy of the directory currently in core is cleared.
`DEVICE DTA6 ASSIGNED`	User mounts a new DECtape on the previously assigned drive. The new DECtape directory is read into core when next accessed.
`.DEASSIGN PTP:↵`	User deassigns PTP, thereby clearing the logical name ABC.
`.R PIP↵`	User requests PIP.
`*ABC:←DEF:FILEB↵`	User requests that file FILEB be transferred from device DEF to device ABC.
`?DEVICE ABC DOES NOT EXIST`	The logical device name ABC is no longer assigned.

(continued on next page)

```
*↑C                          User returns to monitor mode.

.ASSIGN DTA6: XYZ:)          User requests drive DTA6 again with logical name XYZ.
                             The logical name DEF is no longer associated with DTA6.
                             The old directory is cleared from core.

DEVICE DTA6 ASSIGNED         User mounts a new DECtape. The new directory is read
                             into core when next accessed.
```

1.4.2.3 File Structure Names - Disk devices are grouped according to file structures, which are logical arrangements of 128-word blocks on one or more disk units of the same type. Examples of types of disk units are: an RP02 disk pack or an RM10B drum. Although a file structure can exist on exactly one disk unit, it can be distributed over several disk units of the same type and designated by a single name. However, two file structures cannot exist on the same unit. Each file structure has a SIXBIT name specified by the operator at structure definition time. This name can consist of five or less alphanumeric characters and must not duplicate a physical device name, a unit name, or an existing file structure name. The recommended names for public file structures are DSKA, DSKB, ..., DSKN in order of decreasing speed.

1.4.2.4 File Specifications - All information (programs and data) in the system is stored as named files. Each named file has associated with it a file specification which consists of

1. the physical device name or file structure name,
2. the filename,
3. the filename extension,
4. the ordered list of directory names, and
5. the access protection code.

The first four items of the file specification are necessary to uniquely identify a disk file. File specifications are ignored when given for devices other than DECtape or disk since these two devices are the only directory-oriented devices. In addition, items 4 and 5 do not apply to DECtapes.

The physical device name used for DECtape or the file structure name used for disk may be any legal device name discussed in the foregoing sections. A colon should always follow the device name; e.g., DTA3:. The filename is from one to six SIXBIT characters; all characters after the sixth are ignored. The filename extension is a period following by zero to three characters and is used to indicate the type of information in the file. (Refer to Appendix A for a list of standard filename extensions.) It is recommended that only the standard extensions be used even though other extensions are valid. Most programs only recognize filenames and extensions consisting of letters and digits. The ordered list of directory names identifies the disk area in which the file is stored. This list can be a user file directory (UFD) represented by the project-programmer number of the owner of the files in the directory or can be a user file directory followed by one or more sub-file directories (SFDs). (Refer to the DECsystem-10 Monitor Calls for a description of SFDs.) The directory name must be enclosed in square brackets. The access protection of the file is a three-digit code designating which users can read or write the file and must be enclosed in angle brackets. The protection code is specified only for output files. For a given file, the users are divided into three groups: the owner of the file, the users with the same project number as the owner, and the rest of the users. The standard protection code is 057 which allows users in the owner's project to read and execute the file and prevents access by all other users. (For a complete description of access protection, refer to DECsystem-10 Monitor Calls.) The standard protection code can be redefined by the various installations.

In command strings, the filename, the device name if the file is not on disk, and the directory name if the file is not in the user's disk area, are required. The filename extension, the device name if the file is on the disk, the directory name if the file is in the user's disk area, and the protection code are optional. The following are examples of file specifications:

TEXT.MAC	filename and extension
DTA3:FILEA	device and filename
DSK:PROG2.CBL[10,16]	device, filename, extension, and directory name
DSKA:MAIN.F4[27,235] <057>	complete file specification

Many command strings allow the wildcard construction to be used. This means that the filename, the extension, or the directory name may be replaced totally with an asterisk or partially with a question mark to designate certain filenames, extensions, or directories. The asterisk is used as a wild field to designate the entire filename, extension, or directory name. For example,

filename.*	All files with this filename and any extension.
*.ext	All files with this extension and any filename.
.	All files.
. [project,*]	All files in directories with this project number and any programmer number.

The question mark is used as a wild character to designate part of the filename, extension, or directory name. A question mark is used for each character that is to be matched; i.e., PR?? matches on four characters or less. For example,

filename .M??	All files with this filename and any extension beginning with M.
TES??.ext	All files with this extension and any filename up to 5 characters beginning with TES.
??.???	All files with filenames of two characters or less.

(continued on next page)

[25,5??]	All files in directories with the project number 25 and the programmer numbers 500-577.

The asterisk and the question mark can be specified together in the same construction.

??.*	All files with filenames of two characters or less.

In addition, the directory name can be specified with the project number, the programmer number, or both numbers missing. The following are examples of the various ways of representing a particular directory.

[15,23]	The UFD [15,23].
[,30]	The UFD that has the user's project number and the specified programmer number (30).
[36,]	The UFD that has the specified project number (36) and the user's programmer number.
[,]	The user's UFD.
[-]	The user's default directory which may be different from his UFD (refer to the DECsystem-10 Monitor Calls and the SETSRC program).

The number sign can be used to represent a filename or extension that contains characters that cannot be typed because they have special meanings in the system. For example, if a file with the name *.MAC were typed in a command string, the user would be referencing all files with the extension .MAC since the * designates all files with the specified extension MAC. To allow a filename or extension to be typed that is composed of special characters, the user employs the number sign followed by the octal representation of the SIXBIT filename or extension. For example, #120000000000 represents the file named *. If letters or digits are part of the filename or the extension containing the special characters, the octal representation of the letters or digits must also appear following the number sign. In other words, the number sign must be placed at the beginning of the filename and all characters following must be represented in octal. Furthermore, this construction can be used to read the contents of a UFD. For example, #000010000073.UFD [1,1] represents the file named 10,73.UFD in the directory [1,1].

The programs that recognize the number sign are DUMP, DIRECT, PIP, and QUEUE.

1.5 COMPIL-CLASS COMMANDS

Certain monitor commands simplify communication between the user and the system programs of the DECsystem-10 by allowing the user to type a short, concise monitor command string that causes a series of operations to be performed. These commands are known as COMPIL-class commands and are described in detail in Chapter 2. These commands cause the monitor to run the COMPIL program, which deciphers the command and constructs new command strings for the system program (e.g., TECO, PIP, LINED, FORTRAN) that actually processes the command. Each time CREATE, MAKE, EDIT, or TECO is executed, the command with its arguments is written as a temporary file in core or on the disk. Therefore, the file specification last edited may be recalled for the next edit without specifying the arguments again. (This is an exception to the requirement that the filename must always be specified.) For example, if the command

```
        .CREATE PROGX.MAC
```

is executed, then the user may later type the command

```
        .EDIT
```

instead of

```
        .EDIT PROGX.MAC
```

assuming no other EDIT-class command that changed the filename was used in the interim.

The COMPILE, LOAD, EXECUTE, and DEBUG commands with their arguments are also written in a temporary file so that the file specification given last may be recalled without specifying the arguments again.

The temporary files containing these file specifications have filenames of the following form:

```
        nnnxxx.TMP
```

where nnn is the user's job number in decimal, with leading zeros to make three digits, and xxx specifies the use of the file. Refer to Appendix C for a list of the temporary files.

1.5.1 Indirect Commands (@ Construction)

When there are many program names and switches, they can be put into a file and do not have to be typed in for each compilation. This is accomplished by the use of the @ file construction, which may be combined with any COMPIL-class command.

The @ file may appear at any point after the first word in the command. In this construction, the word <u>file</u> must be a filename, which may have an extension and a project-programmer number. If the extension is omitted, a search is made for the command file with a null extension and then for a command file with the extension .CMD. The information in the specified command file is then put into the command string to replace the characters @ file.

For example, if the file FLIST contains the string

```
        FILEB,FILEC/LIST,FILED
```

then the command

```
        .COMPILE FILEA,FILEB,FILEC/LIST,FILED,FILEZ
```

could be replaced by

```
        .COMPILE FILEA,@FLIST,FILEZ
```

Command files may contain the @ file construction to a depth of nine levels.[1] If this process of indirection results in files pointing in a loop, the maximum depth is rapidly exceeded and an error message is produced.

The following rules apply in handling format characters in a command file.

a. Spaces are used to delimit words but are otherwise ignored. Similarly, the characters TAB, VTAB, and FORM are treated like spaces.

b. To allow long command strings, command terminators (CARRIAGE RETURN, LINE FEED, ALTMODE) are ignored if the first nonblank character after a sequence of command terminators is a comma. Otherwise, they are treated either as commas by the COMPILE, LOAD, EXECUTE, and DEBUG commands or as command terminators by all other COMPIL-class commands.

c. Blank lines are completely ignored because strings of returns and line feeds are considered together.

d. Comments may be included in command files by preceding the comment with a semicolon. All text from the semicolon to the line feed is ignored.

e. If command files are sequenced, the sequence numbers are ignored.

1.5.2 The + Construction[2]

A single relocatable binary file may be produced from a collection of input source files by the "+" construction. For example: a user may wish to compile the parameter file, PAR.MAC, the switch file, SWIT.MAC, and the file that is the body of the program, MAIN.MAC. This is specified by the following command:

```
        .COMPILE PAR+SWIT+MAIN
```

---

[1]However, if BLISS, SNOBOL, and MACX11 (the PDP-11 assembler for the PDP-10) are added as processors, one less level of indirecting for each processor is obtained. These processors will be recognized only when the appropriate assembly switches are set. These assembly switch settings are not supported.

[2]Use in COMPILE, LOAD, EXECUTE, and DEBUG commands only.

The name of the last input file in the string is given to any output (.REL, .CRF, and/or .LST) files (e.g., MAIN in the preceding example). The source files in the "+" construction may each contain device and extension information and project-programmer numbers.

1.5.3 The = Construction[1]

Usually the filename of the relocatable binary file is the same as that of the source file, with the extension specifying the difference. This can be changed by the "=" construction, which allows a filename other than the source filename to be given to the associated output files. For example: if a binary file named BINARY.REL is desired from a source program named SOURCE.MAC, the following command is used.

```
.COMPILE BINARY=SOURCE
```

This technique may be used to specify an output name to a file produced by the use of "+" construction. To give the name WHOLE.REL to the binary file produced by PART1.MAC and PART2.MAC, the following is typed.

```
.COMPILE WHOLE=PART1+PART2
```

Although the most common use of the "=" construction is to change the filename of the output files, this technique may be used to change any of the other default conditions. The default condition for processor output is DSK:source.REL [self]. For example: if the output is desired on DTA3 with the filename FILEX, the following command may be used:

```
EXECUTE DTA3:FILEX=FILE1.F4
```

1.5.4 The <> Construction[1]

The <> construction causes the programs within the angle brackets to be assembled with the same parameter file. If + is used, it must appear before the <> construction. For example to assemble the files LPTSER.MAC, PTPSER.MAC, and PTRSER.MAC, each with the parameter file PAR.MAC, the user could type

```
.COMPILE PAR+LPTSER, PAR+PTPSER, PAR+PTRSER
```

With the angle brackets, however, the command becomes

```
.COMPILE PAR+<LPTSER,PTPSER,PTRSER >
```

However, the following command is invalid:

```
.COMPILE <LPTSER,PTPSER,PTRSER >+PAR
```

[1]Used in COMPILE, LOAD, EXECUTE, and DEBUG commands only.

1.5.5 Compile Switches

The COMPILE, LOAD, EXECUTE, and DEBUG commands can be modified by including switches in the command string. These switches can be used to indicate the processor to be used, to force a compilation, to generate listings, to create libraries, to search user libraries, and to obtain loader maps. Each switch is preceded by a slash and terminated with a non-alphanumeric character, usually a space or a comma. The switch used can be abbreviated if the abbreviation uniquely identifies the switch.

The switches used with these four commands are either temporary or permanent. A temporary switch applies only to the file immediately preceding it. An intervening space or comma cannot separate the filename and the switch. For example,

```
.COMPILE PROG,TEST/MACRO,SUBLET
```

The /MACRO switch applies only to the file named TEST.

A permanent switch applies to all files following it until modified by a subsequent switch. It is separated from the file by spaces, commas, or a combination of both. For example,

```
.COMPILE PROG /MACRO TEST,SUBLET
.COMPILE PROG,/MACRO,TEST,SUBLET
.COMPILE PROG,/MACRO TEST,SUBLET
.COMPILE PROG /MACRO,TEST,SUBLET
```

In all four examples, the /MACRO switch applies to the files named TEST and SUBLET.

The switches that can be used with the COMPILE, LOAD, EXECUTE, and DEBUG commands are described in the individual command explanations in Chapter 2.

1.5.6 Standard Processor

Files with recognizable processor extensions (e.g., .MAC, .CBL, .F4, .ALG) are always translated by the processor implied by the extension.[1] For example, a file named DATPRO.CBL will be processed by the COBOL compiler. Files without a recognizable processor extension are compiled or assembled according to the standard processor, which is normally FORTRAN at the beginning of the command string. The user can control the setting of the standard processor by including switches in the COMPILE, LOAD, EXECUTE, or DEBUG command string. Refer to the appropriate command descriptions in Chapter 2 for the switches used to change the standard processor.

[1] By setting the appropriate assembly switches, SNOBOL, BLISS, and MACX11 (the PDP-11 assembler for the PDP-10) will be recognized as processors. However, these assembly switch settings are not supported.

In the following examples, the installation has chosen FORTRAN as the standard processor. The command

```
.COMPILE NOEXT
```

causes the file named NOEXT (with a null extension) to be compiled by FORTRAN. The command

```
.COMPILE FILEZ.MIN
```

also compiles the file with FORTRAN since .MIN is not recognized as a processor extension. The command

```
.COMPILE APART,DATA/COBOL, TEST
```

causes the files APART and TEST to be compiled by FORTRAN and the file DATA by COBOL.

The switches used to change the standard processor can be temporary or permanent switches (refer to Paragraph 1.5.5). For example,

```
.COMPILE APART, /COBOL DATA,TEST
```

causes APART to be compiled by FORTRAN, and DATA and TEST to be compiled by COBOL.

Note that if source files are specified with the appropriate extensions, the subject of the standard processor can be disregarded, since files with processor extensions are always translated by the processor implied.

1.5.7 Processor Switches

Occasionally it is necessary to pass switches to the assembler or compiler being used in a COMPILE, LOAD, EXECUTE, or DEBUG command. For each translation (assembly or compilation), the COMPIL program sends a command string to the translator containing three parts: the source files, a binary output file, and a listing file. To incluae switches with these files, the user must:

a. If the + construction is used, group the switches according to each related source file.

b. Group the switches according to the three types of files (source, binary, and listing) for each file.

c. For each source file, separate the groups of switches by commas.

d. Enclose all the switches for each source file within one set of parentheses.

(SSSS)	Only source switches are present.
(SSSS,BBBB)	Source and binary switches are present.
(SSSS,BBBB,LLLL)	Source, binary, and listing switches are present.

e. Place each parenthesized string immediately after the source file to which it refers.

The processor switches are listed in Table 1-2, along with their meanings and the types of files to which they apply.

Table 1-2
Processor Switches

Processor	Source	Binary	Listing	Meaning
ALGOL		D		Set dynamic storage region for own arrays (known as the heap).
	E			The source file has line numbers in columns 73-80.
	L			List the source program.
			N	Suppress the error print out on the terminal.
	Q			Delimit the words in quotes.
	S			Suppress the listing of the source program.
COBOL	A	A	A	Allow the listing of code generated.
			C	Produce a cross-referenced listing of all user-defined items in the source program.
	E	E	E	Check the program for errors but do not generate code.
	L			Use the preceding file descriptor as a library file whenever the COPY verb is encountered.
	M	M	M	Print a map showing the parameters of the user-defined item.
			N	Suppress output of source errors on the terminal.
		P		Do not generate trace calls and symbols.
		R		Produce a two-segment object program. The high segment contains the resident sections of the Procedure division; the low segment contains everything else. When the object program is loaded, LIBOL is added to the high segment.
	S	S	S	The source file has sequence numbers in columns 1-6 and comments starting in column 73.
	W	W	W	Rewind the magnetic tape.
		Z	Z	Zero the DECtape directory.
FORTRAN	A	A	A	Advance magnetic tape reel by one file.
	B	B	B	Backspace magnetic tape reel by one file.
			C	Generate a CREF-type cross-reference listing.
			D	List error message codes only.
			E	Print an octal listing of the binary program in addition to the symbolic listing. Must be accompanied by /M.
			I	Translate the letter D in column 1 as a space and treat the line as a normal FORTRAN statement.
			M	Include MACRO coding in output listing.
			N	Suppress output of error messages on the terminal.
		S		Produce code for execution on the KA10 if running on the KI10, and vice-versa.
	T	T	T	Skip to the logical end of magnetic tape.
	W	W	W	Rewind the magnetic tape.
		Z	Z	Zero the DECtape directory.

(continued on next page)

Table 1-2 (Cont)
Processor Switches

Processor	Source	Binary	Listing	Meaning
MACRO				
	A	A	A	Advance magnetic tape reel by one file.
	B	B	B	Backspace magnetic tape reel by one file.
			C	Produce listing file in a format acceptable as input to CREF.
			E	List macro expansions.
			F	Byte sizes match the format of the instruction.
			G	Byte sizes are two 18-bit fields.
			L	Reinstate listing (used after list suppression by S switch).
			M	Suppress ASCII text in macro and repeat expansion (SALL).
			N	Suppress error printouts on the terminal.
	O	O	O	Allow literals to occupy only one line.
	P	P	P	Increase the size of the pushdown list.
	Q	Q	Q	Suppress questionable (Q) error indications on the listing.
			S	Suppress listing.
	T	T	T	Skip to the logical end of magnetic tape.
	W	W	W	Rewind the magnetic tape.
			X	Suppress all macro expansions.
		Z	Z	Zero the DECtape directory.

Examples:

`.DEBUG TEST(N)`	Suppress typeout of errors during assembly.
`.COMPILE OUTPUT=MTA0:(W,S,M)/L`	Rewind the magtape (W), compile the first file, produce binary output for the KI10(S), and include the MACRO coding in the output listing (M). Output files are given the names OUTPUT.REL and OUTPUT.LST.
`.COMPILE/MACRO A=MTA0:(W,,Q)/L`	Rewind the magtape (W), compile the first file, and suppress Q (questionable) error indications on the listing. Note that when a binary switch is not present, the delimiting comma must appear.
`.COMPILE /MACRO A=MTA0:(,,Q)/L`	Compile file at current position of the tape and suppress Q error indications on the listing. Note that when the source and binary switches are not present, the delimiting commas must appear.

1.5.8 LOADER Switches

In complex loading processes, it may be necessary to pass switches to the LOADER to direct its operation. This is accomplished by the % character. The % has the same meaning as that of the / in the

LOADER'S command string (refer to the LOADER documentation). Also, like the /, the % takes a leading sign (+ or -) and one letter (or a sequence of digits and one letter) following it. Therefore, to set a program origin of 6000 for program C, the user types

```
.LOAD A,B,%6000OC,D
```

The COMPIL program allows more than one LOADER switch to be specified. For example:

```
.LOAD PROG %F/MAP
```

Refer to the LOAD command in Chapter 2 for a description of /MAP.

The most commonly used LOADER switches are:

a. %S Load with symbols.
b. %nO Set program origin to n.
c. %F Cause early search of the default libraries.
d. %P Prevent search of the default libraries.

CHAPTER 2
SYSTEM COMMANDS AND PROGRAMS

Although there is one operating system for all configurations of the DECsystem-10, some commands may not be included in each DECsystem-10. This is especially true of the DECsystem-1040, the basic system intended for small installations that do not want all of the system's features because of a constraint on core. Commands are deleted from the DECsystem-1040 by feature test switches (recognized by the beginning characters FT) defined at MONGEN time. In the standard DECsystem-1040, many of these switches are not set and, therefore, the corresponding commands are not available. This saves core but limits various features of the operating system. In the command descriptions that follow, the Characteristics section indicates if the switch is normally off in the DECsystem-1040. If not stated, the command is available on all DECsystem-10s.

In many cases, there are two commands to run a program. For example, the indirect command MAKE and the direct command R TECO both run the TECO program. In the DECsystem-1040, the switch implementing the indirect command may not be set but the switch implementing the direct command is always set. Therefore, it is always possible to run a program with the .R or .RUN command, even if the switch implementing the corresponding indirect command is off.

2.1 COMMANDS BY FUNCTIONAL GROUPS

Although the commands are arranged in alphabetical order for ease of reference, they can be divided into functional groups for ease of learning. These groups with their associated commands are as follows.

2.1.1 Job Initialization Commands

Since the system is limited to authorized persons, these commands protect the system from unauthorized use.

INITIA
LOGIN

2.1.2 Facility Allocation Commands

The monitor allocates peripheral devices, file structure storage, and core memory to users on request and protects these allocated facilities from interference by other users. Software provisions are incorporated in the monitor to differentiate the central station from the remote stations. Certain monitor commands, for example, ASSIGN and PLEASE, include station identification arguments to allow both user-access and allocation of system resources at any station. This feature gives the user considerable flexibility in allocating system facilities and directing input and output to the station of his choice. For example, by specifying a station number, the user can assign devices and input data from a peripheral device at a station other than his own. In addition, by using the LOCATE command, he can logically establish his job at a station other than his physical station. If the station identification argument is not included in a command, the system automatically directs input and output to the user's logical station. The user's logical station is the same as his physical station if he has not issued the LOCATE command.

When a nonsharable device is assigned to a job, it is removed from the monitor's pool of available resources. Any attempt by another user to reference or assign the device fails. Thus, a user should never leave the system without first returning his allocated facilities to the monitor pool. Allocated facilities are automatically returned to the monitor pool when the user deassigns them or kills his job. Until a user returns these facilities, no other users may utilize them except through operator intervention.

Assignable devices (i.e., nondisk and nonspooled devices) in the monitor's pool of available resources are designated as being either unrestricted or restricted devices. An unrestricted device can be assigned (ASSIGN command or INIT UUO) by any user. A restricted device can be assigned only by a privileged job (i.e., a job logged in under [1,2] or running with the JACCT bit set). However, a nonprivileged user can have a restricted device assigned to him via the MOUNT command. This command allows operator intervention for the selection or denial of a particular device; thus the operator can control the use of the assignable devices. This is particularly useful when there are multiprogramming batch and interactive jobs competing for the same devices. The restricted status of a device is set or removed by the OPSER commands :RESTRICT and :UNRESTRICT.

The facility allocation commands are as follows:

ASSIGN	DISMOUNT	REASSIGN	SET DENSITY
CLOSE	FINISH	SET BLOCKSIZE	SET DSKPRI
CORE	LOCATE	SET CDR	SET HPQ
DEASSIGN	MOUNT	SET CPU	SET SPOOL
			SET TTY or TTY

2.1.3 Source File Preparation Commands

These commands call the system editing programs in order to create or edit a specified text file. The system editing programs available are LINED (a line-oriented editor) and TECO (a character-oriented editor). In general, the editor used to create the file should be used for editing, since LINED requires line-blocked files and TECO does not.

CREATE
EDIT
MAKE
TECO

2.1.4 File Manipulation Commands

The commands in this group allow the user to manipulate his files to any desired extent. He can list source files, and DECtape and disk directories on the terminal or the line printer, possibly via the spooling mechanism. He can delete or rename files from disk and DECtape. In addition, the user can transfer files between standard I/O devices, perform conversion between various core image formats, and read and write various directory formats. Disk space can be either allocated for a new file or re-allocated for an existing file. Finally, the user can place files in the system queues and obtain listings of entries in those queues.

ALCFIL	DIRECT	PRESERVE	REWIND
BACKSPACE	EOF	PRINT	SKIP
BACKUP	FILE	PROTECT	SUBMIT
COPY	FILEX	QUEUE	TPUNCH
CPUNCH	LIST	RENAME	TYPE
DELETE	PLOT	RESTORE	UNLOAD
			ZERO

2.1.5 Object Program Preparation Commands

The commands in this group are used to prepare object programs and save the user's core area as one or two files.

COMPILE	EXECUTE	LOAD
CREF	FUDGE	SAVE
DEBUG	FUDGE2	SSAVE

2.1.6 Object Program Control Commands

By using the commands in this group, the user can load core image files from retrievable storage devices (i.e., disk, DECtape, magnetic tape). These files can be retrieved and controlled from the user's terminal. Files stored on disk and DECtape are addressable by name. Files on magnetic tape require the user to pre-position the tape to the beginning of the file. Refer to DECsystem-10 Monitor Calls, Chapter 1, for a description of the job data area locations referenced by the command descriptions in this group.

CONT (CCONT)	HALT	REENTER
DDT	JCONT	RUN
GET	R	START (CSTART)

2.1.7 Object Program Examination Commands

The commands in this group aid the user in examining and analyzing his object program. Dumps of the user's core area can be taken and later processed by the system program DUMP according to the arguments specified by the user.

D (deposit)
DCORE
DUMP
E (examine)

2.1.8 Multiple Job Control Commands

There is not necessarily a one-to-one relationship between jobs and terminals. A terminal must initiate a job, but the user or operator may issue commands to permit a job to float in a detached state where it is not associated with a particular terminal. Thus, more than one job may be controlled from the same terminal.

ATTACH	CSTART	OPSER
CCONT	DETACH	REATTA

2.1.9 Job Termination Command

When the user leaves the system, all facilities allocated to his job must be returned to the monitor facility pool so that they are available to other users.

KJOB

2.1.10 Sending Messages

The commands in this group allow the user interconsole communication with other users of the system or with operators at any station. In addition, the user may record information in a disk file to be read by the operations staff at a later time.

GRIPE
PLEASE
SEND

2.1.11 Job Information Commands

The user can obtain various job-related information with this group of commands. This information includes the number of his job, the quotas for each file structure associated with his job, and the running time and disk space that his job has used. In addition, the user may type or modify his file structure search list.

DSK	QUOLST	SET TIME
PJOB	SETSRC	SET WATCH
		TIME

2.1.12 System Information Commands

With the commands in this group, the user is able to obtain system status information, including the time of day, the list of available devices, file structures, and physical units not in file structures, the scheduled use of the system, and the location of a specific peripheral device.

DAYTIME	SCHED	VERSION
RESOURCES	SYSTAT	WHERE

ALCFIL program

Function

The ALCFIL program enables the user to allocate space for a new file or reallocate space for an existing file in one contiguous region on the disk. The size of the region is restricted by the size of the cluster count field (usually 512) times the cluster size of the file structure times the number of pointers in a disk device data block (not less than 10).

Command Format

R ALCFIL

The ALCFIL program responds with

```
/H FOR HELP
FILE?
```

The user may respond with

```
dev:file.ext [proj,prog]
or /H (for help)
or /X (to exit)
```

where dev: is a file structure or physical unit name. If dev: is omitted, DSK is assumed. If one of the other arguments is omitted, 0 is assumed. If a filename is specified, the number of blocks presently allocated, if nonzero, is typed. ALCFIL responds with

```
ALLOCATE?
```

User may type N or N,M (decimal numbers)

N = total number of blocks to be allocated for the file.
M = logical block within the file structure or unit (depending on dev:) where the allocation is to begin.

If the total number of blocks requested cannot be allocated (because of disk quotas), a partial allocation is given and the message

```
PARTIAL ALLOCATION ONLY
```

is typed. The user can issue the DIRECT command with the ALLOCATE switch to determine the number of blocks allocated. If the new blocks can be allocated, the message

```
ALLOCATED
```

is typed.

Since an extended ENTER (refer to DECsystem-10 Monitor Calls) is executed to allocate the new blocks, the file need not exist before the blocks are allocated.

Characteristics

The R ALCFIL command:

Places the terminal in user mode.
Runs the ALCFIL program, thereby destroying the user's core image.

Associated Messages

Refer to Chapter 4.

Example

```
.R ALCFIL

/H FOR HELP
FILE? TEST4.TST
ALLOCATE? 2000

ALLOCATED
FILE? TEST5.TST
ALLOCATE? 1000

ALLOCATED
FILE? TEST5.TST
1000 BLOCKS ALREADY ALLOCATED
ALLOCATE? 500

ALLOCATED
FILE? DSKB:FILEA
ALLOCATE? 3000

PARTIAL ALLOCATION ONLY
FILE? /X

EXIT

.DIR/ALLOC

FILEA          175  <057>   14-APR-72     DSKB:


.
```

ASSIGN command

Function

The ASSIGN command allocates an I/O device to the user's job for the duration of the job or until a DEASSIGN command is given. This command, applied to DECtapes, clears the copy of the directory currently in core, forcing any directory references to read a new copy from the tape. (Refer to DECsystem-10 Monitor Calls for further details.)

Although DECtape is the only device that should be ASSIGNed before use, to ensure that the monitor has a copy of the proper DECtape directory in its core area, it is wise to ASSIGN all devices, such as magnetic tape, before use.

Command Formats

1. ASSIGN phys-devn log-dev

   phys-devn = any physical device listed in Table 1-1 in Paragraph 1.4.2.2, followed by a 1-to-3 digit number representing a specific unit, or any file structure name. This argument is required. With this command format, the monitor attempts to assign the device specifically requested. If unable to assign the device, the monitor types an appropriate message (refer to Chapter 4).

   log-dev = a logical name assigned by the user. This argument is optional. Except for disk devices, only one logical name can be assigned to a physical device. Subsequent ASSIGN commands to all devices except disk devices replace the old logical name with the new one. Logical names are disassociated from all devices by the DEASSIGN command.

2. ASSIGN phys-devSnn log-dev

   phys-devSnn = any physical device followed by the letter S and a 1 or 2 digit number representing a specific station, or any file structure name. This argument is required. With this command format, the monitor attempts to assign a device at the requested station. An appropriate message is typed if the device cannot be assigned (refer to Chapter 4).

   log-dev = same as above.

3. ASSIGN phys-dev log-dev

   phys-dev = any physical device followed by a null argument implying any device of the designated type, or any file structure name. This argument is required. With this command format, the monitor attempts to assign the requested device at the user's logical station. If this type of device does not exist at the user's logical station, the monitor attempts to assign the device at the central station. If unable to assign the device, the monitor types an appropriate message (refer to Chapter 4).

   log-dev = same as above.

Characteristics

The ASSIGN command:

Leaves the terminal in monitor mode.

Restrictions

A comma may not be used to separate the logical and physical device names. If a comma is used, the monitor terminates its scan at the comma; therefore, the logical name is not assigned.

Non-privileged jobs (i.e., jobs not logged in as [1,2] or running with JACCT set) can only use this command to allocate unrestricted I/O devices. Restricted devices can be obtained by non-privileged jobs via the MOUNT command. The ASSIGN when issued by a privileged job allocates both restricted and unrestricted devices.

Associated Messages

Refer to Chapter 4.

Examples

```
.ASSIGN LPT2:↵
LPT2 ASSIGNED
```

The user assigns a specific line printer (LPT2).

```
.AS CDRS2:↵
CDR4 ASSIGNED
```

The user assigns any available card reader at station 2.

```
.ASSIGN TTY1:LPT:↵
TTY1 ASSIGNED
```

The user assigns TTY 1 and gives it logical name LPT.

```
.AS LPT:↵
LPT4 ASSIGNED
```

The user assigns any available line printer. The LPT chosen is at either the user's station or the central station.

```
.ASSIGN DTA2:↵
?DEVICE NOT ASSIGNABLE
```

A non-privileged user attempted to allocate a restricted device (DTA2).

```
.ASSIGN DTA:↵
DTA4 ASSIGNED
```

The user then uses the generic device name (DTA) to obtain the device. He could have used the MOUNT command to assign the restricted device DTA2.

ATTACH command

Function

The ATTACH command detaches the current job, if any, and connects the terminal to a detached job.

Command Format

ATTACH job [proj,prog]

job = the job number of the job to which the terminal is to be attached. This argument is required.

[proj,prog] = the project-programmer number of the originator of the desired job. This argument may be omitted if it is the same as the job to which the terminal is currently attached. The operator (device OPR) or a user logged-in under [1,2] may always attach to a job although another terminal is attached, provided he specifies the proper [proj,prog].

To prevent users from attaching the jobs without knowing the PASSWORD associated with the job, a new job is temporarily created when the [proj,prog] argument is specified. This temporary job runs LOGIN to check the password. This can result in the current job not being able to attach to the specified job if the job capacity of the system would be exceeded with the creation of the temporary job. However, the current job is still detached even if there are no available jobs. The operator or any job logged-in as [1,2] can always attach to another job since they do not require the creation of a temporary job.

Characteristics

The ATTACH command:

Leaves the terminal in monitor mode.
Does not require LOGIN.
Depends on FTATTACH which is normally absent in the DECsystem-1040.

Restrictions

Remote users cannot attach to jobs with a project number of 1. Batch users cannot issue this command.

Associated Messages

Refer to Chapter 4.

Examples

1.
```
.ATT 1⏎
FROM JOB 5

.
```
The user attaches to job 1 from job 5. The two jobs have the same [proj,prog] and therefore, the argument is not required.

2.
```
.LOG 27,235⏎
JOB 7 5S04 TTY25
PASSWORD:
1634 23-FEB-72 WED
```
The user logs-in and is given job 7. TTY25 is now attached to job 7.

```
.ATTACH 36 [50,27]⏎
FROM JOB 7
PASSWORD:
```
The user attaches to an existing job (36) and thereby detaches his current job (7). Since the [proj, prog] associated with job 36 is different than the user's, he must specify the [proj,prog] of the desired job. The system then requests the PASSWORD. If the given PASSWORD is correct, the terminal is attached to job 36.

```
.
```
The terminal is attached to job 36.

```
.ATTACH 7⏎
?CAN'T ATT TO JOB
```
The user attempts to attach to job 7. The command fails because the [proj,prog] of job 7 is not the same as the [proj,prog] of job 36. The terminal is still attached to job 36.

```
.K/F⏎
JOB 36,USER [50,27] LOGGED OFF TTY25 1635 23-FEB-72
RUNTIME 0MIN,00.34SEC
```
The user killed job 36. The terminal is currently not attached to any job.

```
.ATTACH 7⏎
?CAN'T ATT TO JOB
```
Since the terminal is currently not attached to a job, the command fails because there is no [proj, prog] to compare with the [proj,prog] of job 7.

```
.ATTACH 7 [27,235]⏎
PASSWORD
```
The command is accepted and the PASSWORD is requested. The message FROM JOBn is not output since the terminal was not attached to a job.

```
.
```
The terminal is attached to job 7.

BACKSPACE command [1]

Function

The BACKSPACE command spaces a magnetic tape backward a specified number of files or physical records. This command, depending on its arguments, is equivalent to the following PIP command strings:

```
MTAn: (M #nB) ←
MTAn: (M #nP) ←
```

Command Formats

1. BACKSPACE MTAn: x FILES

   skips backward x files.

2. BACKSPACE MTAn: x RECORDS

   skips backward x records.

Characteristics

The BACKSPACE command:

Leaves the terminal in monitor mode.
Runs the PIP program.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.BAC MTA2: 7 RECORDS )
.BACKSP MTA3: 11 FILES)
```

[1] This command runs the COMPIL program, which interprets the command before running the PIP program.

BACKUP program

Function

The BACKUP program enables the user to save disk files on magnetic tape (MTA), DECtape (DTA), or disk (DSK). The save can be of the entire disk or selected subsets of the disk.

The BACKUP program places data in the following files:

1. BACKUP SET file

   This file contains the data saved on the backup medium (MTA, DSK, DTA) with one BACKUP command. When data is saved on disk, the BACKUP SET file is one file with file delimiting control words. When written on magnetic tape, it is several files written in buffered binary mode.

   The BACKUP SET file is composed of a BACKUP header, a user set, and a BACKUP trailer. The BACKUP header is one block in length and contains information concerning the creation of the BACKUP SET. This information includes the name and date of creation, the name of the system, and the user identification.

   The USER SET contains the directories and the files associated with the directories for all user areas. Even if a user has files on more than one file structure, his files will be saved together. For example, all files on all file structures for user 1,2 will be stored before the files for user 1,3. In other words, the user set is ordered according to project-programmer number, not according to file structures. However, within individual project-programmer numbers, all files on one file structure are saved before files on another.

   The BACKUP trailer contains information about the user set that was just created. This information includes the time, the date, and the length of the user set. The BACKUP trailer immediately follows the user set.

2. INDEX file

   This file contains the directories and the filenames of all the disk areas that have been saved on the backup medium. In addition, it contains the relative block number in the BACKUP SET file where each element (file) begins. The index file is the last file written on the backup medium and is separated from the BACKUP SET file. It can be saved on DECtape and can be listed, if desired.

3. COMMAND RECOVERY file

   This file contains information that indicates how much of the user's command has been processed and how much of the command remains. It is updated at every check point in order to make crash recovery possible.

4. Log File

   The user is given a log file to aid him in error analysis. This file contains a record of either all actions performed by the BACKUP and RESTORE programs or only errors encountered in processing. The log file is a listing file.

BACKUP program (Cont)

Command Format

R BACKUP

The user may type any of the following commands after the slash output by the BACKUP program. These commands are stored in core and are processed only when a START command is given by the user. All commands are terminated with a carriage return. Multiple files may be specified in one command string by separating the filenames with commas. The full wildcard construction may be used to replace the filename, the extension, or the directory (refer to Paragraph 1.4.2.4).

Command	Explanation
BACKSPACE FILE	Backspaces the magnetic tape to a user file header and positions the tape immediately before the header.
BACKSPACE SET	Backspaces the magnetic tape to a BACKUP header and positions the tape either immediately before the header or to the beginning of the tape if there is no BACKUP header (i.e., there is only one BACKUP SET on the tape).
BACKSPACE UFD	Backspaces the magnetic tape to a UFD header and positions the tape immediately before the header.
BACKUP dev2: file descriptor ← dev1: [proj,prog] /switch	Writes the designated file on dev2 from dev1. The user may specify files to be taken from and/or written to a user's area other than his own, provided that he has access privileges to the user areas. The device arguments are required. /switch = /EXCEPT file descriptor Indicates the files and/or areas that should not be written as BACKUP files.
DELETE dev: file.ext	Deletes the file named from the specified device. The specified device must be the device for which a BACKUP has been taken.

DENSITY MTAn:x	Sets the magnetic tape density as specified by x.
	x = 2 200 bpi
	x = 5 556 bpi
	x = 8 800 bpi
	The default is the system standard defined at MONGEN time.

DUMP ON dev: file.ext	Dump the contents of the BACKUP set file beginning at the present position and ending at the next file control word. All types of errors are ignored. The device on which the dump is to be written may not be a listing device.
ERROR DUMP dev:	Returns to the last file control word and dumps the file on the device specified if a transmission error occurs during the backup of any file.

(continued on next page)

Command	Explanation
INDEX dev: file.ext	Writes the index file with the designated filename on the device named. The index file is also written on the disk and saved as the last file on the backup medium. The default is DSK:MTnnnn.BKP where nnnn corresponds to the tape sequence number.
LOG dev: file.ext /switch	Writes a file on the specified device which contains a record of the operations performed. The default is DSK:BACKUP.LOG. /Switch = /ERROR Logs only the errors. This switch is optional. If omitted, all operations are recorded.
PARITY dev: ODD or EVEN	Specifies the parity on magnetic tape as odd or even. The default is odd.
REWIND dev:	On magnetic tape, closes the backup set and rewinds the tape. On disk, closes the backup set.
START	Begins execution of a series of commands sent previously. Commands are not processed until a START command is given. If there are no commands to be processed when the START is executed, the command recovery file is searched for executable commands.
UNLOAD dev:	Performs a rewind and unload to magnetic tape.

The user may restart the BACKUP program at any time. By issuing a ↑C ↑C START sequence, the user can cancel present operations and specify new commands. A ↑C START sequence deletes the command recovery file if the next command given to the BACKUP program is not START. If START is the next command to BACKUP, the command recovery file is scanned, and the BACKUP program continues according to the information in the file. A ↑C CONT sequence does not delete the command recovery file, but completes the current requests. After all requests have been completed, the BACKUP program closes out the log file,.and types BACKUP COMPLETED and an asterisk on the user's terminal indicating that it is ready for more requests.

MTA rewinds due to the magnetic tape being filled are actually rewind and unload operations to insure that the magnetic tape is not overwritten. When the BACKUP program reaches completion, the magnetic tape last written on remains in position unless a REWIND command is given.

Characteristics

The R BACKUP command:

Runs the BACKUP program, thereby destroying the user's core image.

BACKUP program (Cont)

Associated Messages

Refer to Chapter 4.

Examples

```
.R BACKUP↵

/BACKUP MTA1:←DSKB:[10,225]*.*↵
/INDEX DSKC:BAKFIL.LST↵
/START↵

!$BACKUP COMPLETED AT 16:45:22
/↑C

.
```

CLOSE command

Function

The CLOSE command terminates any input or output currently in progress on the specified device, and automatically performs the CLOSE UUO (refer to DECsystem-10 Monitor Calls). Files are CLOSEd, but not RELEASEd, and logical names and device assignments are preserved. Since most programs CLOSE files when they finish performing a command string, the CLOSE command is provided for the occurrence of a program not terminating or a program being debugged. This command causes any disk files being written to be entered into the user's UFD. If a CLOSE is not done, the next RESET by a command (R, RUN, GET) or program will delete the partially written file.

Command Format

CLOSE dev

dev = the logical or physical name of the device on which I/O is to be terminated. This argument is optional.

If dev is omitted, I/O is terminated on all devices, except for the job's controlling terminal, and all files are CLOSEd.

Characteristics

The CLOSE command:

Leaves the terminal in monitor mode.
Requires core.
Depends on FTFINISH which is normally absent in the DECsystem-1040.

Restrictions

The user cannot continue, but can start at the beginning or enter DDT.

Associated Messages

Refer to Chapter 4.

Examples

```
.CLOSE PTR:)
.
.CLOSE DEVA:)
.
.CLOSE)
.
```

COMPILE command [1]

Function

The COMPILE command produces relocatable binary files (.REL files) and/or compilation listings for the specified source program files. The assembler or compiler used is determined by the source file extension or by switches in the command string. If no switches appear in the command string, the following translators are used:

Source File Extension	Translator Used
.ALG	ALGOL compiler
.BLI	BLISS compiler[2]
.CBL	COBOL compiler
.F4	FORTRAN compiler (F40)
.MAC	MACRO assembler
.P11	MACX11 assembler[2]
.SNO	SNOBOL compiler[2]
Other than above, or null	Standard processor, which is usually FORTRAN at the beginning of the command string (refer to Paragraph 1.5.6).

NOTE

If a source file has a recognizable processor extension (see above), the processor cannot be changed with a switch. The only time that a processor can be specified with a switch is when the source file has a non-recognizable processor extension or a null extension.

Normally the source file is translated if there is no corresponding binary (.REL) file or if the source file's date and time is later than or equal to the binary file's date and time. If the binary file is newer than the source file, the source file is not translated and the current .REL file is used. However, switches can be used to override this action.

Each time the COMPILE, LOAD, EXECUTE, or DEBUG command is executed, the command with its arguments is remembered in a temporary file on disk, or in core if the monitor has the TMPCOR feature. Therefore, the filename used last can be recalled for the next command without specifying the arguments again (refer to Paragraph 1.5).

The COMPILE command accepts several command constructions: the @ construction (indirect commands), the + construction, the = construction, and the < > construction. Refer to Paragraph 1.5 for a complete description of each of these constructions.

[1] This command runs the COMPIL program, which interprets the command before running the appropriate processor.

[2] SNOBOL, BLISS, and MACX11 (the PDP-11 assembler for the PDP-10) will be recognized as processors only if the appropriate assembly switches are set. However, these assembly switch settings are not supported.

Command Format

COMPILE list

list = a single file specification, or a string of file specifications separated by commas. A file specification consists of a device name, a filename with or without an extension, and a directory name.

The following switches can be used to modify the command string. These switches can be temporary or permanent switches (refer to Paragraph 1.5.5). Note that all the switches allowed with the LOAD, EXECUTE, and DEBUG commands can be used with the COMPILE command. However, only the switches pertinent to COMPILE are listed below; the others are ignored.

/ALGOL	Compile the file with ALGOL. Assumed for files with the extension of .ALG.
/BLISS[1]	Compile the file with BLISS. Assumed for files with the extension of .BLI.
/COBOL	Compile the file with COBOL. Assumed for files with the extension of .CBL.
/COMPILE	Force a compilation on this file even though a binary file exists with a newer date and time than the source file. This switch is used to obtain an extra compilation (e.g., in order to obtain a listing of the compilation) since normally compilation is not performed if the binary file is newer than the source file.
/CREF	Produce a cross-reference listing file on the disk for each file compiled for later processing by the CREF program. These files have the filename of the source file and the extension of .CRF. The file can then be listed with the CREF command. However, with COBOL files, the cross-referenced listing is always appended to the listing file. No additional command need be given to obtain the listing.
/FORTRAN	Compile the file with FORTRAN. Assumed for files with the extension of .F4 and all files with non-recognizable processor extensions (if FORTRAN is the standard processor).

[1] BLISS will be recognized as a processor only if the appropriate assembly switch is set. However, this assembly switch setting is not supported.

COMPILE command (Cont)

Command Format (cont)

/FUDGE	Create a disk file containing the names of the .REL files produced by the command string. When the FUDGE command is given, PIP reads this file in order to generate a library REL file. Arguments to this switch are: /FUDGE:dev:file.ext [proj,prog] dev: - the device on which to write the file. DSK: is assumed. file.ext - the name of the library file. The filename is required. If the extension is omitted, it is assumed to be .REL. [proj,prog] - the directory in which to place the file. The user's directory is assumed if none is given. This switch is permanent in the sense that it pertains to all .REL files generated by the command string.
/LIST	Generate a disk listing file, for each file compiled, with the filename of the source file and the extension of .LST. These files can be listed later with the LIST command. Unless this switch is specified, listing files are not generated except in COBOL; COBOL listings are always generated.
/MACRO	Assemble the file with MACRO. Assumed for files with extension of .MAC.
/MACX11[1]	Assemble the file with MACX11. Assumed for files with extension of .P11.
/NOCOMPILE	Complement the /COMPILE switch by not forcing a compilation on a source file whose date is not as recent as the date on the binary file. /NOCOMPILE is the default action.
/NOLIST	Do not generate listing files. This is the default action except for COBOL files; COBOL listings are always generated.
/SNOBOL[1]	Compile the file with SNOBOL. Assumed for files with an extension of .SNO.

[1]SNOBOL and MACX11 (the PDP-11 assembler for the PDP-10) will be recognized as processors only if the appropriate assembly switches are set. However, these assembly switch settings are not supported.

Characteristics

The COMPILE command:

Leaves the terminal in monitor mode.
Runs the appropriate processor.

Restrictions

The wildcard construction cannot be used.

Associated Messages

Refer to Chapter 4.

Examples

```
.COMPILE PROG,TEST.MAC,MANAGE/COBOL
```

Compiles PROG (with null extension) with FORTRAN, TEST.MAC with MACRO, and MANAGE (with null extension) with COBOL only if REL files do not exist with later dates. A listing file is generated only for MANAGE. The files generated are PROG.REL, TEST.REL, MANAGE.REL, and MANAGE.LST.

```
.COMPILE /LIST SIGN.MAC,TABLES/NOLIST,MULTI.ALG
```

Compiles SIGN.MAC with MACRO, TABLES (with null extension) with FORTRAN, and MULTI.ALG with ALGOL. Listing files are generated for SIGN.MAC and MULTI.ALG.

```
.COMPILE/CREF/COMPILE DIVIDE,SUBTRC,ADD
```

Forces a compilation of the source files although current .REL files exist and generates cross-referenced listing files. The files created are DIVIDE.CRF, DIVIDE.REL, SUBTRC.CRF, SUBTRC.REL, ADD.CRF, and ADD.REL.

```
.COMPILE /FUDGE:MONITR.REL@LIBALL
```

Compiles the files contained in the command file LIBALL and enters the names of all the REL files generated in a temporary disk file. When the FUDGE command is given, PIP generates the library REL file with name MONITR.REL. The library is created with the REL files in the same order as they were specified in the command file.

COMPILE command (Cont)

Examples (cont)

```
.COMPILE OUTPUT=MTA0:(W,S,M)/L
```

Rewinds the magnetic tape (W), compiles the first file with FORTRAN, produces binary output for the KA10 (S), and includes the MACRO coding in the output listing (M). These switches are processor switches (refer to Paragraph 1.5.7). A listing file is generated with the name OUTPUT.LST, along with the file OUTPUT.REL.

CONTINUE command

Function

The CONTINUE command starts the program at the saved program counter address stored in .JBPC by a HALT command (↑C) or a HALT instruction. Refer to DECsystem-10 Monitor Calls for a description of the job data area.

Command Format

CONTINUE

Characteristics

The CONTINUE command:

Places the terminal in user mode.
Requires core.
Does not require LOGIN.

Associated Messages

Refer to Chapter 4.

Example

```
.RUN LOOP)          Run a program called LOOP in your disk area.
↑C
↑C                  Stop the program.
.DAYTIME)           Check the time of day.
23-FEB-72 16:33:10
.CONT)              Continue the program.
```

COPY command [1]

Function

The COPY command transfers files from one standard I/O device to another. The command string can contain one device output specification and any number of input specifications. The equal sign separates the destination (output) side from the source (input) side. This command runs PIP and performs the basic PIP function of transferring files.

Command Format

COPY dev: file.ext [proj,prog] <nnn> = dev: file.ext [proj,prog], file.ext [proj,prog], ...

dev: = a physical or logical device name. If the device name is omitted, DSK: is assumed.

file.ext = the name of the file(s) to be used on input or for output. If the output filename is omitted, the input filemane is assumed. PIP combines the files if many input files are being transferred to one output file. If many input files are being transferred to the same number of output files, PIP uses the /X switch to keep the files separate. The wildcard construction is allowed.

[proj,prog] = the disk area in which either the files are to be read or written. If this argument is omitted, the user's default disk area is assumed. The user may transfer files to or from another area only if he has access to the area.

<nnn> = the protection code to be given to the output file(s). If omitted, the system standard is assigned.

Switches can be passed to PIP by enclosing them in parentheses in the COPY command string. When COMPIL interprets the command string, it passes the switches on to PIP.

Characteristics

The COPY command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

[1] This command runs the COMPIL program, which interprets the command before running PIP.

Examples

```
.COPY=DTA3: FILNAM.MAC, MANY.CBL, COMMON.ALG )
```

The three files from DTA3 are transferred to the user's disk area with the same filenames.

```
.COPY DTA3: OUTPUT=*.* )
```

All files in the user's disk are transferred to one file on DTA3 with the name OUTPUT.

```
.COPY FILEA.*=DTA1: SOURCE.* )
```

The input files on DTA1 named SOURCE with any extension are transferred to DSK with the filename FILEA and the same extension. The number of output files equal the number of input files.

```
.COPY YOURS.CBL [20, 17] = MINE.CBL )
```

The file MINE.CBL from the user's disk area is transferred to [20,17] disk area with the name YOURS.CBL. The user must have privileges to write in area [20,17].

COPY program

Function

The COPY program is a DECtape copy routine that allows the user to

1. Copy the entire contents of an input DECtape to an output DECtape.
2. Zero all blocks on an output DECtape and clear the directory.
3. Perform a word-by-word comparison of two DECtapes.
4. Load a bootstrap loader and write it in blocks 0, 1, and 2 of the output DECtape.

Command Format

```
.R COPY↵
*output DTA:=input DTA: /switches
```

/switches = one or more of the following switches. Switches are preceded by a slash or enclosed in parentheses and can appear anywhere in the command string.

/C	Copy all blocks from the input DECtape to the output DECtape.
/G	Do not restart the program after a parity error. Output an error message and continue the program.
/H	Type the available switches and their meanings.
/L	Load the bootstrap loader into a core buffer. COPY expects the loader to be on logical device PTR in the file named BSLDR.REL. Note that COPY must be SAVed if the loader is to be preserved with the COPY core image.
/N	Suppress the directory listing.
/T	Write the bootstrap loader in blocks 0, 1, and 2 of the output DECtape. This switch accepts, as input from the terminal, a core bank or offset. The loader is offset and then written on the tape. core bank = nnnK (16K to 256K) offset = 1000 to 777600 octal
/V	Verify the similarities of the two DECtapes by performing a word-by-word comparison and typing on the terminal the number of discrepancies discovered.
/Z	Zero all blocks of the output DECtape and clear the directory.
/6	Look for the directory in PDP-6 format (i.e., in block one instead of block 144).

Command Format (cont)

If no switches are specified, /C (copy) and /V (verify) are assumed by default. Note that upon completion, the directory in core may not agree with the directory of the output DECtape. The output DECtape should be reassigned to guarantee that the directory in core is up-to-date.

Characteristics

The R COPY command:

Places the terminal in user mode.
Runs the COPY program, thereby destroying the user's core image.

Associated Messages

Refer to Chapter 4.

Examples

`.R COPY↵`	Run COPY
`*DTA7: = DTA3:↵`	Copy the contents of DTA3 to DTA7 and determine if the two DECtapes are the same (default condition). If the DECtapes disagree, the number of discrepancies is typed on the terminal.
`*DTA2:/Z=↵`	Zero all blocks and clear the directory on DTA2.
`*↑C`	Return to monitor mode.
`.ASSIGN DSK:PTR:↵`	The bootstrap loader must be on logical device PTR.
`.RENAME BSLDR.REL=DTBOOT.REL↵`	COPY expects the bootstrap loader to be named BSLDR.
`.R COPY↵`	Run COPY
`*/L↵`	Load the bootstrap loader into a core buffer.
`*↑C`	Return to monitor mode.
`.SAVE DSK:COPY↵`	Save COPY so that the bootstrap loader is preserved with the COPY core image.

COPY program (Cont)

Examples (cont)

```
.START↵                                Start the COPY program.
*DTA5:/T=↵                             Write the bootstrap loader in blocks 0, 1, and 2 of
                                       DTA5.
TYPE CORE BANK AND OFFSET FOR DTBOOT
                                       Respond with size of core bank or offset.
64K↵                                   Size of core bank (64K core bank = 177000 offset,
                                       top of core -1000)
*↑C                                    Return to monitor mode.
```

CORE command

Function

The CORE command types or modifies the amount of core assigned to the user's job. Because programs usually allocate core, the user generally does not need this command. It is included for completeness and is used more frequently in non-swapping systems than in swapping systems.

If the job is locked into core, this command with a nonzero argument cannot be satisfied and therefore gives an erroneous return.

Command Format

CORE n

n = a decimal number. This argument is optional.

If n is omitted, the monitor types out the amount of core used and does not change the core assignment.

If n = 0, the low and high segments disappear from the virtual addressing space of the job.

If n >0, n represents the total number of blocks of core to be assigned to the job from this point on.

If n is less than high plus minimum low segment size, n plus high segment size is assumed.

Core arguments can be specified in units of 1024 words or 512 words (a page) by following n with the letter K or P, respectively. For example, 3P represents 3 pages or 1536 words. If K or P is not specified, K (1024 words) is assumed.

On systems with the KA10 processor (DECsystem-1040, 1050, or 1055), 1024 words is the minimum unit of allocation and therefore, all arguments are rounded up to the nearest multiple of 1024 words. For example, 3P on the KA10 is treated the same as 2K.

Characteristics

The CORE command:

Leaves the terminal in monitor mode.
Does not operate when a device is currently transmitting data.

CORE command (Cont)

Associated Messages

Refer to Chapter 4.

Examples

```
.CORE 5)

.CORE)
5+0/46K CORE
VIR. CORE LEFT = 274

.CORE 10P)
.CORE
10+0/93P CORE
VIR.CORE LEFT = 549P
```

CPUNCH command

Function

The CPUNCH command is used to place entries into the card punch output queue. This command is equivalent to the following form of the QUEUE command:

```
QUEUE CDP: job name = list of input specifications
```

Command Format

```
CPUNCH jobname = list of input specifications
```

jobname = name of the job being entered into the queue. The default is the name of the first file in the request, not the name of the first file given. These differ when the the first file given does not yet exist.

input specifications = a single file specification or a string of file specifications, separated by commas, for the disk files being processed. A file specification is in the form dev:file.ext [proj,prog].

dev: = any file structure to which CDPSPL will have access; the default is DSK:.

file.ext = names of the files. The filename is optional. The default for the first filename is *, the default for subsequent files is the last filename used. The extension can be omitted; the default is .CDP.

[proj,prog] = a directory to which the user has access; the user's directory is assumed if none is specified.

Note that if all arguments to the command are omitted (i.e., only the command name is given), the listing of all entries in the card punch queue for all jobs of all users is output.

The wildcard construction can be used for the input specifications. Switches that aid in constructing the queue entry can also appear as part of the input specifications. These switches are divided into three categories:

1. Queue-operation - Only one of these switches can be placed in the command string because they define the type of queue request. The switch used can appear anywhere in the command string.
2. General - Each switch in this category can appear only once in the command string because they affect the entire request. The switch used can appear anywhere in the command string.
3. File control - Any number of these switches can appear in the command string because they are specific to individual files within the request. The switch used must be adjacent to the file to which it applies. If the switch precedes the filename, it becomes the default for subsequent files.

CPUNCH command (Cont)

Command Format (cont)

The following switches can be used with the CPUNCH command.

Switch	Explanation	Category
/AFTER:tt	Process the request after the specified time; tt is either in the form of hhmm (time of day) or +hhmm (time later than the current time). The resulting AFTER time must be less then the DEADLINE time. If the switch, or the value of the switch, is omitted, no AFTER constraints are assumed.	General
/BEFORE:t	Queue only the files with a creation date before time t where t is in the form dd-mmm-yy hhmm. If the switch, or the value of the switch, is omitted, no BEFORE constraints are assumed.	General
/BEGIN:n	Start the output on the nth card. The default is to begin output on the first card.	File Control
/COPIES:n	Repeat the output the specified number of times. N must be less than 64. If more than 63 copies are needed, two separate requests must be made. If this switch is not specified, the default is 1.	File Control
/CREATE	Make a new entry into the card punch output queue. This switch is the default for the queue-operation switches.	Queue Operation
/DEADLINE:tt	Process the request before the specified time; tt is either in the form hhmm (time of day) or +hhmm (time later than the current time). The resulting DEADLINE time must be greater than the AFTER time. If the switch, or the value of the switch, is omitted, no DEADLINE constraints are assumed.	General
/DISPOSE:DELETE	Delete the file after spooling.	File Control
/DISPOSE:PRESERVE	Save the file after spooling. This is the default for all files except files with extensions .LST, .TMP, or .CDP (if protection of .CDP is 0xx).	File Control

Command Format (cont)

Switch	Explanation	Category
/DISPOSE:RENAME	Rename the file from the specified directory immediately, remove it from the logged-out quota, and delete it after spooling. This is the default for files with extensions of .LST, .TMP, and, if the protection is 0xx, .CDP.	File Control
/F	List the entries in the card punch queue, but do not update the queues. Therefore, the list may not be an up-to-date listing but the listing will be faster than with /LIST.	Queue Operation
/FORMS:a	Place the output on the specified form. The argument to the switch must be six alphabetic characters. The default is that normal forms are used.	General
/KILL	Remove the specified entry from the card punch queue. This switch can be used for deleting a previously-submitted request as long as the request has not been started by the Spoolers.	Queue Operation
/LIMIT:n	Limit the output to the specified number of cards.	General
/LIST	After updating the queues, list the entries in the card punch queue; if this switch, along with all other switches, is omitted, all entries for all jobs of all users are listed.	Queue Operation
/MODIFY	Alter the specified parameters in the job. This switch requires that the user have access rights to the job. It can be used for altering a previously submitted request as long as the request has not been started by the Spooler.	Queue Operation

CPUNCH command (Cont)

Command Format (cont)

Switch	Explanation	Category
/NEW	Accept the request even if the file does not yet exist. An appropriate error message is given if the file does not exist by the time the request is processed by the spooler.	File Control
/NULL	Accept the request even if there is nothing in the request (i.e., create a queue entry to be later modified). No error message is given if there are no files in the request.	General
/OKNONE	Do not output message if no files match the wildcard construction. This is assumed at KJOB time.	File Control
/PHYSICAL	Suppress logical device name assignments for the device specified.	File Control
/PRIORITY:n	Assign the specified external priority (n=0 to 62) to the request. The larger the number, the greater priority the job has. The default is 10 if no switch is given and 20 if the switch is specified without a value.	General
/PROTECT:nnn	Assign the protection nnn (octal) to the job. If the switch, or the value of the switch, is omitted, the standard protection is assumed.	General
/PUNCH:026	Punch the files in 026 Hollerith code. If a /PUNCH: switch is not given the files are punched according to the data mode specified in the file.	File Control
/PUNCH:ASCII	Punch the files in ASCII card code. If a /PUNCH: switch is not given, the files are punched according to the data mode specified in the file.	File Control
/PUNCH:BINARY	Punch the files in binary card code. If a /PUNCH: switch is not given, the files are punched according to the data mode specified in the file.	File Control
/PUNCH:D029	Punch the files in DEC029 card code. If a /PUNCH: switch is not given, the files are punched according to the data mode specified in the file.	File Control
/PUNCH:IMAGE	Punch the files in image card code. If a /PUNCH: switch is not given, the files are punched according to the data mode specified in the file.	File Control
/REMOVE	Remove the file from the queue. This switch is valid only with the /MODIFY and can be used to remove a previously submitted file as long as CDPSPL has not started processing the request.	File Control

Command Format (cont)

Switch	Explanation	Category
/SEQ:n	Specify a sequence number to help in identifying a request to be modified or deleted.	General
/SINCE:t	Queue only the files with creation dates after the specified time. t is in the form dd-mmm-yy hhmm.	General
/START:n	Begin on the nth line of the file. If the switch, or the value of the switch, is omitted, the spooler starts with the first line.	File Control
/STRS	Search for the file on all file structures in the search list and take each occurrence. The default is to take just the first occurrence.	File Control
/UNPRESERVED	Output the files only if they are not preserved (i.e., the first digit is 0). This switch avoids redundant punching.	General

Characteristics

The CPUNCH command:

Leaves the terminal in monitor mode.
Runs the QUEUE program, thereby destroying the user's core image.
Depends on FTQCOM which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.CPUNCH SYSTAT.MAC/PUNCH:ASCII↲
```

Punch the file SYSTAT.MAC in ASCII format.

```
.CPUNCH SYSTAT.REL/PUNCH:BINARY/AFTER:1700↲
```

Punch the file SYSTAT.REL in binary format, but do not begin punching it until after 5:00 pm.

CREATE command [1]

Function

The CREATE command runs LINED (Line Editor for disk) and opens a new file on disk for creation. Refer to the LINED writeup in the DECsystem-10 Software Notebooks.

Command Format

CREATE file.ext

file.ext = any legal filename and filename extension. The filename is required; the filename extension is optional.

Characteristics

The CREATE command:

Places the terminal in user mode.
Runs the LINED program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Example

```
.CREATE TEST1.F4)

*
```

[1] This command runs the COMPIL program, which interprets the commands before running LINED.

CREF command

Function

The CREF command runs CREF and lists on the line printer (LPT) any cross-reference listing files generated by previous COMPILE, LOAD, EXECUTE, and DEBUG commands, using the /CREF switch, since the job was initiated. The file containing the names of these CREF-listing files is then deleted so that subsequent CREF commands will not list them again. The output goes either to LPT immediately or to the disk to be spooled later to LPT. When the logical device name LPT is assigned to a device other than the line printer, the CREF files are stored on that device with the same filename and the extension .LST.

Command Format

CREF

Characteristics

The CREF command:

Leaves the terminal in monitor mode.
Runs the CREF program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Example

`.COMPILE/CREF@PROMAC`	Compile the files contained in the command file PROMAC and produce CREF input compatible cross-reference listing files on the disk.
`.CREF`	Process and list the cross-reference listing files produced by the COMPILE command.
`.LOAD/C/MAP:NAME@CONALL`	Compile and load the files contained in the command file CONALL. Produce a loader map with the filename NAME and CREF input compatible cross-reference listing files on the disk.
`.ASSIGN MTA1 LPT`	Assign the logical name LPT to MTA1.
`.CREF`	Store the CREF files on MTA1 to be listed at a later time.

CSTART command
CCONTINUE command

Function

The CSTART and CCONTINUE commands are identical to the START and CONTINUE commands, respectively, except that the terminal is left in the monitor mode.

Command Format

CSTART adr
CCONTINUE

adr = the address at which execution is to begin if other than the location specified within the file (.JBSA). If adr is not specified, the starting address comes from .JBSA. An explicit starting address of 0 may be specified for adr.
To use:

1. Begin the program with the terminal in user mode.
2. Type control information to the program, then type ↑C to halt the job with the terminal in monitor mode.
3. Type CCONTINUE to allow job to continue running and leave the terminal in monitor mode.
4. Additional monitor commands can now be entered from the terminal.

Characteristics

The CSTART and CCONTINUE commands:

Leave the terminal in monitor mode.
Require core.
Depend on FTATTACH which is normally absent in the DECsystem-1040.

Restrictions

These commands should not be used when the user program (which is continuing to run) is also requesting input from the terminal. These commands are not available to Batch users.

Associated Messages

Refer to Chapter 4.

Example

```
.TYPE LOOP.F4↵
   ACCEPT 1Ø,I
1Ø FORMAT (I)
   DO 2Ø J=1,I
2Ø CONTINUE
   END
```

The user types his source program.

```
.EXECUTE LOOP↵

FORTRAN:LOOP.F4
LOADING
LOOP 2K CORE
EXECUTION
```

The user compiles, loads, and executes the program.

```
1ØØØØØØ
```

The user indicates that the program should loop 1000000 times.

```
↑C
↑C
```

The user stops the program.

```
.CCONT↵
```

The user continues the program but keeps the terminal in monitor mode.

```
.TIME↵

Ø.95
19.62
KILO-CORE-SEC=133
```

The user times the program.

```
.TIME↵
1.45
23.Ø7
KILO-CORE-SEC=157
```

The program is still running.

```
.SYSTAT↵
PLEASE TYPE ↑C FIRST
```

SYSTAT would cause the program to terminate.

```
.TIME↵
Ø.ØØ
23.Ø7
KILO-CORE-SEC=157
```

The program appears to have finished because the runtime has stopped incrementing. The program will not output until the CONT command is given.

```
.↑C
```

Return to the monitor.

```
.CONT↵
```

Continue the program so it can complete its typing.

```
CPU TIME:5.55 ELAPSED TIME:1:5.73
NO EXECUTION ERRORS DETECTED

EXIT

.
```

D (deposit) command

Function

The D command deposits information in the user's core area (high or low segment). When debugging a sharable program with the D command, the SAVE command should be used rather than the SSAVE command (refer to Appendix D).

Command Format

D lh rh adr

lh = the octal value to be deposited in the left half of the location. This argument is required.

rh = the octal value to be deposited in the right half of the location. This argument is required.

adr = the address of the location into which the information is to be deposited. This argument is optional.

If adr is omitted, the data is deposited in the location following the last D adr or in the location of the last E adr (whichever was last).

Characteristics

The D command:

Leaves the terminal in monitor mode.
Requires core.

Associated Messages

Refer to Chapter 4.

Example

```
.D 266000 2616 141
```

Deposit in location 141.

```
.E 140
000140/ 047000 000000     .D 47000 1
```

Examine location 140.
Since adr is omitted, the deposit is in the location of the last E command.

```
.E
000140/ 047000 000001     .
```

The examine is of the location of the previous D command.

DAYTIME command

Function

The DAYTIME command types the date followed by the time of day. The date and time are typed in the following format:

dd-mmm-yy hh:mm:ss

where

dd	= day
mmm	= month
yy	= year
hh	= hours
mm	= minutes
ss	= seconds to nearest hundredth.

Command Format

DAYTIME

Characteristics

The DAYTIME command

Leaves the terminal in monitor mode.
Does not require LOGIN.
Does not destroy the user's core area.

Example

```
.DAY↵
11-SEP-70 22:36:34

.DA↵
15-DEC-71 :47:02

.
```

DCORE command

Function

The DCORE command causes the DAEMON program to write a core-image file of the user's core area that includes all accumulators and all relevant job tables. The jog can continue to run; i.e., the DCORE command does not destroy the user's core area. The file produced may be later processed by the DUMP program, if the user so desires.

The DAEMON-written file consists of four categories: JOB, CONFIGURATION, DDB, and CORE. Each category has a two-word header, the first word contains the category number and the second word contains the number of data words in the category. The categories are as follows:

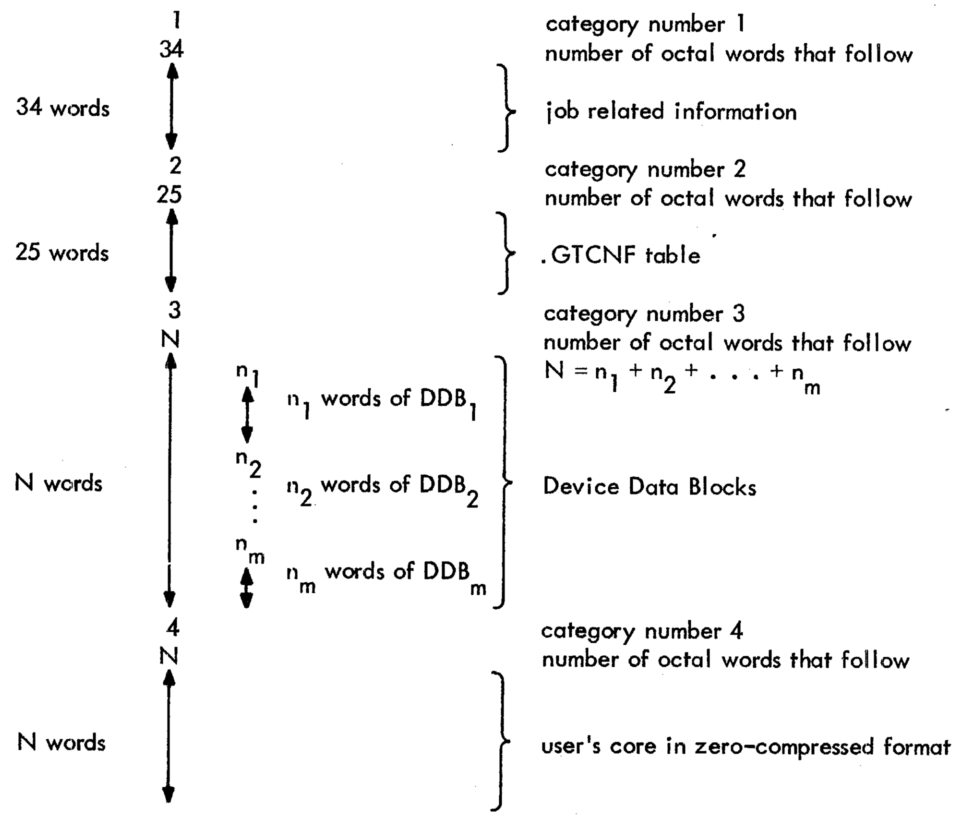
Mnemonic	Category Number	Description
JOB	1	Job related information.
CONFIGURATION	2	The Configuration Table (.GTCNF) from the GETTAB UUO.
DDB	3	The device data blocks (DDB) assigned to this job.
CORE	4	The user's core area, both low and high segments, in zero-compressed format (refer to Appendix D).

The third word of each category begins the data for that category. DAEMON treats each category as a file and the addresses within the category start at zero. The user cannot examine the category header nor can he read past the end of one category into the next category.

DCORE command (Cont)

Function (cont)

The DAEMON-written file appears as follows:



Category 1 presently contains the following information, but may expand as more GETTAB entries appear.

Word 1	Version of DAEMON that wrote the file. DATE the file was written in standard system format. TIME in milliseconds that the file was written. JOB NUMBER in left half, HIGH SEG number (or 0) in right half.

(continued on next page)

Function (cont)

Word 5	LH is reserved, TTY LINE NUMBER in right half.
	.GTSTS (job status word) for job.
	.GTSTS for high segment.
	.GTPPN (project-programmer number) for job.
	.GTPPN for high segment.
	.GTPRG (user program name) for job.
	.GTPRG for high segment.
	.GTTIM (total time used) for job.
	.GTKCT (kilo-core-ticks) for job.
	.GTPRV (privilege bits) for job.
	.GTSWP (swapping parameters) for job.
Word 20	.GTSWP for high segment.
	.GTRCT (disk blocks read) for job.
	.GTWCT (disk blocks written) for job.
	.GTTDB (time of day of last disk allocation, number of disk blocks allocated) for job.
	.GTDEV (device or file structure name) for high segment.
	.GTNM1 (first half of user's name) for job.
	.GTNM2 (last half of user's name) for job.
	.GTCNO (charge number) for job.
	.GTTMP (TMPCOR pointers) for job.
	.GTWCH (WATCH bits) for job.
	.GTSPL (spooling control bits) for job.
	.GTRTD (real-time status word) for job.
Word 34	.GTLIM (time limit in jiffies) for job.

Category 2 presently contains the following information, but may expand if more .GTCNF entries are added.

%CNFG0 ↓ %CNFG4	Name of system in ASCIZ.
%CNDT0	Date of system in ASCIZ.
%CNDT1	
%CNTAP	Name of system device in SIXBIT.
%CNTIM	Time of day in jiffies.
%CNDAT	Today's date.
%CNSIZ	Highest location in monitor +1.
%CNOPR	Name of OPR TTY console.

(continued on next page)

DCORE command (Cont)

Function (cont)

%CNDEV	LH = beginning of DDB chain.
%CNSGT	LH = -# of high segments, RH = +# of jobs.
%CNTWR	Non zero if system has two register hardware and software.
%CNSTS	LH = feature switches, RH = current state of switches.
%CNSER	Serial number of processor.
%CNNSM	# of nanoseconds per memory cycle.
%CNPTY	PTY parameters for Batch.
%CNFRE	AOBJN word to use bit map in monitor.
%CNLOC	LH = 0, RH = address of free 4-word core block area.
%CNSTB	Link to STB chain for remote Batch.

Category 3 contains the device data blocks currently in use for this job. Each DDB is preceded by a word containing the length of the DDB.

Category 4 is a compressed core image of both the high and low segments, i.e., it contains only nonzero words.

Command Format

DCORE dev:name.ext [proj,prog]

dev: = a disk-like device on which the core-image file is to be written. If omitted, DSK is assumed.

name.ext = the name of the file to be written. The default filename is nnnDAE, where nnn is the job number in decimal, and the default extension is .TMP. If a filename is specified, the default extension is .DAE.

[proj,prog] = the disk area other than that of the user. If omitted, the user's disk area (the number under which he is logged in) is assumed.

Characteristics

The DCORE command:

Leaves the terminal in monitor mode.
Runs the DAEMON program.
Can continue after command.
Depends on FTDAEM which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.DCORE)
```

The core image file is written on the user's area of the disk with the name nnnDAE.TMP where nnn is the user's job number.

```
.DCORE DSKB:FILEC)
```

The core image file is written in the user's area on DSKB with name FILEC.DAE.

```
.
```

DDT command

Function

The DDT command copies the saved program counter value from .JBPC into .JBOPC and starts the program at an alternate entry point specified in .JBDDT (beginning address of DDT as set by Linking Loader). DDT contains commands to allow the user to start or resume at any desired address. Refer to DECsystem-10 Monitor Calls for a description of the job data area locations.

If the job was executing a UUO when interrupted (i.e., it was in exec mode and not in TTY input wait or SLEEP mode), the monitor sets a status bit (UTRP) and continues the job at the location at which it was interrupted. When the UUO processing is completed, the monitor clears the status bit, sets .JBOPC to the address following the UUO, and then traps to the DDT address found in .JBDDT. If the job is in exec mode and in TTY input wait or SLEEP mode, the trap to the DDT address occurs immediately and .JBOPC contains the address of the UUO. If the job is in user mode, the trap also occurs immediately. Therefore, it is always possible to continue the interrupted program after trapping to DDT by executing a JRSTF @ .JBOPC.

For additional information on the DDT program, refer to the DDT Programmer's Reference Manual in the DECsystem-10 Software Notebooks.

Command Format

DDT

Characteristics

The DDT command:

Places the terminal in user mode.
Requires core.
Requires the user to have a job number.

Associated Messages

Refer to Chapter 4.

Examples

```
.TYPE LOOP.MAC↵                         Type an undebugged program.
LOOP: JRST LOOP
      END LOOP

.DEBUG LOOP                             Assemble and load the program with DDT.
MACRO:.MAIN
LOADING

LOOP 3K CORE
DDT EXECUTION

↑Z
.SAVE↵                                  Save the program.
JOB SAVED

.START                                  Start the program.

↑C
↑C                                      Stop it.
.DDT↵                                   Enter DDT.

LOOP/JRST LOOP CALLI12                  Fix the program.
      JRSTF @.JBOPC$X
EXIT

.
```

DEASSIGN command

Function

The DEASSIGN command returns one or more devices currently ASSIGNed to the user's job back to the monitor pool of available devices and clears any logical names. Restricted devices are returned to the restricted pool, and unrestricted devices to the unrestricted pool. Note that an INITed device is not returned to the monitor pool unless a RELEASE UUO is done, only the logical name is cleared. Therefore, this command is provided for programs that are not termimating or programs that are being debugged. The command, applied to DECtapes, clears the copy of the directly currently in core, forcing the next directory reference to read a new copy from the tape. (Refer to DECsystem-10 Monitor Calls for further details.)

Command Format

DEASSIGN dev

dev = either the logical or physical device name. This argument is optional. If it is not specified, all devices assigned to the user's job, except the job's controlling terminal, are deassigned, and the logical name of the controlling terminal is cleared.

Characteristics

The DEASSIGN command:

Leaves the terminal in monitor mode.

Associated Messages

Refer to Chapter 4.

Examples

```
.DEASSIGN LPT:)
.
```

The line printer is returned to the monitor's pool of available resources.

```
.DEASSIGN)
.
```

All devices assigned to the job are returned.

DEBUG command [1]

Function

The DEBUG command translates the specified source files if necessary (function of the COMPILE command), loads the REL files generated (function of the LOAD command), and prepares for debugging. DDT (the Dynamic Debugging Technique program) is loaded first, followed by the user's program with local symbols. Upon completion of loading, control is transferred to the DDT program. This program is used to check programs section by section by allowing the user to examine and modify the contents of any location either before execution or during breakpoints. Refer to the DDT documentation for a description of the DDT commands.

Each time a COMPILE, LOAD, EXECUTE, or DEBUG command is executed, the command with its arguments is remembered in a temporary file on disk, or in core if the monitor has the TMPCOR feature. Therefore, the last filename used can be recalled for the next command without specifying the arguments again (refer to Paragraph 1.5).

The DEBUG command accepts several command constructions: the @ construction (indirect commands), the + construction, the = construction, and the <> construction. Refer to Paragraph 1.5 for a complete description of each of these constructions.

Command Format

DEBUG list

list = a single file specification, or a string of file specifications separated by commas. A file specification consists of a device name, a filename with or without an extension, and a directory name.

The following switches can be used to modify the command string. These switches can be temporary or permanent (refer to Paragraph 1.5.5).

/ALGOL	Compile the file with ALGOL. Assumed for files with the extension of .ALG.
/BLISS[2]	Compile the file with BLISS. Assumed for files with the extension of .BLI.
/COBOL	Compile the file with COBOL. Assumed for files with the extension of .CBL.

(continued on next page)

[1] This command runs the COMPIL program, which interprets the command before running the appropriate processor, the LOADER, and DDT.

[2] BLISS will be recognized as a processor only if the appropriate assembly switch is set. However, this assembly switch setting is not supported.

DEBUG command (Cont)

Command Format (cont)

/COMPILE	Force a compilation on this file even though a binary file exists with a newer date and time than the source file. This switch is used to obtain an extra compilation (e.g., in order to obtain a listing of the compilation) since normally compilation is not performed if the binary file is newer than the source file.
/CREF	Produce a cross-reference listing file on the disk for each file compiled for later processing by the CREF program. These files have the filename of the source file and the extension of .CRF. The file can then be listed with the CREF command. However, with COBOL files, the cross-reference listing is always appended to the listing file. No additional command need be given to obtain the listing.
/FORTRAN	Compile the file with FORTRAN. Assumed for files with the extension of .F4 and all files with non-recognizable processor extensions (if FORTRAN is the standard processor).
/FUDGE	Create a disk file containing the names of the .REL files produced by the command string. When the FUDGE command is given, PIP reads this file in order to generate a library REL file. Arguments to this switch are: /FUDGE:dev:file.ext[proj,prog] dev: - the device on which to write the file. DSK: is assumed. file.ext - the name of the library file. The filename is required. If the extension is omitted, it is assumed to be .REL. [proj,prog] - the directory in which to place the file. The user's directory is assumed if none is given. This switch is permanent in the sense that it pertains to all REL files generated by the command string.
/LIBRARY	Load the files in library search mode. This mode causes a program file in a special library file to be loaded only if one or more of its declared entry symbols satisfies an undefined global request in the source file. The system libraries are always searched. Refer to the LOADER documentation.

(continued on next page)

Command Format (cont)

/LIST	Generate a disk listing file, for each file compiled, with the filename of the source file and the extension.LST. These files can be listed later with the LIST command. Unless this switch is specified, listing files are not generated except in COBOL; COBOL listings are always generated.
/LMAP	Produce a loader map during the loading process (same action as /MAP) containing the local symbols.
/MACRO	Assemble the file with MACRO. Assumed for files with extensions of .MAC.
/MAP	Produce a loader map during the loading process. When this switch is encountered, a loader map is requested from the loader. After the library search of the system libraries, the map is written in the user's disk area with either the filename specified by the user (e.g., /MAP:file) or the default filename MAP.MAP. This switch is an exception to the permanent switch rule in that it causes only one map to be produced even though it may appear as a permanent switch.
/MACX11[1]	Assemble the file with MACX11. Assumed for files with the extension .P11.
/NOCOMPILE	Complement the /COMPILE switch by not forcing a compilation on a source file whose date is not as recent as the date on the binary file. Note that this switch is not the same as the /REL switch, which turns off all compilation, even if the source file is newer than the REL file. /NOCOMPILE is the default action.
/NOLIST	Do not generate listing files. This is the default action except for COBOL files; COBOL listings are always generated.
/NOSEARCH	Loads all routines of the file whether the routines are referenced or not. Since this is the default action, this switch is used only to turn off library search mode (/LIBRARY). This switch is not equivalent to the /P switch of the LOADER, which does not search any libraries. The /NOSEARCH default is to search the system libraries.

(continued on next page)

[1]MACX11 (the PDP-11 assembler for the PDP-10) will be recognized as a processor only if the appropriate assembly switch is set. However, this assembly switch setting is not supported.

DEBUG command (Cont)

Command Format (cont)

/REL	Use the existing REL files although newer source files may be present.
/SNOBOL[1]	Compile the file with SNOBOL. Assumed for files with extensions of .SNO.

Characteristics

The DEBUG command:

Places the terminal in user mode.
Runs the appropriate processor, the LOADER, and DDT.

Associated Messages

Refer to Chapter 4.

Examples

```
.DEBUG/L FILEA,FILEB,FILEC/N,FILED)     Generate listings for FILEA, FILEB, and
                                        FILED
```

```
.DEBUG TEST)
MACRO: TEST
LOADING

LOADER 2K CORE
DDT EXECUTION

./      BLT 15,Ø(16)
```

[1]SNOBOL will be recognized as a processor only if the appropriate assembly switch is set. However, this assembly switch setting is not supported.

DELETE command [1]

Function

The DELETE command deletes one or more files from disk or DECtape.

Command Format

DELETE list

list = a single file specification or a string of file specifications separated by commas. The full wildcard construction (* and ?) can be used.

If a device or file structure name is specified, it remains in effect until changed or until the end of command string is reached.

Characteristics

The DELETE command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.DEL *.MAC )
FILES DELETED:
T1.MAC
T2.MAC
T3.MAC
14 BLOCKS FREED

.DEL TEST1.MAC )
FILES DELETED:
TEST1.MAC
3 BLOCKS FREED

.
```

[1] This command runs the COMPIL program, which interprets the command before running PIP.

DELETE command (Cont)

Examples (cont)

```
.DEL TEST??.F4
FILES DELETED:
TEST.F4
TESTS.F4
TEST03.F4
TESTZ.F4
23 BLOCKS FREED
```

DETACH command

Function

The DETACH command disconnects the terminal from the user's job without affecting the status of the job. The user terminal is now free to control another job, either by initiating a new job or attaching to a currently running detached job.

Command Format

DETACH

Characteristics

The DETACH command:

Detaches the terminal.
Depends on FTATTACH which is normally absent in the DECsystem-1040.

Restrictions

This command is not available to Batch users.

Associated Messages

Refer to Chapter 4.

Example

```
.DETACH ↵
FROM JOB 1

.
```

DIRECT command

Function

The DIRECT command lists the directory entries specified by the argument list. The standard output consists of the following columns: filename, filename extension, length in blocks written, protection, creation date, version number, structure name, and directory name.

Command Format

DIRECT output specification = list of input specifications

list = A single file specification, or a string of files specifications separated by commas or plus signs. The devices used on input can be DSK:, DTA:, MTA:, and TMP: (TMPCOR). If the device is a magnetic tape, the tape is rewound before and after the listing operation and analyzed to determine if it is a FAILSAFE or BACKUP tape. The default input specification is DSK:*.*, and the user's directories in all file structures defined by the job's search list are listed. Generally, a device name, an extension, or a directory name that precedes the filename becomes the default for all succeeding fules in the list. The full wildcard construction can be used.

output specification = This argument (and the equal sign) is optional. If the entire output specification is omitted, the default is TTY:. If an output filename is given, the default device is DSK:. If an output filename is not given, and one is needed, the filename is generated from the time of day as hhmmss. The default output extension is .DIR. The wildcard construction cannot be used in the output specification.

The following switches may be used in the command string. Switches that precede the filename become the default for all succeeding files. Switches can be abbreviated as long as the abbreviation is unique.

/ACCESS:n	Update the access date to the current date for any file of n blocks or less accessed by the DIRECT program. n is the written length unless the ALLOC switch is used and is a decimal number. If /ACCESS is omitted, the date is not changed. If /ACCESS is specified but :n is omitted, n=5 is assumed.
/ALLOC	List the allocated length of the file instead of the written length. The allocated length is used by LOGOUT in checking quotas. (Disk and magnetic tape only.)

(continued on next page)

Command Format (cont)

/CHECKSUM	Compute and print an 18-bit checksum for each file. This checksum is computed by rotating the result left one bit before adding each word. (Disk and magnetic tape only.)
/DENSITY:n	Use the specified density when reading a magnetic tape. N is 200, 556, or 800 bpi. The default is installation dependent and is modified by the SET DENSITY command.
/DETAIL	Print all nonzero words in the LOOKUP block. The protection and data mode are also listed, even if they are zero. The author is not listed if it is the same as the owner of the directory. (Disk and magnetic tape only.)
/FAST	List short form of directory (i.e., filename, extension, structure name, and directory name). Equivalent to /F.
/HELP	Help text which indicates some of the switches available and how to use them. Equivalent to /H.
/HELP:S	List all switches (S) without their explanations. An asterisk prefixes those switches which have a single-letter abbreviation.
/LIST	List the output on device LPT:. Equivalent to /L.
/MARKS	Indicate each tape mark and UFD when reading a magnetic tape.
/OKNONE	Suppress the error message if no files match the wildcard construction.
/PARITY:ODD /PARITY:EVEN	Specify the parity to be used when reading a magnetic tape. The default is ODD.
/PHYSICAL	Ignore logical names used for device names.
/PROTECTION:nnn	Give the output file the protection nnn (octal).
/RUN:file spec	Run the specified program when this command is finished.
/RUNOFFSET:n	Run the program specified with /RUN with an offset of n. If the switch is omitted, the default is 0; if the switch is given without a value, the default is 1.
/SLOW	Output a full listing that includes the filename, extension, length in blocks written, protection, creation time, access date, structure name, and directory name. Equivalent to /S. (Disk and magnetic tape only.)

(continued on next page)

DIRECT command (Cont)

Command Format (cont)

/SORT	List the file structure name and directory name on each line instead of only on the first line in which they change. Multiple spaces are output instead of TABs. This switch is used to prepare a file to be sorted by the SORT program.
/SUMMARY	Output only the summary line which indicates the total number of blocks and files. Note a /F /SUMMARY lists a /F listing followed by the summary.
/TITLES	Cause a heading to be output on each page consisting of a label for each column, date, time, and page number. Standard output to the line printer has this heading.
/UNITS	List the name of the actual disk unit instead of the file structure name.
/WIDTH:n	Output several entries on a single line to make the output n columns wide. For example, if /F is specified for output to the scope, four filenames appear per line. The default for n is 64 columns.
/WORDS	Output the length of the file in words instead of blocks.

Characteristics

The DIRECT command:

Leaves the terminal in monitor mode.
Runs the DIRECT program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Examples

`.DIR DTA3:` ↲	Lists all files on DTA3.
`.DIR *.MAC` ↲	Lists all files with MAC filename extension in all file structures in the job search list.
`.DIR TEST.F4[27,60]` ↲	Lists the directory entry for file TEST.F4 in user area 27, 60.

(continued on next page)

Examples (cont)

```
.DIRECT)

SAMPL   CTL     1       <155>   4-MAY-71        DSKC:   [27,235]
PIP     DAE     0       <055>   25-FEB-72
G16DAE  TMP     22      <055>   25-FEB-72
WEIRD           4       <055>   25-FEB-72
WEIRD   SAV     21      <055>   25-FEB-72       34(70)

  TOTAL OF 48 BLOCKS IN 5 FILES ON DSKC: [27,235]

.DIRECT/ALLOC

SAMPL   CTL     3       <155>    4-MAY-71       DSKC:   [27,235]
PIP     DAE     2       <055>   25-FEB-72
G16DAE  TMP     24      <055>   25-FEB-72
WEIRD           6       <055>   25-FEB-72
WEIRD   SAV     23      <055>   25-FEB-72       34(70)

  TOTAL OF 58 BLOCKS IN 5 FILES ON DSKC: [27,235]

.DIRECT/DETAIL)

DSKC0:SAMPL.CTL[27,235]
ACCESS DATE: 5-JAN-72
CREATION TIME, DATE: 16:51  4-MAY-71
ACCESS PROTECTION: 155
MODE: 14
WORDS WRITTEN: 54.
ESTIMATED LENGTH: 5.
BLOCKS ALLOCATED: 3.
DATA BLOCK IN DIRECTORY: 303.

DSKC0:PIP.DAE[27,235]
ACCESS DATE: 25-FEB-72
CREATION TIME, DATE: 13:47 25-FEB-72
ACCESS PROTECTION: 055
MODE: 17
BLOCKS ALLOCATED: 2.
AUTHOR: 1,2
DATA BLOCK IN DIRECTORY: 303.
```

(continued on next page)

DIRECT command (Cont)

Examples (cont)

```
DSKC0:G16DAE.TMP[27,235]
ACCESS DATE: 25-FEB-72
CREATION TIME, DATE: 14:10 25-FEB-72
ACCESS PROTECTION: 055
MODE: 17
WORDS WRITTEN: 2816.
BLOCKS ALLOCATED: 24.
AUTHOR:1,2
DATA BLOCK IN DIRECTORY: 303.

DSKC0:WEIRD.[27,235]
ACCESS DATE: 25-FEB-72
CREATION TIME, DATE: 14:08 25-FEB-72
ACCESS PROTECTION: 055
MODE: 1
WORDS WRITTEN: 471.
BLOCKS ALLOCATED: 6.
DATA BLOCK IN DIRECTORY: 303.

DSKC0:WEIRD.SAV[27,235]
ACCESS DATE: 25-FEB-72
CREATION TIME, DATE: 14:09 25-FEB-72
ACCESS PROTECTION: 055
MODE: 10
WORDS WRITTEN: 2566.
VERSION:34(70)
BLOCKS ALLOCATED: 23.
DATA BLOCK IN DIRECTORY: 303.

  TOTAL OF 48 BLOCKS IN 5 FILES ON DSKC: [27,235]
```

```
.DIRECT[40,*]
```

Lists the directory entries for user with project number 40 and the user's programmer number.

DISMOUNT command

Function

The DISMOUNT command allows a user to return devices to the monitor pool of available resources and to remove a file structure from his search list. Restricted devices are returned to the restricted pool and unrestricted devices to the unrestricted pool. The command applied to non-file structures is identical to the DEASSIGN command if the user waits for completion of the operator action. If the user does not wait for completion (e.g., he types a control-C after the message OPERATOR NOTIFIED), the device is not deassigned, but the request to the operator is still queued for the purpose of removing the media. The user must then issue the DEASSIGN command to release the device. This command applied to file structures enforces logged-out quotas (if necessary), allows physical removal of disk packs (if there are no other users of the pack), and removes the file structure name from the job's search list.

The UMOUNT program runs privileged in the user's core area when the DISMOUNT command is typed. This program scans the user's command string, checks its validity, and performs as much of the requested action as possible. The UMOUNT program can complete all actions requested by the DISMOUNT command except for the action of physically removing packs, tapes, or cards. When operator action is required, the UMOUNT program writes a command file on 3,3 disk area and notifies the OMOUNT program (running on the operator's terminal) to perform the action. When the operator action has been completed, OMOUNT deletes the command file and notifies UMOUNT (if UMOUNT is waiting) to inform the user of completion.

Command Format

DISMOUNT dev: switches

dev: = any previously ASSIGNed or MOUNTed device or file structure name. This argument is required.

switches = the following switches are optional and only enough characters to make the switch unique are required.

/CHECK	Check and list pending requests.
/HELP	Type this list.
/PAUSE	Notify the user before requesting operator action. The user can then abort the command if desired.

(continued on next page)

DISMOUNT command (Cont)

Command Format (cont)

/REMOVE	Notify operator to physically remove disk packs, tape, or cards. A file structure is removed from the system only if no other users are using it. A request to remove the pack is queued to the operator and the message WAITING . . . is typed to the user. If the user does not want to wait for confirmation of the operator action, he may type control-C. This switch must be specified to notify the operator to remove the pack, even if no other jobs are using it.

Characteristics

The DISMOUNT command:

Places the terminal in user mode.
Runs the UMOUNT program, thereby destroying the user's core image.
Depends on FTCCLX and FTMOUN which are normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.DISMOUNT DSKA:)
DSKA DISMOUNTED
```
The user dismounts the file structure DSKA. This does not require an operator action.

```
.DISMOUNT DTA4:/R)
OPERATOR NOTIFIED
```
The user asks the operator to deassign DTA4 and remove the tape.

```
WAITING...
```
The command is waiting for completion of the operator action.

```
↑C
```
The user does not wish to wait for confirmation of removal.

```
.DISMOUNT/CHECK)
NONE PENDING
0,COMMANDS IN QUEUE
.
```
The user checks for completion and determines that his request is finished.

DSK command

Function

The DSK command types disk usage for the combined structures of the job, since the last DSK command, followed by the total disk usage since the job was initialized (logged in). Disk usage is typed in the following format:

```
RD, WT=I,J
RD,WT=M,N
```

where I and J are the incremental number of 128-word blocks read and written since the last DSK command, and M and N are the total number of 128-word blocks read and written since the job was initialized.

NOTE

I and J are kept modulo 4096. If automatic READ or WRITE print outs have been enabled using the SET WATCH command, I and J are usually zero, since the SET WATCH output also resets these values.

Command Format

DSK job

job = the job number of the job for which the disk usage is desired. This argument is optional.

If job is omitted, the job to which theterminal is attached is assumed.

If job is supplied (whether the job of this user or another user) the incremental quantities are not reset to zero.

Characteristics

The DSK command:

Leaves the terminal in monitor mode.

Associated Messages

Refer to Chapter 4.

DSK command (Cont)

Example

```
.DSK↵
RD,WT=12,Ø
RD,WT=475,243

.
```

DUMP command

Function

The DUMP command calls the DAEMON program to write a core image file (function of the DCORE command) and then invokes the DUMP program to analyze the file written and to provide printable output. The core image file is named nnnDAE.TMP where nnn is the user's job number. This file is described in detail in the DCORE command description.

Command Formats

1. DUMP /command /command /command ...
2. DUMP @ dev:file.ext [proj,prog]
3. DUMP

The commands that appear in the DUMP command string are passed to the DUMP program and therefore are described in the DUMP program description. A DUMP command using a command file can also specify these commands. A DUMP command without any arguments prints a short dump of the user's core area via the command file QUIKDM.CCL which resides on device SYS:.

Characteristics

The DUMP command:

Leaves the terminal in monitor mode.
Runs the DAEMON program.
Depends on FTDAEM which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.DUMP/OUT:TTY:/MODE:ASCII,SIXBIT/WIDTH:7,10/JUST:L,R-↲
7RIGHTMA:26/D[3000 & 3004
```

This command string writes a core image file named nnnDAE.TMP and invokes the DUMP program to perform the output. The output goes to the terminal and the modes used on output are ASCII and SIXBIT. The ASCII field is 7 characters long, left justified and the SIXBIT field is 10 characters long, right justified. The right margin of the output in 26 characters. The dump consists of the contents of word 3000 to word 3004. The hyphen is used to continue the command string onto the next line.

DUMP program

Function

The DUMP program provides printable dumps of arbitrary data files in modes and forms specified by the user. The DUMP program accepts any data file as input and produces an ASCII file suitable for listing by PIP, the output spoolers, or other listing programs. For example, the DUMP program takes core image files prepared by the DAEMON program or SAVEd files produced by the monitor. For a description of the DAEMON-written file, refer to the DCORE command.

Command Formats

```
1.  R DUMP↲
    /command↲

2.  R DUMP↲
    /@ dev:file.ext [proj,prog]
```

NOTE
DUMP indicates its readiness by typing a slash (/) instead of an asterisk.

The commands with their arguments are as follows. Lines can be continued by typing a hyphen followed by a carriage return.

Command	Argument	Meaning
ADDRESS	ON or OFF	Specifies if the address is to be dumped along with its contents. The default is ON.
ALL		Dumps the entire file. If the file is a DAEMON core image file, the entire category is dumped.
APPEND		Appends the output to the output file. The existing output file is not overwritten. This command is the default; its complement is SUPERSEDE.
AUTOFORMAT	ON or OFF	Attempts to format output with line feeds, form feeds, and titles, if ON. If OFF, the user is responsible for all formatting. The default is ON.
CATEGORY	mnemonic for name of category. Can be JOB, CONFIGURATION, DDB, or CORE.	Selects the category of the DAEMON dump file to be used. Addressing begins with 0 at the beginning of each category. The default category is CORE. If the input file is not a DAEMON file, this switch has no effect.

(continued on next page)

Command Formats (cont)

Command	Argument	Meaning
CLOSE		Closes the output file. After this command is given, another OUT command should be given before the next command which does any output.
DUMP	dump descriptor, dump descriptor,...	Dumps the specified bytes in the current modes.
EJECT		Starts a new page in the output file.
EXIT		Closes all files and returns control to the monitor (↑Z has the same effect).
INPUT	<file descriptor>	Specifies the input file. The defaults are: DSK:nnnDAE.TMP where nnn is the job number; the user's directory. If the filename is specified, it determines the extension from the set .TMP, .DAE, .SHR, .SAV, .HGH, .LOW, .XPN, and .DMP in that order. If an extension is specified with no filename, the extension determines the filename.
IRADIX	decimal number	Specifies radix for numbers for input. This command uses decimal to compute the argument. The default is 10 for decimal. The argument must be numeric. If numeral is 0 or is missing, the input radix is set back to its default value.
JUSTIFY	LEFT, CENTER, or RIGHT	Specifies the justification of the output in the output field. If the output overflows the output field, the entire output appears; it is not truncated. This switch is used in a one-to-one relationship with the MODE and WIDTH commands. If there are more MODE commands, an argument of LEFT is used. If there are more JUSTIFY commands, they are ignored.
LEFTMARGIN	expression	Sets the left margin of the output file. The default is 0.
LINEPAGE	expression	Specifies the number of lines per output page. The default is 50.

(continued on next page)

DUMP program (Cont)

Command Formats (cont)

Command	Argument	Meaning
MODES	ASCII, DECIMAL, NULL, NUMERIC, OCTAL, RADIX50, SIXBIT, SOCTAL, or SYMBOLIC.	Selects the type of output. ASCII dumps the word as a single right justified character if bits 0-28 are zero or as 5 ASCII characters if bits 0-28 are non-zero. Non printing characters print as a space. DECIMAL dumps as a signed decimal number. NULL declares that nothing is to be dumped. NUMERIC dumps as a signed number in the current output radix. OCTAL dumps as half-words separated by a comma (default) and takes 13 positions. RADIX50 dumps in RADIX50. SIXBIT dumps as one SIXBIT character if bits 0-24 are zero, or 6 SIXBIT characters if bits 0-24 are nonzero. SOCTAL dumps as signed octal and suppresses leading zeroes. SYMBOLIC dumps as a symbolic instruction. Any mode specification can appear more than once in the command string. The output is in the same order as the MODE list.
NUMPAGE	expression	Specifies that pages are to be numbered. If expression is 0, page numbering is turned off. If expression is not 0, page numbering begins at page = <expression>. If command is omitted, numbering starts at the first page.
ORADIX	decimal number	Specifies radix for numbers for output. The default is 10 for decimal. If number is 0, the standard is used. The argument to this command is decimal and must not be an expression.
OUTPUT	<file descriptor>	Specifies the output file. The defaults are: LPT:, the filename of the input file; the extension .LSD; the user's directory.
RIGHTMARGIN	expression	Sets the right margin of the output file. A field may overflow the right margin if it will not fit between the left and right margins. If ADDRESS is ON, the new line will have an address typed. If a page overflow occurs, a title line may also be printed.
SUPERSEDE		Specifies that the output is to supersede an existing file of the same name, if there is one. The complement of this command is APPEND, which is the default.

(continued on next page)

Command Formats (cont)

Command	Argument	Meaning
SYFILE	<file descriptor>	Specifies the file to take symbols from if XTRACT command is specified. Defaults are: DSK:, the filename of the input file; one of the saved file extensions; the user's directory area.
TDUMP	dump descriptor, dump descriptor,...	Dumps specified bytes to both output file and TTY.
TITLE	<string of characters>	Specifies a title to be included in the subsequent page headings. If no argument is specified, titling is turned off. After this command, an EJECT command should be given to skip to a new page.
TYPE[1]	DAE	Specifies that the input file is separated by DAEMON.
TYPE	DAT	Specifies that the input file is a data file (i.e., no special format; therefore, no special processing is done).
TYPE	HGH	Specifies that the input file is in .HGH file format.
TYPE	LOW	Specifies that the input file is in .LOW file format.
TYPE	SAV	Specifies that the input file is in .SAV file format.
TYPE	SHR	Specifies that the input file is in .SHR file format.
TYPE	XPN	Specifies that the input file is in .XPN file format.
WIDTH	expression	Selects the width of each output mode (see the MODE and JUSTIFY commands). If a MODE command is specified without a corresponding WIDTH, the byte is dumped in exactly the number of positions required followed by 3 blanks. If a WIDTH command is specified, no free blanks are output. If a MODE specification overflows its WIDTH specification, the entire output is given without justification. If expression is omitted, a null list will be generated.
XTRACT		Uses the file specified in the last SYFILE command as a core image and extracts the symbol table.

[1]If TYPE is not specified, the extension of the input file is used to determine the type of file being produced. If the extension is not one recognized in the TYPE command, TYPE DAE is assumed.

DUMP program (Cont)

Command Formats (cont)

An expression is an octal or decimal number, a symbol, arithmetic operations using expressions (+, -, *, /, and ↑ grouped with parentheses), or contents operators ([, \, and @). A symbol is a string of SIXBIT characters, or program symbol, where program defines the program containing symbol.

A text string is a string of characters enclosed in single quotes. Special characters are represented by patterns of graphic characters. To override these special patterns, a double quote indicates that the next character is to be accepted as is, without including it as part of a special pattern. The following patterns represent non-graphic characters and are replaced in output strings by the characters represented unless a double quote appears.

<EL>	- end line, <CR-LF>
<VT>	- vertical tab
<FF>	- form feed
<AL>	- altmode
<HT>	- horizontal tab
↑<letter>	- control character
\<letter>	- lower case character

A byte descriptor is the description of the byte in the input file to be dumped. The format is:

WORDS <POS, SIZE>

where

WORDS = the address of the word desired.

POS = the position of the byte within the word. It specifies the bit number of the left-most bit in the byte.

SIZE = the number of bits in the byte. It may be any size and can cross word or block boundaries.

A dump descriptor has the form

<FROM byte-descriptor> & <TO byte descriptor>

signifying everything from the first byte descriptor to the second.

Characteristics

The R DUMP command:

Places the terminal in user mode.
Is used with disk monitors only.
Runs the DUMP program.

Associated Messages

Refer to Chapter 4.

E (examine) command

Function

The E command examines a core location in the user's area (high or low segment).

Command Format

E adr

adr is required the first time the E or D command is used. If adr is specified, the contents of the location are typed out in half-word octal mode.

If adr is not specified, the contents of the location following the previously specified E adr or the location of the previous D adr (whichever was last) are typed out.

Characteristics

The E command:

Leaves the terminal in monitor mode.
Requires core.

Associated Messages

Refer to Chapter 4.

Example

```
.E 140
000140/ 264000 002616    .E
000141/ 000000 000000    .E
000142/ 000000 000000    .
```

EDIT command [1]

Function

The EDIT command runs LINED (Line Editor for disk) and opens an already existing line sequence-numbered file on disk for editing. Refer to the LINED writeup in the DECsystem-10 Software Notebooks.

Command Format

EDIT file.ext

file.ext = a filename and filename extension of an existing file. This argument is optional if a CREATE or EDIT command has been given since the initialization of the job, because the arguments of the EDIT-class commands are remembered in temporary files on the disk or in core if the monitor has the TMPCOR feature.

Characteristics

The EDIT command:

Places the terminal in user mode.
Runs the LINED program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Example

```
.EDIT TEST.F4↵
*
```

[1] This command runs the COMPIL program, which interprets the command before running LINED.

EOF command [1]

Function

The EOF command writes an end of file mark on the specified magnetic tape. This command is equivalent to the following PIP command string:

```
MTAn: (MF) ←
```

Command Format

```
EOF MTAn:
```

Characteristics

The EOF command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.EOF MTA3:↲

.
```

[1] This command runs the COMPIL program, which interprets the command before running the PIP program.

EXECUTE command [1]

Function

The EXECUTE command translates the specific source files if necessary (function of COMPILE command), loads the REL files generated into a core image (function of LOAD command), and begins execution of the program. The assembler or compiler used is determined from the source file extensions or from switches in the command string (refer to the COMPILE command). If a REL file already exists with a newer date than that of the source file, compilation is not performed (unless requested explicitly via a switch).

This command is equivalent to a LOAD and START sequence of commands.

Each time a COMPILE, LOAD, EXECUTE, or DEBUG command is executed, the command with its arguments is remembered in a temporary file on disk, or in core if the monitor has the TMPCOR feature. Therefore, the last filename used can be recalled for the next command without specifying the arguments again (refer to Paragraph 1.5).

The EXECUTE command accepts several command constructions: the @ construction (indirect commands), the + construction, the = construction, and the < > construction. Refer to Paragraph 1.5 for a complete description of each of these constructions.

Command Format

EXECUTE list

list = a single file specification, or a string of file specifications separated by commas. A file specification consists of a device name, a filename with or without an extension, and a directory name.

The following switches can be used to modify the command string. These switches can be temporary or permanent switches (refer to Paragraph 1.5.5).

/ALGOL	Compile the file with ALGOL. Assumed for files with the extension of .ALG.
/BLISS[2]	Compile the file with BLISS. Assumed for files with the extension of .BLI.
/COBOL	Compile the file with COBOL. Assumed for files with the extension of .CBL.

(continued on next page)

[1] This command runs the COMPIL program, which interprets the command before running the appropriate processor and the LOADER.

[2] BLISS will be recognized as a processor only if the appropriate assembly switch is set. However, this assembly switch setting is not supported.

EXECUTE command (Cont)

Command Format (cont)

/COMPILE	Force a compilation on this file even though a binary file exists with a newer date and time than the source file. This switch is used to obtain an extra compilation (e.g., in order to obtain a listing of the compilation) since normally compilation is not performed if the binary file is newer than the source file.
/CREF	Produce a cross-reference listing file on the disk for each file compiled for later processing by the CREF program. These files have the filename of the source file and the extension of .CRF. The files can then be listed with the CREF command. However, with COBOL files, the cross-referenced listing is appended to the listing file. No additional command need be given to obtain the listing.
/FORTRAN	Compile the file with FORTRAN. Assumed for files with the extension of .F4 and all files with nonrecognizable processor extensions (if FORTRAN is the standard processor).
/FUDGE	Create a disk file containing the names of the .REL files produced by the command string. When the FUDGE command is given, PIP reads this file in order to generate a library REL file. Arguments to this switch are: /FUDGE:dev:file.ext[proj,prog] dev: - the device on which to write the file. DSK: is assumed. file.ext - the name of the library file. The filename is required. If the extension is omitted, it is assumed to be .REL. [proj,prog] - the directory in which to place the file. The user's directory is assumed if none is given. This switch is permanent in the sense that it pertains to all REL files generated by the command string.
/LIBRARY	Load the files in library search mode. This mode causes a program file in a special library file to be loaded only if one or more of its declared entry symbols satisfies an undefined global request in the source file. The system libraries are always searched. Refer to the LOADER documentation.
/LIST	Generate a disk listing file, for each file compiled, with the filename of the source file and the extension of .LST. These files can be listed later with the LIST command. Unless this switch is specified, listing files are not generated except in COBOL; COBOL listings are always generated.

(continued on next page)

Command Format (cont)

/LMAP	Produce a loader map during the loading process (same action as /MAP) containing the local symbols.
/MACRO	Assemble the file with MACRO. Assumed for files with extensions of .MAC.
/MACX11[1]	Assemble the file with MACX11. Assumed for files with extensions of .P11.
/MAP	Produce loader maps during the loading process. When this switch is encountered, a loader map is requested from the loader. After the library search of the system libraries, the map is written in the user's disk area with either the filename specified by the user (e.g., /MAP:file) or the default filename MAP.MAP. This switch is an exception to the permanent switch rule in that it causes only one map to be produced even though it appears as a permanent switch.
/NOCOMPILE	Complement the /COMPILE switch by not forcing a compilation on a source file whose date is not as recent as the date on the binary file. Note that this switch is not the same as the /REL switch, which turns off all compilation, even if the source file is newer than the REL file. /NOCOMPILE is the default action.
/NOLIST	Do not generate listing files. This is the default action except for COBOL files; COBOL listings are always generated.
/NOSEARCH	Loads all routines of the file whether the routines are referenced or not. Since this is the default action, this switch is used only to turn off library search mode (/LIBRARY). This is not equivalent to the /P LOADER switch, which does not search any libraries; the /NOSEARCH switch scans the system libraries.
/REL	Use the existing REL files although newer source files may be present.
/SNOBOL[2]	Compile the file with SNOBOL. Assumed for files with an extension of .SNO.

[1]MACX11, the PDP-11 assembler for the PDP-10, will be recognized as a processor only if the appropriate assembly switch is set. However, this assembly switch setting is not supported.

[2]SNOBOL will be recognized as a processor only if the appropriate assembly switch is set. However, this assembly switch setting is not supported.

EXECUTE command (Cont)

Characteristics

The EXECUTE command:

Places the terminal in user mode.
Runs the appropriate processor and the LOADER.
Starts the execution of the compiled and loaded program.

Associated Messages

Refer to Chapter 4.

Examples

```
.EXECUTE TEST)
MACRO: TEST
LOADING

LOADER 2K CORE
EXECUTION
```

FILCOM program

Function

The FILCOM program is used to compare two versions of a file and to output any differences. Generally, this comparison is line by line for ASCII files or word by word for binary files. FILCOM determines the type of comparison to use by examining either the switches specified in the command string or the extensions of the files. Switches always take precedence over file extensions.

Command Format

```
.R FILCOM↲
*output dev:file.ext [proj,prog] = input dev1:file.ext [proj,prog],
        input dev2:file.ext [proj,prog]
```

output dev:	= the device on which the differences are to be output.
input dev:	= the device on which an input file resides.

Defaults

1. If the entire output specification is omitted (including the equal sign), the output device is assumed to be TTY.
2. If an output filename is specified, the default output device is DSK.
3. If the output filename is omitted, the second input filename is used, unless it is null. In this case, the filename FILCOM is used.
4. If the output extension is omitted, .SCM is used on a source compare and .BCM is used on a binary compare.
5. If the [proj,prog] is omitted (input or output side), the user's default directory is assumed.
6. If an input device is omitted, it is assumed to be DSK.
7. If the filename and/or extension of the second input file is omitted, it is taken from the first input file.
8. A dot following the filename of the second input is necessary to explicitly indicate a null extension, if the extension of the first input file is not null.

FILCOM program (Cont)

Command Format (cont)

9. The second input file specification cannot be null unless a binary compare is being performed. In a binary compare, if the first input file is not followed by a comma and a second input file descriptor, the input file is compared to a zero file and is output in its entirety. This gives the user a method of listing a binary file. Refer to Example 4.

Switches

The following switches can appear in the command string, depending on whether a source compare or a binary source compare is being performed.

Binary Compare

/H	Type list of switches available (help text).
/nL	Specify the lower limit for a partial binary compare (n is an octal number). This switch, when used with the /nU switch, allows a binary file to be compared only within the specified limits.
/nU	Specify the upper limit for a partial binary compare (n is an octal number). This switch, when used with the /nL switch, allows a binary file to be compared only within the specified limits.
/W	Compare files in binary mode without expanding the files first (refer to Appendix D). This switch is used to compare two binary files with ASCII extensions.
/X	Expand SAV files before comparing them in binary mode. This action removes differences resulting from zero compression (refer to Appendix D).

Source Compare

/A	Compare files in ASCII mode. This switch is used to force a source compare on two ASCII files with binary extensions.
/B	Compare blank lines. Without this switch, blank lines are ignored.
/C	Ignore comments (all text on a line following a semicolon) and spacing (spaces and tabs). This switch does not cause a line consisting entirely of a comment to become a blank line, which is normally ignored.

(continued on next page)

Command Format (cont)

Source Compare (cont)

/H	Type list of switches available (help text).
/nL	Specify the number of lines that determine a match (n is an octal number). A match means that n successive lines in each input file have been found identical. When a match is found, all differences occurring before the match and after the previous match are output. In addition, the first line of the current match is output after the differences to aid in locating the place within each file at which the differences occurred. The default value for n is 3.
/S	Ignore spaces and tabs.
/U	Compare in update mode. This means that the output file consists of the second input file with vertical bars next to the lines that differ from the first input file. This feature is useful when updating a document because the changes made to the latest edition are flagged with change bars in the left margin. The latest edition of the document is the second input file.

If switches are not specified in the command string, the files are compared in the mode implied by the extension. The following extensions are recognized as binary and cause a binary compare if one or both of the input files have one of the extensions.

```
.BAC    .HGH    .RMT
.BIN    .LOW    .RTB
.BUG    .MSB    .SAV
.CAL    .OVR    .SFD
.CHN    .QUE    .SHR
.DAE    .QUF    .SVE
.DCR    .REL    .SYS
.DMP    .RIM    .UFD
                .XPN
```

Binary files are compared word by word starting at word 0 except for the following two cases:

1. Files with extensions .SHR and .HGH are assumed to be high segment files. Since the word count starts at 400000, upper and lower limits, if used, must be greater than (or equal to in the case of the lower limit) 400000.
2. Files with extensions .SAV, .LOW, and .SVE are assumed to be compressed core image files and are expanded before comparing.

FILCOM program (Cont)

Command Format (cont)

Conflicts are resolved by switches or defaults. If a conflict arises in the absence of switches, the files are assumed to be ordinary binary files.

Output

In most cases, headers consisting of the device, filename, extension, and creation date of each input file are listed before the differences are output. However, headers do not appear on output from the /U switch (update mode on source compare).

Source compare output - After the headers are listed, the following notation appears in the left column of the output

```
n)m
```

where

n is the number of the input file, and
m is the page number of the input file (see examples).

The right column lists the differences occurring between matches in the input files. Following the list of differences, a line identical to each file is output for reference purposes.

The output from the /U switch differs from the above-described output in that the output file created is the second input file with vertical bars in the left column next to the lines that are different from the first input file.

Binary compare output - When a difference is encountered between the two input files, a line in the following format appears on the output device:

```
octal loc.      first file-word      second file-word      XOR of both words
```

If the exclusive OR (XOR) of the two words differs only in the right half, the third word output is the absolute value of the difference of the two right halves. This usually indicates an address that changed.

If one input file is shorter than the other, after the end of file is encountered on the shorter file, the remainder of the longer file is output.

Characteristics

The R FILCOM command:

Places the terminal in user mode.
Runs the FILCOM program, thereby destroying the user's core image.

Associated Messages

Refer to Chapter 4.

Examples

1. The user has the following two ASCII files on disk:

```
First File          Second File

FILE A              FILE B

A                   A
B                   B
C                   C
D                   G
E                   H
F                   I                   page 1
G                   J
H                   1
I                   2
J                   3
K
L
M

First File          Second File

N                   N
O                   O
P                   P
Q                   Q
R                   R
S                   S
T                   T                   page 2
U                   U
V                   V
W                   4
X                   5
Y                   W
Z                   X
                    Y
                    Z
```

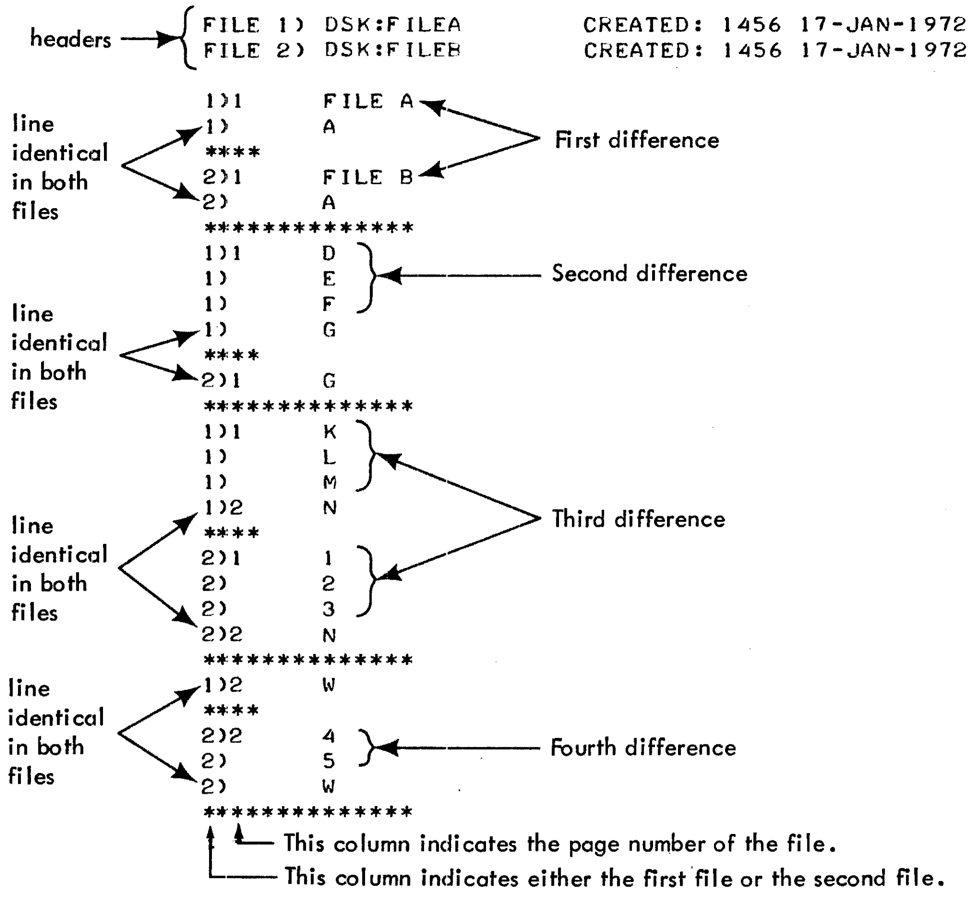
(continued on next page)

FILCOM program (Cont)

Examples (cont)

To compare the two files and output the differences on the terminal, the following sequence is used:

`.R FILCOM↵`	Run the FILCOM program.
`*FILEA,FILEB↵`	Compare the two files on disk and output the differences on the terminal. By default, three consecutive identical lines determine a match.



```
FILE 1) DSK:FILEA          CREATED: 1456 17-JAN-1972
FILE 2) DSK:FILEB          CREATED: 1456 17-JAN-1972

1)1     FILE A
1)      A
****
2)1     FILE B
2)      A
*************
1)1     D
1)      E
1)      F
1)      G
****
2)1     G
*************
1)1     K
1)      L
1)      M
1)2     N
****
2)1     1
2)      2
2)      3
2)2     N
*************
1)2     W
****
2)2     4
2)      5
2)      W
*************
```

headers
line identical in both files
First difference
line identical in both files
Second difference
line identical in both files
Third difference
line identical in both files
Fourth difference
This column indicates the page number of the file.
This column indicates either the first file or the second file.

(continued on next page)

Examples (cont)

To compare the two files and output the differences on the line printer, the following commands are used. Note that in this example the number of successive lines that determines a match has been set to 4 with the /4L switch.

```
.R FILCOM↵
*LPT:/4L=FILEA,FILEB↵

        FILE 1) DSK:FILEA          CREATED: 1456 17-JAN-1972
        FILE 2) DSK:FILEB          CREATED: 1456 17-JAN-1972

        1)1       FILE A
        1)        A
        1)        B
        1)        C
        1)        D
        1)        E
        1)        F
        1)        G
        ****
        2)1       FILE B
        2)        A
        2)        B
        2)        C
        2)        G
        **************
        1)1       K
        1)        L
        1)        M
        1)2       N
        ****
        2)1       1
        2)        2
        2)        3
        2)2       N
        **************
        1)2       W
        ****
        2)2       4
        2)        5
        2)        W
        **************
```

These lines are listed as being different because the /4L switch specifies that 4 consecutive lines must be found identical in the two files before they are considered as a match.

(continued on next page)

FILCOM program (Cont)

Examples (cont)

To compare the two files so that the second input file is output with vertical bars in the left column next to the lines that differ from the first input file, use the following command sequence.

```
.R FILCOM↲
*LPT:/U=FILEA,FILEB↲

                    FILE B

                    A
                    B
                    C
                    G
                    H
                    I
                    J
                    1
                    2
                    3
                    N
                    O
                    P
                    Q
                    R
                    S
                    T
                    U
                    V
                    4
                    5
                    W
                    X
                    Y
                    Z
```

The lines with vertical bars indicate the differences between the two files.

The lines with vertical bars indicate the differences between the two files.

2. To compare two binary files on the disk and output the differences on the terminal, use the following command sequence.

```
.R FILCOM↲
*TTY:←DSK:DIAL.REL,DIAL2↲
FILE 1) DSK:DIAL.REL     CREATED: 0000 23-DEC-1971
FILE 2) DSK:DIAL2.REL    CREATED: 0000 12-AUG-1971

000000  000004 000001    000004 000060           000057
000002  000000 054716    000311 372712    000311 326004
000003  000006 000001    017573 510354    017575 510355
000004  000000 000000    017573 513216    017573 513216
```

Examples (cont)

3. To compare two high segment files, the command sequence below is used. Note that the locations begin at 400000.

```
.R FILCOM)
*TTY:←SYS:TABLE.SHR, TABLE.SHR)
FILE 1) SYS:TABLE.SHR    CREATED: 2020 24-JAN-1972
FILE 2) DSK:TABLE.SHR    CREATED: 1829 30-NOV-1971

400000  001611 400010   001630 407157   000021 007147
400003  006675 000000   015024 407670   013651 407670
400004  005600 000070   004700 000113   001100 000163
400005  545741 444562   554143 625700   011602 261262
400010  634000 000000   260740 403516   454740 403516
400011  474000 000000   200000 414036   674000 414036
400012  402000 000156   202000 000720   600000 000676
400013  200040 406354   201000 000472   001040 406726
```

4. To list a binary file, use the following command sequence.

```
.R FILCOM)
*TTY:←SYS:DOT.REL)
000000  000004 000001
000001  000000 000000
000002  000000 054716
000003  000006 000001
000004  000000 000000
000005  000007 517716
000006  000001 000002
000007  000000 000000
  •
  •
  •
```

Note that the following sequence will not work because of the terminating comma.

```
*TTY:←SYS:DOT.REL,)

?COMMAND ERROR
```

FILCOM program (Cont)

Examples (cont)

5. To compare two binary files between locations 150-160 (octal).

```
.R FILCOM↲
*TTY:/15ØL/16ØU←SYS:SYSTAT.SAV,SYS:SYSDPY.SAV↲
FILE 1) SYS:SYSTAT.SAV   CREATED: Ø818 3Ø-NOV-1971
FILE 2) SYS:SYSDPY.SAV   CREATED: 1642 29-NOV-1971

ØØØ15Ø   2ØØ4ØØ ØØØ137    2ØØ74Ø ØØ3217    ØØØ34Ø ØØ332Ø
ØØØ151   26Ø74Ø ØØ4226    4Ø45ØØ ØØ4242    66424Ø ØØØØ64
ØØØ152   26Ø74Ø ØØ4253    6615ØØ ØØ2ØØØ    4Ø124Ø ØØ6253
ØØØ153   2ØØØ4Ø ØØ5Ø11    26Ø74Ø ØØ2723    Ø6Ø7ØØ ØØ7732
ØØØ154   26Ø74Ø ØØ4Ø63    2ØØØ4Ø ØØ4243    Ø6Ø7ØØ ØØØ22Ø
ØØØ155   2Ø1Ø41 777777    2Ø2Ø4Ø ØØ3241    ØØ3ØØ1 774536
ØØØ156   Ø47Ø4Ø ØØØØ42    2ØØØ4Ø ØØ4241    247ØØØ ØØ42Ø3
ØØØ157   254ØØØ ØØØ174    251Ø4Ø ØØ4142    ØØ5Ø4Ø ØØ4Ø36
ØØØ16Ø   476ØØØ ØØ6774    211Ø4Ø ØØØ144    667Ø4Ø ØØ663Ø
```

6. To compare two .SAV files. Note that the files are expanded before the comparison.

```
.R FILCOM↲
*TTY:←SYS:TRY1.SAV,SYS:TRY.SAV↲
FILE 1) SYS:TRY1.SAV     CREATED: 2Ø43 Ø5-JAN-1972
FILE 2) SYS:TRY.SAV      CREATED: Ø818 3Ø-NOV-1971

ØØØ114   ØØ4ØØØ ØØØ14Ø    ØØØØØØ ØØØØØØ    ØØ4ØØØ ØØØ14Ø
ØØØ116   777536 ØØ5536    ØØØØØØ ØØØØØØ    777536 ØØ5536
ØØØ117   ØØØØØØ ØØ5536    ØØØØØØ ØØØØØØ           ØØ5536
ØØØ12Ø   ØØ6ØØØ ØØØ14Ø    ØØ7222 ØØØ14Ø    ØØ1222 ØØØØØØ
ØØØ121   ØØØØØØ ØØ6ØØØ    ØØØØØØ ØØ7222           ØØ1222
ØØØ13Ø   Ø1ØØØØ ØØØØØ5    ØØØØØØ ØØØØØØ    Ø1ØØØØ ØØØØØ5
ØØØ133   ØØ3727 ØØ5777    ØØ6643 ØØ7777    ØØ5164 ØØ2ØØØ
ØØØ137   ØØ34ØØ ØØØØ7Ø    Ø467ØØ ØØØØØ4    Ø453ØØ ØØØØ74
ØØØ14Ø   264ØØØ ØØ1454    Ø47ØØØ ØØØØØØ    223ØØØ ØØ1454
ØØØ141   26ØØ4Ø ØØ1773    2ØØØ4Ø ØØ5Ø75    Ø6ØØØØ ØØ47Ø6
ØØØ142   2Ø124Ø ØØ1447    4Ø2ØØØ ØØ6644    60324Ø ØØ72Ø3
ØØØ143   54224Ø ØØ1634    251Ø4Ø ØØ7221    7132ØØ ØØ6415
ØØØ144   26ØØ4Ø ØØ2774    4Ø3ØØØ ØØØØ15    663Ø4Ø ØØ2761
ØØØ145   621ØØØ ØØØØ1Ø    476ØØØ ØØ6715    257ØØØ ØØ67Ø5
ØØØ146   2ØØ24Ø ØØ35Ø4    2ØØ74Ø ØØ66Ø6    ØØØ5ØØ ØØ53Ø2
ØØØ147   25124Ø ØØØØ12    Ø5114Ø ØØ5Ø76    2ØØ3ØØ ØØ5Ø64
ØØØ15Ø   4Ø2ØØØ ØØ3613    2ØØ4ØØ ØØØ137    6Ø24ØØ ØØ3724
ØØØ151   2Ø1Ø4Ø ØØ373Ø    26Ø74Ø ØØ4226    Ø617ØØ ØØ7516
ØØØ152   2ØØ26Ø ØØ3632    26Ø74Ø ØØ4253    Ø6Ø52Ø ØØ7461
ØØØ153   32124Ø ØØØ164    2ØØØ4Ø ØØ5Ø11    1212ØØ ØØ5175
```

FILE command

Function

The FILE command provides remote control of DECtape-to-disk and disk-to-DECtape transfers on operator-handled DECtapes.

Command Formats

1. FILE C

   Checks the queue of FILE commands to be read to determine if any of the user's requests are still pending. No argument is required. Pending requests will be listed.

2. FILE D, id, file.ext, file.ext, ...

   Deletes the specified files from DECtape. Requires Tape ID and list of filenames as arguments. The tape ID is any alphanumeric name of 6 characters or less that is used to identify the tape. Upon completion, an automatic FILE L is performed.

3. FILE F, id, file.ext, file.ext, ...

   Files information onto a DECtape. Requires Tape ID and list of filenames as arguments. Upon completion, an automatic FILE L is performed.

4. FILE L, id

   Reads the directory of a DECtape and places it in the user's disk area as an ASCII file with filename id.DIR. id is any alphanumeric name of 6 characters or less that is used to identify the tape. It is the only argument. The user may then read the directory with a monitor command string. (See Examples).

5. FILE R, id, file.ext, file.ext, ...

   Recalls (transfers) information from the user's DECtape to the disk. Requires Tape id and list of filenames as arguments. If the specified files already exist, they are superseded with the ones from the DECtape. If the specified files do not exist, they will be created on the first file structure in the job's search list for which creation is allowed. After the files are transferred, an automatic FILE L is performed.

FILE command (Cont)

Command Formats (cont)

6. FILE W

   Waits until all of the user's pending requests are processed before continuing. If there are pending requests, the message WAITING . . . is typed to the user. Control returns when all requests have been processed. The user may type control-C if he decides not to wait.

7. FILE Z, id, file.ext, file.ext, . . .

   Zeroes the directory of the DECtape before the files are copied and then performs the same operations as the F option. Requires Tape id and may have a list of filenames as arguments. After the files are copied, an automatic FILE L is performed.

The C and W funcitons are the only requests that are performed immediately. The other requests are placed in a queue to be performed whenever possible. The user's terminal and job are free to proceed before the request is completed. The function argument is optional. If the function argument is not specified, a brief dialogue is performed.

In most cases the user does not need to specify which file structures the files are on because UMOUNT determines this (with LOOKUPs) and passes the information to OMOUNT.

However, file structure names may be specified in file descriptors. When no structure name is explicitly typed, the default is initially the first file structure in the user's search list (implied by DSK:) on which he is allowed to create files. Refer to the description of the SETSRC program. When a file structure name is typed or implied, it becomes the new default.

The asterisk construction may be used, but care should be taken when generic DSK: is typed. Because DSK: may define many file structures, the single file structure is chosen as follows:

> When the asterisk construction is used for the filename or extension, the first structure on which the user may create files in his search list is used. This is called the user's standard file structure.

If the asterisk construction is not used and the file exists, the first file structure in the search list that contains the specified file is used, unless overridden by a default. (See Examples.) If the file does not exist, the standard structure is used.

WARNING

If the user has a search list with multiple file structures, the asterisk construction when used with the FILE R command can cause files to be created rather than superseded.

Characteristics

The FILE command:

Leaves the terminal in monitor mode.
Runs the UMOUNT program, thereby destroys the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Restrictions

The project-programmer number may not be specified in file descriptors.

Associated Messages

Refer to Chapter 4.

Examples

```
.FILE R,MINE,MAIN.F4,SUBFIL.MAC↲

REQUEST STORED
2.COMMANDS IN QUEUE
```

The files MAIN.F4 and SUBFIL.MAC are taken from the user's DECtape labeled MINE and placed on the first file structure in the user's search list for which creation is allowed. There are two commands in the queue (counting this one).

```
.FILE C↲
2. R JOB24 TTY5 27,235 MINE DSKB:,DSKB:MAIN,F4,SUBFIL.MAC↲
3. COMMANDS IN QUEUE
```

The user checks to see if his request is still waiting to be processed. The first line of the output indicates that the user's request is second in the queue (2.), that the request made is a RECALL (R), that the user's job is 24 (JOB24), that the user is on terminal 5 (TTY5) under the project-programmer number of [27,235] (27,235), that the tape is identified by the name MINE (MINE), and that the files will be written in the directory on DSKB: (DSKB:). The second line indicates that there are 3 commands in the queue.

(continued on next page)

FILE command (Cont)

Examples (cont)

`.FILE L, 4↲`	The user wants the directory on the DECtape labeled 4 to be placed in his disk area as an ASCII file.
`.TYPE 4.DIR↲`	The user then reads the directory file with the TYPE command.

If the user's search list is as follows:

DSKA/N, DSKB, DSKC

with file A on DSKA, file B on DSKB, and file C on DSKC, the following commands are equivalent:

```
The user types:           The user could have typed:     The command as passed to OMOUNT:

.FILE F,2,A,B,C           .FILE F,2,DSK:A,DSK:B,         .FILE F,2,DSKA:A, DSKB:B,
                                DSK:C                          DSKC:C
```

The first file structure that contains each file is used.

```
.FILE R,3,A,DSKB:B,C      .FILE R,3,DSK:A,DSKB:B,        .FILE R,3,DSKA:A,DSKB:B,
                                DSKB:C                         DSKB:C
```

The user changes the default to DSKB and even though file C exists on DSKC, file C is created on DSKB; files A and B are superseded.

```
.FILE F,1,*.*             .FILE F,1,DSK:*.*              .FILE F,1,DSKB:*.*
```

Because the asterisk convention was used, the first file structure on which the user may create files (DSKB) is used.

```
.FILE R,2,A,C.*,          .FILE R,2,DSK:A,               .FILE R,2,DSKA:A,
DSKC:B.*                  DSKB:C.*,DSKC:B.*              DSKB:C.*,DSKC:B.*
```

Because of the asterisk convention, DSKB is used for file C (even though file C exists on DSKC). The user explicitly typed a structure name for file B; therefore, DSKC is used even though file B is on DSKB. File A is superseded.

FILEX program

Function

The FILEX program is a general file transfer program intended to convert between various core image formats, and to read and write various directory formats. Files are transferred as 36-bit data. The only processing on the data is that necessary to convert between various core image representations.

Command Format

```
.R FILEX)
*dev:ofile.ext [proj,prog] <nnn>/switches = dev:ifile.ext [proj,prog]/switches
```

If the project-programmer and/or the switches appear after the device name, they apply to all the following files. If they appear after the filename, the specifiers apply only to the preceding file. The input filename or extension may be * in which case the usual processing of the * construction occurs (refer to the TYPE command). The output filename and extension may be * in which case the filename and extension of the input file is copied. If the output filename or extension is missing, the same procedure occurs as with the * construction, except that all core image files are written with the default extension and format appropriate to the output device (unless overridden by switches).

If a protection <nnn> is not specified, files are written with the system standard protection unless the files are being written on SYS. On SYS, files are written with protection <155>, except for files with extension .SYS. These files have the default protection of <157>.

Meaning of Switches:

Help text

/H	-	to obtain an explanation of the command string and individual switches.

DECtape Format Specifiers

/F	-	PDP-15 DECtape format
/M	-	MIT project MAC PDP-6/10 DECtape format
/O	-	Old DEC PDP-6 DECtape format
/T	-	normal PDP-10 directory format
/V	-	PDP-11 DECtape format (Note that PDP-11 contiguous files are not supported by FILEX.)

FILEX program (Cont)

Command Format (cont)

File Format Specifiers

/A	-	ASCII processing; meaningful only for PDP-11 and PDP-15 tapes.
/B	-	binary processing; overrides default extension.
/C	-	compressed; save file format. This format is assumed for files with extensions .SAV, .LOW, .SVE. The default output extension is .SAV unless the input extension is .LOW or .SVE, in which case the extension remains unchanged.
/D	-	dump format. This format is assumed for files with extension .DMP.
/E	-	expanded core image files (used by FILDDT). This format is assumed for files with extension .XPN. The default output extension is .XPN.
/I	-	image processing; meaningful only for PDP-11 and PDP-15 tapes.
/S	-	simple block (SBLK) format, project MAC's equivalent of .SAV format. The default output extension is .BIN.

DECtape Processing Specifiers

/G	-	(go on), ignores read errors on input device. FILEX checks the always-bad-checksum bit in the 5-series monitor, so this switch is not needed for files with .RPABC on (e.g., CRASH.SAV).
/L	-	(list), causes a directory on an input DECtape file to be typed on the terminal, or causes a directory listing of the output DECtape at the end (i.e., after the output).
/P	-	(preserved), causes quick processing (/Q) and preserves the scratch file after processing for use by another command.
/Q	-	(quick), causes an input or output DECtape to be processed quickly via a scratch file.
/R	-	(reuse), reuses a scratch file preserved by a /P in a previous command.
/Z	-	(zero), causes the appropriate format of a zeroed directory to be written on a DECtape output file. If TAPEID appears in the output specifier, then TAPEID is written as the tape identifier in the directory. TAPEID is preceded by a up arrow (↑) and may be 6 characters on a PDP-10 tape, 3 characters on a project MAC tape, and is not present on a PDP-6 tape.

Characteristics

The R FILEX command:

Runs the FILEX program, thereby destroying the user's core image.

Examples

`.R FILEX`
`*DSK:←DTA1:TEST.DMP/C`
	The dump format file is compressed and written as TEST.SAV.

`.R FILEX`
`*DSK:SER105.XPN[10,1]←DSKC:CRASH.SAV[1,4]`
	Copy CRASH.SAV to an expanded format file for FILDDT to examine.

FINISH command

Function

The FINISH command terminates any input or output currently in progress on the specified device and automatically performs the RELEASE UUO (which CLOSES the files) and DEASSIGN command, thus making the device available to another user. This command is preferred over the DEASSIGN command because it completely disassociates an INITed device from the user's job, thereby preventing the user from continuing his program. If the user wishes to continue his program, he should use the DEASSIGN command.

Command Format

FINISH dev

dev = the logical or physical name of the device on which I/O is to be terminated. This argument is optional.

If dev is omitted, I/O is terminated on all devices, except the job's controlling terminal and the logical name of the controlling terminal is cleared.

Characteristics

The FINISH command:

Leaves the terminal in monitor mode.
Requires core.
Depends on FTFINISH which is normally absent in the DECsystem-1040.

Restrictions

The user cannot continue his program if the device was INITed, but he can start at the beginning or enter DDT.

Associated Messages

Refer to Chapter 4.

Examples

```
.FINISH CDR:↵
.
.FINISH DTA7:↵
.
.FINISH LPT:↵
.
```

FUDGE command [1]

Function

The FUDGE command causes PIP to read a temporary file generated by a previous COMPILE, LOAD, EXECUTE, or DEBUG command using the /FUDGE switch and to create a library REL file. The library is created with the REL files in the same order in which they were specified in the command string containing the /FUDGE switch.

NOTE

Since the COMPIL program sorts out files by compilers, mixed FORTRAN and MACRO programs are sorted so that all FORTRAN programs are compiled first and MACRO programs second. However, the /FUDGE switch combines them in the order in which the COMPIL program encountered them.

Command Format

FUDGE

Characteristics

The FUDGE command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.COMPIL/FUDGE:LIBARY/MACRO TEST,MATH,DATPRO.CBL,SCIENC.F4)
```

Create a disk file named LIBARY which contains the names of all the REL files produced.

```
.FUDGE)
```

Create the library file and call it LIBARY. This file contains the following: TEST.REL, MATH.REL, DATPRO.REL, and SCIENC.REL.

[1] This command runs the COMPIL program, which interprets the command before running the PIP program.

FUDGE2 program

Function

The FUDGE2 program is used to update files containing one or more relocatable binary programs and to manipulate programs within program files. Three files are used in the updating process:

1. A master file containing the file to be updated.
2. A transaction file containing the file of programs to be used when updating.
3. An output file containing the updated file.

All three files can be on the same device if the device is DSK. The two input files can be on the same DECtape.

The desired function of FUDGE2 is specified by a command code at the end of the command string. Only one command code can be specified in each command string. The command string is then terminated with an ALTmode, represented in this manual by a dollar sign ($). Switches can also be used to manipulate file directories and to position a magnetic tape.

Command Format

```
.R FUDGE2)
*output dev:file.ext=master dev:file.ext<programs>, transaction dev:file.ext<programs>
         (command)$
```

output dev:	= the device on which the updated file is written. If omitted, DSK is assumed.
master dev:	= the device containing the file to be updated. If omitted, the default is DSK.
transaction dev:	= the device containing the files of programs to be used in the updating process. When more than one file is transferred from magnetic tape or paper tape, a colon must follow the device name for each file. For example, MTA: : : Transfer 3 files If the device is omitted, DSK is assumed.
file.ext	= the filename and extension of each file. Filenames must be specified for directory devices, but the extension can be omitted. If the extension is not given, it is assumed to be .REL unless the /L switch appears in the command string. In this case, the output extension .LST is assumed.

(continued on next page)

FUDGE2 program (Cont)

Command Format (cont)

file.ext (cont)	Project-programmer numbers appearing after a filename apply to that file only. If the project-programmer number appears before the filename, it applies to all subsequent files until another device is specified. The asterisk convention can be used with the input files (refer to Paragraph 1.4.2.4).
<programs>	= Names of programs (on DSK or DTA only) to be used in the updating process. They are grouped within angle brackets in the same order as they appear in the file and are separated by commas. When manipulating all the programs within a file, only the filename need be specified. Program names cannot appear for the output file.
(command)	= Code for the function to be performed. This code can be either preceded by a slash or enclosed in parentheses and must appear at the end of the command string. Each command results in the updated file being output to the output device. The command codes are as follows:

A	Append the specified programs in the transaction file(s) to the master file.
C	Delete local symbols from the master file.
D	Delete the specified programs from the master file.
E	Extract the specified files and/or programs from the input files. The entire file is extracted if program names are not specified.
H	Type the commands and switches available.
I	Insert programs from the specified transaction files into the master file. The programs from the transaction files are inserted immediately before the specified programs in the master file.
L	List the names and lengths of all relocatable programs within a file. The length is in one of two forms: low segment break, high segment break or program break, absolute break The length of FORTRAN programs is not output.

(continued on next page)

Command Format (cont)

(command) (cont)	R	Replace the specified programs in the master file with the specified programs in the transaction file. The number of replacing programs must be the same as the number of programs to replace.
	S	List all the entry points within a program. These entry points are listed across the page.
	X	Write index blocks into a library file. Index blocks are used in a direct access library search (refer to the LOADER documentation). This command implies a C command.

File directories can be manipulated and magnetic tapes positioned by including switches in the command string. These switches can appear anywhere in the command string and are preceded by a slash or enclosed in parentheses. The following switches are available:

/B	Backspace a magnetic tape one file.
/K	Advance a magnetic tape one file.
/T	Skip to the logical end-of-tape on a magnetic tape.
/W	Rewind a magnetic tape.
/Z	Clear the directory of the output DECtape.

Characteristics

The R FUDGE2 command:

Places the terminal in user mode.
Runs the FUDGE2 program, thereby destroying the user's core image.

Associated Messages

Refer to Chapter 4.

FUDGE2 program (Cont)

Examples

```
.R FUDGE2
*LPT:=DTA1:LIB40(L)$
```

List all relocatable programs (.REL) from the file LIB40, located on DTA1, on the line printer.

```
*DSK:LIB4BB=DTA2:LIB4AA <EXP.3,EXP.3C>,
DTA1:F4<EXP.3A,EXP.3B>(R)$
```

Replace programs EXP.3 and EXP.3C located in file LIB4AA on DTA2, with programs EXP.3A and EXP.3B in File F4 on DTA1; write out the new LIB4AA file on disk and call it LIB4BB.

```
*DTA1:NFILE=DSK:MFILE<M1,M2,M3,M4>
DTA3:TFILEA<TA1,TA2>
DTA4:TFILEB<TB1,TB2>/I$
```

Insert into MFILE the programs TA1 and TA2 from TFILEA, and TB1 and TB2 from TFILEB. Create NFILE with the following order:

TA1,M1,TA2,M2,TB1,M3,TB2,M4

Insertion is on a one-to-one basis. If there are more programs to be inserted than specified programs before which they are to be inserted, the extra files are ignored.

```
*DTA1:NFILE=DSK:MFILE<M1,M2,M3,M4>
DTA3:TFILEA
DTA4:TFILEB/I$
```

However, in this example (where TFILEA and TFILEB contain the programs TA1 and TA2 and TB1 and TB2, respectively) create an NFILE with the following order:

TA1,TA2,M1,TB1,TB2,M2,M3,M4

```
*DTA2:TESTA=MTA1:(WK),MTA2: :(ZA)$
```

Clear the directory of DTA2; rewind MTA1 and advance the tape one file; append the first two program files from MTA2 to the second file on MTA1 and write out the resultant file on disk, calling it TESTA.

```
*OUTPUT=LIBARY,DTA1:LIBARY<FILEY,FILEZ>/A$
```

Append the programs FILEY and FILEZ contained in the file LIBARY on DTA1 to the end of the file LIBARY on disk. Write the new file on disk and call it OUTPUT.

(continued on next page)

Examples (cont)

`*NEWFIL=OLDFIL<TEST,SUBTRC,MULTI>,BASFIL<PROG,`
`ROUTIN,ANSWER>,SUBFIL<MATH>(E)$`
	Extract the specified programs from the files OLDFIL, BASFIL, and SUBFIL and create a new output file called NEWFIL. The order of the programs in NEWFIL is as follows: TEST, SUBTRC, MULTI, PROG, ROUTIN, ANSWER, MATH.

`*↑C`	Return to the monitor.

GET command

Function

The GET command loads a core image from a retrievable storage device but does not begin execution.

This command clears all of user core. However, programs should not count on this action and should explicitly clear those areas of core that are expected to contain zeroes (i.e., programs should be self-initializing). This action allows programs to be restarted by a ↑C, START sequence without having to do another GET command.

On magnetic tape, if the low or high segment is missing, a null record is output before the EOF for the missing segment so that two EOFs cannot occur consecutively. Therefore, a saved null segment does not appear as a logical EOT (2 EOFs in a row).

Command Format

GET dev:file.ext [proj,prog] core

The arguments and the defaults are the same as in the RUN command.

The extension applies to the low file, not the high file. An extension of .SHR, then .HGH, is assumed for the high file. If the user types an extension of .SHR or .HGH, the extension is treated as a null extension since .SHR and .HGH are confusing as low file extensions.

Characteristics

The GET command:

Leaves the terminal in monitor mode.
Does not operate when the device is currently transmitting data.

Associated Messages

Refer to Chapter 4.

Example

```
.GET SYS:PIP↵
JOB SETUP

.GET TEST↵
JOB SETUP
```

GLOB program

Function

The GLOB program reads multiple binary program files and produces an alphabetical cross-referenced list of all the global symbols (symbols accessible to other programs) encountered. This program also searches files in library search mode, checking for globals, if the program file was loaded by the LOADER in library search mode (refer to the LOADER documentation).

The GLOB program has two phases of operation; the first phase is to scan the input files and build an internal symbol table, and the second, to produce output based on the symbol table. Because of these phases, the user can input commands to GLOB in one of two ways. The first way is to specify one command string containing both the output and input specifications. (This is the command string format most system programs accept.) The second is to separate the command string into a series of input commands and output commands.

Command Formats

1. R GLOB

   outdev:file.ext[proj,prog] = input dev:file.ext[proj,prog],file.ext,...,dev:file.ext [proj,prog] ($)

2. R GLOB

   followed by one or more input commands in the form

   dev:file.ext[proj,prog],file.ext[proj,prog],...,dev:file.ext[proj,prog],... ↵

   and then one or more output commands in the form

   outdev:file.ext[proj,prog] = ($)

When the user separates his input to GLOB into input commands and output commands (Command Format #2), the input commands contain only input specifications and the output commands, only output specifications. Each output command causes a listing to be generated; any number of listings can be printed from the symbol table generated from the current input files as long as no input commands occur after the first output command. When an input command is encountered after output has been generated, the current symbol table is destroyed and a new one begun.

Defaults

1. If the device is omitted, it is assumed to be DSK. However, if the entire output specification is omitted, the output device is TTY.

(continued on next page)

GLOB program (Cont)

Command Format (cont)

Defaults (cont)

2. If the output filename is omitted, it is the name of the last input file. The input filenames are required.
3. If the output extension is omitted, .GLB is used. If the input extension is omitted, it is assumed to be .REL unless the null extension is explicitly specified by a dot following the filename.
4. If the project-programmer number [proj,prog] is omitted, the user's default directory is used.
5. An ALTmode terminates the command input and signals GLOB to output the cross-referenced listing. In other words, a listing is not output until GLOB encounters an ALTmode. The ALTmode appears at the end of the command string shown in Command Format #1 or at the end of each output command shown in Command Format #2.

Switches

Switches control the types of global listings to be output. Each switch can be preceded by a slash, or several switches can be enclosed in parentheses. Only the most recently specified switch (except for L, M, P, Q, and X, which are always in effect) is in effect at any given time. If no switches are specified, all global symbols are output. The following switches are available.

/A	Output all global symbols. This is the default if no switches are specified.
/E	List only multiple defined or undefined (erroneous) symbols.
/F	List nonrelocatable (fixed) symbols only.
/H	List the switches available (help text).
/L	Scan programs only if they contain globals previously defined and not yet satisfied (library search mode).
/M	Turn off library search mode scanning resulting from a /L switch.
/N	List only symbols which are never referenced.
/P	List all routines that define a symbol to have the same value. The routine that defines the symbol first is listed followed by a plus (+) sign. Subsequent routines that define the symbol are listed preceded by a plus sign.
/Q	Suppress the listing of subsequent definers that result from the /P switch.

(continued on next page)

Command Format (cont)

Switches (cont)

/R	List only relocatable symbols.
/S	List symbols with non-conflicting values that are defined in more than one program.
/X	Do not print listing header when output device is not the terminal, and include listing header when it is the terminal. Without this switch, the header is printed on all devices except the terminal. The listing header is in the following format:

```
FLAGS   SYMBOL   OCTAL VALUE   DEFINED IN   REFERENCED IN
```

Symbols listed are in alphabetical order according to their ASCII code values. The octal value is followed by a prime (') if the symbol is relocatable. The value is then relative to the beginning of the program in which the symbol is defined. Flags preceding the symbol are shown below.

M	Multiply defined symbol (all values are shown).
N	Never referred to (i.e., was not declared external in any of the binary programs).
S	Multiply specified symbol (i.e., defined in more than one program but with non-conflicting values). The name of the first program in which the symbol was encountered is followed by a plus sign.
U	Undefined symbol.

Characteristics

The R GLOB command:

Places the terminal in user mode.
Runs the GLOB program, thereby destroying the user's core image.

Associated Messages

Refer to Chapter 4.

GLOB program (Cont)

Examples

`.R GLOB↵`	Run the GLOB program.
`*LPT:=MAIN,DTA2:SUB40,SUB50 Ⓢ`	All global symbols in the programs MAIN (on DSK), SUB40, and SUB50 (on DTA2) are listed on the line printer. Along with the symbol is listed its value, the program in which it is defined, all programs in which it is referenced, and any error flags.
`*DTA4:BATCH.REL,DATA,DTA6:NUMBER.REL,CLASS↵`	
`*DSK:MATH.REL,LIBARY.↵`	The programs to be scanned are BATCH.REL, DATA.REL on DTA4; NUMBER.REL, CLASS.REL on DTA6; and MATH.REL, LIBARY.null on DSK.
`*LPT:=/F Ⓢ`	List only nonrelocatable symbols on the line printer.
`*DSK:SYMBOL=/R Ⓢ`	List only relocatable symbols in the file named SYMBOL in the user's default directory.

`*TTY:=/E Ⓢ`
`U EXTSYM SUBRTE`
	Print all erroneous symbols on the terminal. EXTSYM is an undefined symbol appearing in the program SUBRTE.

`*↑C`	Return to monitor mode.

GRIPE program

Function

The GRIPE program accepts text from a user and records it in a disk file, thereby enabling users to record comments and complaints to be read at a later time by the operations staff.

Command Format

R GRIPE

When the GRIPE program responds with a YES?, type the text, using as many lines as necessary, terminated with an ESCAPE. The text is written as a file with <157>protection and includes a header with the date, time, and project-programmer number of the user writing the comment. Therefore, the user does not need to identify himself.

Characteristics

The R GRIPE command:

Places the terminal in user mode.
Runs the GRIPE program, thereby destroying the user's core image.

Associated Messages

Refer to Chapter 4.

Example

```
.R GRIPE)

YES? (TYPE ESCAPE WHEN THROUGH) THIS CONSOLE IS
ALMOST OUT OF PAPER$
THANK YOU


.
```

HALT command

Function

The HALT (↑C) command transmits a HALT command to the monitor command interpreter. It stops the job and stores the program counter in the job data area (.JBPC). Refer to the DECsystem-10 Monitor Calls for a description of the job data area.

Command Format

HALT (↑C)

Characteristics

The HALT (↑C) command:

Places the terminal in monitor mode.
Does not require LOGIN.

Example

```
↑C

.
```

HELP command

Function

The HELP command is used to obtain useful documentation on various system features.

Command Formats

1. HELP

   outputs the instructions for receiving information.

2. HELP *

   outputs the names of features that have available documentation.

3. HELP name

   outputs the information on the named feature.

Only the first six characters of the argument to the command are scanned. These characters must be A through Z, 0 through 9, or asterisk (*).

Characteristics

The HELP command:

Leaves the terminal in monitor mode.
Does not require LOGIN.

Associated Messages

Refer to Chapter 4.

HELP command (Cont)

Examples

```
.HELP *⤶

HELP IS AVAILABLE FOR THE FOLLOWING:
BATCON  CDPSPL  CDRSTK  DIRECT  FAILSA  FGEN
HELP    LPTSPL  MATCH   MTCOPY  PIP     PLTSPL
PTPSPL  SOUP    TECO


.HELP DIRECT⤶

TYPE OUT=INPUT+INPUT+...
/ACCESS:N = ACCESS ALL LISTED FILES UNDER N BLOCKS LONG
/ALLOCATED = GIVE ALLOCATED LENGTH
/CHECKSUM = COMPUTE CHECKSUM OF EACH FILE
/DETAIL = EVERYTHING FROM EXTENDED LOOKUP
/F = FAST MODE
/H = THIS TEXT
/L = OUT TO LPT
/PHYSICAL = DO PHYSICAL OPENS
/S = SLOW MODE
/SORT = PREPARE FOR SORTING
/SUMMARY = JUST PRINT SUMMARY LINE
/TITLES = INCLUDE TITLES
/UNITS = GIVE SPECIFIC UNIT
/WIDTH:N = TRY TO FILL PAPER WIDTH OF N COLUMNS
/WORDS = OUTPUT LENGTHS IN WORDS

* IS WILD NAME, ETC.
? IS WILD LETTER OF NAME, ETC.
"OUT=" MAY BE OMITTED
DEFAULT IS TTY:.DIR=DSK:*.*[MY DIRECTORY]
```

INITIA command

Function

The INITIA command performs standard system initialization for the terminal issuing the command. This command is issued automatically at system startup and at the 400 series restart at certain designated terminals, but may be re-issued at any time by the user. This command is used to initiate specific system programs, such as the operator service program, OPSER, on a particular console.

The INITIA command runs SYS:INITIA.SAV which, depending upon the system configuration and the number of the TTY from which it is typed, may cause any of a number of events to occur. For more information, refer to the INITIA specification in Notebook 7 of the DECsystem-10 Software Notebooks.

Command Format

INITIA

Characteristics

The INITIA command:

Leaves the terminal in monitor mode.
Runs a specific system program.
Does not require LOGIN.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.INITIA↲
5S04 SYS #2 22:12:17 TTY24
.
```

JCONTINUE command

Function

The JCONTINUE command forces a continue of the specified job if the job was in a ↑C state because of a call to the device error message routine (HNGSTP).

Command Format

JCONTINUE n

n = the number of the job to be continued. This argument is required.

Characteristics

The JCONTINUE command:

Places the terminal in monitor mode.
Does not require LOGIN.
Depends on FTJCON which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Example

```
.JCONT 14⤸
```

KJOB command

Function

In multiprogramming systems, the KJOB command:

Stops all assigned I/O devices and returns them to the monitor pool.

Returns all allocated core to the monitor pool.

Returns the job number to the pool.

Leaves the console in the monitor mode.

Performs an automatic TIME command.

In swapping systems, the KJOB command performs all the above procedures. In addition, the command responds with

CONFIRM:

The user may type ↑C to abort logout, or type an optional file structure name (or list of file structure names) preceded by one of the following:

F↵	to logout immediately saving all files (including temporary files) as they are. Identical to R LOGOUT, or RUN UUO to LOGOUT.
D↵	to delete all files on the specified file structures. Responds with ARE YOU SURE? Type Y or D for YES, any other character for NO.
K↵	to delete all unprotected files (i.e., files with 0xx protection code) on the specified file structures. If project 1 or other jobs are logged-in with the same project-programmer number, responds with ARE YOU SURE? Type Y or K for YES, any other character for NO.
P↵	to save and protect (i.e., assign a protection code of 1 in the owner's field) all but temporary files (TMP, CRF, LST) on the specified file structures. If project 1 or other jobs are logged-in with the same project-programmer number, responds with ARE YOU SURE? Type Y or P for YES, any other character for NO.
S↵	to save without protecting all but temporary files on the specified file structures. If project 1 or other jobs are logged-in with the same project-programmer number, responds with ARE YOU SURE? Type Y or S for YES, any other character for NO.
L↵	to list the directories of the specified file structures.
I↵	to individually determine what to do with all files on the specified file structure as follows:

(continued on next page)

KJOB command (Cont)

Function (cont)

After each filename is listed, type

P↵	to protect the file.
S↵	to save the file.
K↵	to delete the file.
Q↵	to learn if over logged-out quota on this file structure. If not over quota, nothing is typed, and the same filename is repeated.
E↵	to skip to next file structure and save this file if below logged-out quota for this file structure. If not below logged-out quota, a message is typed and the same filename is repeated.
H↵	to list responses and meanings.

U↵	to individually determine what to do with all but protected files. Protected files are always preserved.
B↵	to delete no files except when user is over the logged-out quota, then delete enough files to be below quota. The files are deleted in the following order: 1) unprotected files according to the category of the file, 2) spooled files not previously queued, and 3) protected files according to the category of the file. The categories of files are as follows: 1) temporary files, 2) relocatable files, 3) backup files, 4) save files, and 5) all other files.
Q↵	to learn if over logged-out quota on the specified file structures.
H↵	to list the KJOB options and their meanings.
W↵	to list the names of the files that are deleted.
X↵	to turn off the listing of the names of the files that are deleted. Complement of W.

If no file structure names are specified, the responses are for all file structure names in the job search list. If file structure names are specified, the responses apply to those file structures, and CONFIRM is retyped. The KJOB command ignores all logical assignments.

The user has the option of going through the CONFIRM dialogue, even if other jobs are logged-in under the same project-programmer number or if he is logged-in under project 1. (However, if sufficient responses are included on the KJOB command line or in a temporary file entered through an alternate entry point, CONFIRM is not typed.) By responding to a CONFIRM message, the user has an opportunity to organize his disk area by deleting or preserving specific files.

The KJOB program calls the QUEUE program to perform the queuing of files which have been deferred to logout time. This includes all spooled output unless the user has specifically queued output spooling earlier. Queuing may be suppressed with the /Z response (see below).

Command Formats

1. KJOB

   CONFIRM:

   When the CONFIRM: response is given, the user may type any of the above-described letters followed by an optional file structure name or list of file structure names separated by commas. The user may type one of the above-described letters, followed by optional file structure names, on the same line as the KJOB command, and the CONFIRM: message will not be typed.

2. KJOB <log file descriptor> = / <letter> <list of file structure names>/<letter> <list of file structure names> etc.

   <log file descriptor> has the following form: <dev:file.ext [proj,prog] >. If the log file is not a disk or spooled device, TTY is used.

   <letter> = any letter from the above-described set. In addition, the following responses are available to any jobs using this command format:

   /Z:n specifies the degree of queuing desired:

   n = 0 suppresses all normal queuing done at LOGOUT time.
   n = 1 queues the log file only.
   n = 2 queues the log file and spooled output (*.LPT, etc.)
   n = 3 queues the log file, spooled output, and *.LST.
   n = 4 queues the log file, spooled output, *.LST, and any requests deferred to LOGOUT time (deferred requests are not yet implemented).

   If Z is given without a value or if there are no spool bits set for job, Z:0 is assumed. Otherwise, /Z:2 is assumed.

   /VL:n specifies that the limit of pages for LPT files is to be n (decimal).

   /VC:n specifies that the limit of cards for CDP files is to be n (decimal).

   /VT:n specifies that the limit of feet of paper tape for PTP files is to be n (decimal).

   /VP:n specifies that the limit of minutes for PLT files is to be n (decimal).

   /VR:n specifies that the priority of the queue request is to be n; n is from 0 through 62. /VR:62 is the standard.

(continued on next page)

KJOB command (Cont)

Command Formats (cont)

/VS:n specifies that the sequence number for the queue request is to be n.

/VD:v specifies that the file disposition of the log file is to be v.

v = D	deletes the log file after printing.
v = P	preserves the log file after printing.
v = R	renames the log file before printing to the queue area and deletes it after printing.

Default is /VD:R.

If a value to the above switches is not specified, the value is equivalent to 0 (e.g., /VD is equivalent to /VD:0). For the /Vx switches, a value of 0 is equivalent to the standard (e.g., /VD = /VD:0 = /VD:R).

The letters must appear on the input side of the command string. If the log file is specified, all TTY output is appended to the log file. If no log file is specified or if the log file is not a disk or spooled device, the default is TTY. In addition, if responses to CONFIRM are required and are not specified on the KJOB command line, these responses will then be read from TTY. Therefore, users should be careful when employing this command format.

3. The KJOB program may be entered at the CCL entry point through the RUN UUO. When this is done, TMPCOR file KJO or disk file nnnKJO.TMP, where nnn is the user's job number in decimal, is used instead of the TTY input. This temporary file has the following format:

```
KJOB <log file descriptor> = / <list of file structure names>/ <letter>
<list of file structure names> etc.
```

Characteristics

The KJOB command:

Detaches the terminal.
Stops all assigned I/O devices since it does not operate when a device is currently transmitting data.
Runs the KJOB and LOGOUT programs.
Does not require LOGIN.

Associated Messages

Refer to Chapter 4.

Examples

1. An example of the CONFIRM dialogue.

```
.K↵
CONFIRM: I↵
DSKB:
TEST4 .TST   <055>  2000. BLKS  : K↵
TEST5 .TST   <055>  505. BLKS   : P↵
T11   .BAK   <055>  5. BLKS   : K↵
T2    .BAK   <055>  5. BLKS   : K↵
T3    .BAK   <055>  5. BLKS   : K↵
TEST  .BAK   <055>  5. BLKS   : K↵
TEST  .REL   <055>  5. BLKS   : S↵
TEST  .MAC   <055>  5. BLKS   : P↵
TEST  .SHR   <055>  30. BLKS   : S↵
JOB 5, USER [10,63] LOGGED OFF TTY24 AT 2309 11-MAY-71
DELETED 5 FILES
SAVED 4 FILES 2565 TOTAL BLOCKS USED
RUNTIME 0 MIN, 00.60 SEC
```

2. An example of the user bypassing the CONFIRM dialogue.

```
.K/F↵
JOB 9, USER [10,110] LOGGED OFF TTY3  1349  18-MAR-71
SAVED ALL 23 FILES (630. DISK BLOCKS)
RUNTIME 1 MIN, 51.52 SEC
```

3. An example of the command when used in the Batch system. The output appears in the log file.

```
12:20:51 MONTR K DSKB0:MUM.LOG[10,110]=/W/B/VL:200
12:21:02 LGOUT JOB 12, USER [10,110] LOGGED OFF TTY50 1221 18-MAR-71
12:21:02 LGOUT SAVED ALL 38 FILES (1275. DISK BLOCKS)
12:21:02 LGOUT ANOTHER JOB STILL LOGGED IN UNDER [10,110]
12:21:02 LGOUT RUNTIME 0 MIN, 00.65 SEC
```

4. An example of the User specifying two switches.

```
 .K/W/B↵
DELETED:
MYFILE
JOB 14, USER [20,275] LOGGED OFF TTY35  1454  13-APR-72
DELETED 1 FILES (1022. DISK BLOCKS)
SAVED 52 FILES (1735. DISK BLOCKS)
RUNTIME 0 MIN, 02.60 SEC
```

LIST command [1]

Function

The LIST command directs PIP to list the contents of named source file(s) on the line printer (LPT). The output goes either to LPT immediately or to the disk to be spooled to LPT if it is being spooled for this job. Refer to the QUEUE and PRINT commands. If the LPT is being spooled, the QUEUE program should always be used since it saves time and disk accesses.

Command Format

LIST list

list = a single file specification or a sting of file specifications separated by commas. A file specification consists of a device name, a filename and extension, and a directory name. This argument is required.

Switches can be passed to PIP by enclosing them in parentheses in the LIST command string. When COMPIL interprets the command string, it passes the switches on to PIP.

Characteristics

The LIST command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.LIST TEST.*↲
.LIST *.MAC↲
.LIST DTA4:A,B,C↲
```

[1] This command runs the COMPIL program, which interprets the command before running PIP.

LOAD command [1]

Function

The LOAD command translates the specified source files if necessary (function of COMPILE command), runs the LOADER, and loads the .REL files generated. The assembler or compiler used is determined by the source file extension or by switches in the command string (refer to the COMPILE command). If a REL file already exists with a more recent date than that of the source file, compilation is not performed (unless requested via a switch).

This command generates a core image but does not begin execution. At this point, the user can start his program or save the core image for future execution.

Each time the COMPILE, LOAD, EXECUTE, or DEBUG command is executed, the command with its arguments is remembered in a temporary file on disk, or in core if the monitor has the TMPCOR feature. Therefore, the filename used last can be recalled for the next command without specifying the arguments again (refer to Paragraph 1.5).

The LOAD command accepts several command constructions: the @ construction (indirect commands), the + construction, the = construction, and the < > construction. Refer to Paragraph 1.5 for a complete description of each of these constructions.

Command Format

LOAD list

list = a single file specification, or a string of file specifications separated by commas. A file specification consists of a device name, a filename with or without an extension, and a directory name.

The following switches can be used to modify the command string. These switches can be temporary or permanent switches (refer to Paragraph 1.5.5).

/ALGOL	Compile the file with ALGOL. Assumed for files with the extension of .ALG.
/BLISS[2]	Compile the file with BLISS. Assumed for files with the extension of .BLI.
/COBOL	Compile the file with COBOL. Assumed for files with the extension of .CBL.

(continued on next page)

[1] This command runs the COMPIL program, which interprets the command before running the appropriate processor and the LOADER.

[2] BLISS will be recognized as a processor only if the appropriate assembly switch is set. However, this assembly switch setting is not supported.

LOAD command (Cont)

Command Format (cont)

/COMPILE	Force a compilation of this file even though a binary file exists with a newer date and time than the source file. This switch is used to obtain an extra compilation (e.g., in order to obtain a listing of the compilation) since normally compilation is not performed if the binary file is newer than the source file.
/CREF	Produce a cross-reference listing file on the disk for each file compiled for later processing by the CREF program. These files have the filename of the source file and the extension of .CRF. The files can then be listed with the CREF command. However, with COBOL files, the cross-referenced listing is always appended to the listing file. No additional command need be given to obtain the listing.
/FORTRAN	Compile the file with FORTRAN. Assumed for files with the extension of .F4 and all files with nonrecognizable processor extensions (if FORTRAN is the standard processor).
/FUDGE	Create a disk file containing the names of the .REL files produced by the command string. When the FUDGE command is given, PIP reads this file in order to generate a library REL file. Arguments to this switch are: /FUDGE:dev:file.ext [proj,prog] dev: - the device on which to write the file. DSK: is assumed. file.ext - the name of the library file. The filename is required. If the extension is omitted, it is assumed to be .REL. [proj,prog] - the directory in which to place the file. The user's directory is assumed if none is given. This switch is permanent in the sense that it pertains to all REL files generated by the command string.
/LIBRARY	Load the files in library search mode. This mode causes a program file in a special library file to be loaded only if one or more of its declared entry symbols satisfies an undefined global request in the source file. The default libraries are always searched. Refer to the LOADER documentation.
/LIST	Generate a disk listing file, for each file compiled, with the filename of the source file and the extension of .LST. These files can be listed later with the LIST command. Unless this switch is specified, listing files are not generated except in COBOL; COBOL listings are always generated.

(continued on next page)

Command Format (cont)

/LMAP	Produce a loader map during the loading process (same action as /MAP) containing the local symbols.
/MACRO	Assemble the file with MACRO. Assumed for files with extensions of .MAC.
/MACX11[1]	Assemble the file with MACX11. Assumed for files with an extension of .P11.
/MAP	Produce a loader map during the loading process. When this switch is encountered, a loader map is requested from the loader. After the library search of the default libraries, the map is written with the filename specified by the user (e.g., /MAP:file) or with the default filename MAP.MAP in the user's disk area. This switch is an exception to the permanent compile switch rule in that it causes only one map to be produced although it may appear as a permanent switch.
/NOCOMPILE	Complement the /COMPILE switch by not forcing a compilation on a source file whose date is not as recent as the date on the binary file. Note that this switch is not the same as the /REL switch, which turns off all compilation, even if the source file is newer than the REL file. /NOCOMPILE is the default action.
/NOLIST	Do not generate listing files. This is the default action except for COBOL files; COBOL listings are always generated.
/NOSEARCH	Load all routines of the file whether the routines are referenced or not. Since this is the default action, this switch is used only to turn off library search mode (/LIBRARY). This is not equivalent to the /P LOADER switch because /P does not search any libraries where /NOSEARCH will scan the default libraries.
/REL	Use the existing .REL files although a newer source file may be present.
/SNOBOL[2]	Compile the file with SNOBOL. Assumed for files with an extension of .SNO.

Characteristics

The LOAD command:

Leaves the terminal in monitor mode.
Runs the appropriate processor and the LOADER.

[1] MACX11 (the PDP-11 assembler for the PDP-10) will be recognized as a processor only if the appropriate assembly switch is set. However, this assembly switch setting is not supported.

[2] SNOBOL will be recognized as a processor only if the appropriate assembly switch is set. However, this assembly switch setting is not supported.

LOAD command (Cont)

Associated Messages

Refer to Chapter 4.

Examples

```
.LOAD TEST)
MACRO: TEST
LOADING

LOADER 2K CORE

EXIT

.
```

LOCATE command

Function

The LOCATE command logically establishes the user's job at a specified station. When the job is initiated, the user's logical station corresponds to his physical station. Therefore, this command is needed only if the user desires to change his logical station.

Command Format

LOCATE nn

nn = the station number.

An argument of 0 denotes the central station. A null argument implies the station of the user's terminal, i.e., his physical station.

Characteristics

The LOCATE command:

Leaves the terminal in monitor mode.
Depends on FTREM which is normally absent in the DECsystem-1040.

Restrictions

The LOCATE command must specify a station that is currently in contact with the central station.

Associated Messages

Refer to Chapter 4.

Examples

```
.LOCATE 2)
.
.LOC 0)
.
```

LOGIN command

Function

The LOGIN command is used to gain access to the system. This command loads a Monitor Support program which accepts the user's LOGIN data. The user types in his project and programmer numbers followed by his password. To login successfully, the project and programmer numbers and the password typed in by the user must match the project and programmer numbers and password stored in the system accounting file (SYS:ACCT.SYS).

Command Format

LOGIN proj,prog

proj,prog = the user's project-programmer number. The project and programmer numbers may be separated by either a comma or a slash. If a slash is used, the message of the day is not output to the user unless the date on the file containing the message (NOTICE.TXT) is later than the last time the user logged-in. If this is true, the message is typed only once, whereas, when the comma is used, the message is output every time the user logs in. This argument may be typed on the same line as the LOGIN command, or on the following line after LOGIN types out the number sign.

Characteristics

The LOGIN command:

Returns the terminal to monitor mode or starts a program running if specified in ACCT.SYS entry for proj,prog.
Runs the LOGIN program.
Does not require LOGIN.

Associated Messages

Refer to Chapter 4.

Example

The following is the procedure used to gain access to the system.

```
.LOGIN 27,235↲
JOB 21    5S0417A TTY23
```

LOGIN types the job number assigned to user (job number 21), followed by monitor name, version number, and console line number. If the user does not type his project-programmer number on the same line as the LOGIN command, LOGIN outputs a number sign indicating that the user should type in his project-programmer number.

```
PASSWORD:
```

System requests user to type his password. User types password followed by carriage return (refer to Paragraph 1.4.2.1). To maintain password security, the monitor does not echo the password. On terminals with local-copy (refer to DECsystem-10 Monitor Calls), a mask is typed to make the password unreadable.

```
1135 8-JUN-71 THUR
TYPE SYS:SCHED FOR NEXT
WEEKS SCHEDULE
.
```

If user entries are correct, the system responds with time, date, day of the week, the message of the day (if any), and a period, indicating readiness to accept another command.

MAKE command [1]

Function

The MAKE command runs TECO (Text Editor and Corrector) and creates a new file on the disk. If a file already exists with the same name, a warning message is given and the file is superseded. Refer to the TECO manual in Notebook 6 of the DECsystem-10 Software Notebooks.

Command Format

MAKE dev:file.ext [proj,prog]

dev: = the device or file structure name on which the file is to be created. If omitted, DSK: is assumed.

file.ext = any legal filename and filename extension. The filename is required; the filename extension is optional.

[proj,prog] = the directory in which the file is to be created. If omitted, the user's default directory is assumed. Note that the default directory may be an SFD or some other UFD.

Characteristics

The MAKE command:

Places the terminal in user mode.
Runs the TECO program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Example

```
.MAKE TEST3.MAC↲

*
```

[1] This command runs the COMPIL program, which interprets the commands before running TECO .

MOUNT command

Function

The MOUNT command allows the user to request assignment of a device via the operator. This command is similar to the ASSIGN command, but, whereas the ASSIGN command operates without operator communication, the MOUNT command requests operator interaction when necessary. For example, if a Batch user requests a DECtape drive and all drives are in use, then the operator can free one for the user, if he wishes. The user can request devices from the restricted pool of devices.

The MOUNT command gives the operator greater control over assignment of devices on the system. When a user requests a device via this command, the operator has the option of either selecting a specific unit (e.g., DTA5) or cancelling the request completely (all units of this type are in use and the operator does not want to free one for this user). The operator may also mount the media for the requested unit if the media is sufficiently identified (e.g., a deck of cards in the card reader or an identified DECtape on a specific drive).

When the MOUNT command is used to gain access to a file structure, it allows the user to specify a particular drive, places the file structure name at the end of the job's search list, and waits for completion of operator action, if desired. Each file structure can have an administrative file, QUOTA.SYS, which contains a list of quotas for all users allowed access to the structure. When the file structure is mounted, a UFD is created for the user if he has an entry in QUOTA.SYS on the file structure.

The MOUNT command runs the UMOUNT program in the user's core area. UMOUNT scans the command string and completes as much of the command as possible without operator intervention. When operator intervention is required, UMOUNT queues a request to the OMOUNT program by writing a command file on the 3,3 disk area. OMOUNT examines these command files and interacts with the operator. When the command file is deleted, the operator action has been completed. UMOUNT waits for this completion of operator action unless the user types a control-C. When a control-C is typed, the user does not receive a message of confirmation, but can later use the /CHECK switch to see if his request is still pending (see Examples).

Command Format

MOUNT dev: log-dev /switches (drives)

dev: = one of the following: (1) a physical device name (e.g., DTA3, CDR, MTA), (2) a logical name previously associated with a physical device by either a MOUNT or ASSIGN command, or (3) a file structure name (one that is already mounted or one whose name appears in STRLST.SYS). This argument is required.

(continued on next page)

MOUNT command (Cont)

Command Format (cont)

log-dev = any SIXBIT name that is not the same as dev:. In other words, it may not be a physical device name or logical name that is currently being used or has previously been used as dev:. This argument is optional.

switches = The following switches are optional and only enough characters to make the switch unique are required. The unique names are underlined below.

/CHECK	Check and list pending requests.
/HELP	Type this list.
/MULTI	Multi-access, disk only, complement of /SINGLE, default condition.
/PAUSE	Notify the user before sending the message to the operator for a request. The user can then abort the command if desired.
/RONLY	Read only, same as /WLOCK.
/SINGLE	Only this job can access files on the structure (single access), file protection is enforced for him, disk only.
/VID:name	A visual identification passed to the operator as a comment to assist him in identifying a particular unit to mount. The argument can be in one of two forms: 1) any string of up to 25 characters containing only letters, digits, periods, and hyphens, or 2) any string of up to 25 characters enclosed in single quotes. However, break characters and single quotes are not allowed in the string. The naming and use of this switch is relevant only to the extent that the installation operator knows what it means. The PLEASE command should be used for any complex procedures or long communications with the operator.
/WENABL	Write enable for this job, complement of /WLOCK, default condition.

(continued on next page)

Command Format (cont)

/WLOCK	Write locked for this job. This job cannot write on this file structure and the monitor will not update BAT blocks or the access date. If /SINGLE is given, the operator may set hardware write lock to ensure that nothing is written.

(drives) = the physical drives on which the units are to be mounted. A drive argument may be used only when mounting file structures. The drives must be in the logical unit order within the file structure. Drive names are separated by commas. Leading and embedded drives that are not specified must be represented by null names (,,DPA3). Unspecified trailing drives may be omitted. Drive names are as follows:

Blank, null - unspecified. UMOUNT finds one of proper type.

Two letters - controller class (e.g., DP).

Three letters - specific controller (e.g., DPA). UMOUNT finds a drive on that controller.

Three letters and one or two digits - specific drive (e.g., DPA0, DPA1).

The user, by specifying a drive list, may force the packs to be mounted on specific drives or controllers. If no drive (or incomplete) specification is given, an available drive of the proper type is found.

Characteristics

The MOUNT command:

Places the terminal in user mode.
Runs the UMOUNT program, thereby destroying the user's core image.
Depends on FTCCLX and FTMOUN which are normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

MOUNT command (Cont)

Examples

```
.MOUNT PRIV:↲
PRIV MOUNTED
```

Asks the operator to mount the file structure PRIV.

```
.MOUNT PAY:(DPA,,DPB)/S
```

Requests that the first unit of file structure PAY be mounted on Controller A, the second unit on any controller, the third unit on controller B, and any remaining units on any drives. The structure will be single access (i.e., available only to this job).

```
.MOUNT MINE:↲
OPERATOR NOTIFIED
WAITING...
↑C
  .
  .
  .
```

Mount the file structure MINE.
The request is queued to the operator.
UMOUNT is waiting for the request to be completed.
The user does not wait for confirmation.

```
.MOUNT/CHECK↲
```

The user wants to know if his request has been processed.

```
NONE PENDING
.R SETSRC↲
*T
DSKA,DSKB,PRIV,PAY,MINE,FENCE
*↑C
.MOUNT DTA INPUT
```

The request has been processed.
The user wants to know if the file structure is in his search list.
The file structure has been added to his search list.

The user wants the operator to select a DECtape drive and assign it with logical name INPUT.

```
OPERATOR NOTIFIED
WAITING...
INPUT (DTA5) MOUNTED
.MOUNT INPUT/VID:325
```

The request is queued to the operator.
UMOUNT is waiting for the request to be completed.
The operator has selected DTA5.
The user asks the operator to mount the DECtape labeled 325. He may use either DTA5 or INPUT to refer to his device. For example, the Batch user would use INPUT since he would not know what DECtape drive he is assigned.

```
OPERATOR NOTIFIED
WAITING...
INPUT (DTA5) MOUNTED
.MOUNT INPUT OUTPUT

OUTPUT (DTA5) MOUNTED
```

The request is queued.
UMOUNT is waiting for confirmation.
The mount is successful.
The user changes the logical name to OUTPUT.
The logical name INPUT is no longer valid.
The mount is successful.

OPSER program

Function

The OPSER program facilitates multiple job control from a single operator terminal by allowing the operator to run several jobs called subjobs from his terminal. The OPSER program acts as the supervisor of the various subjobs by allowing monitor level or user level commands to be passed to all of the subjobs or to selected subjobs. Output from the various subjobs may be retrieved by OPSER.

The subjobs of OPSER run on psuedo-TTYs (refer to DECsystem-10 Monitor Calls) and all initializations of the pseudo-TTYs are performed by OPSER. The operator needs only to provide the subjob name, either an OPSER-provided subjob number or an operator-assigned name. System programs that require a dedicated terminal can be run as subjobs of OPSER. By running system jobs on pseudo-TTYs, OPSER is able to maintain an I/O link between the running jobs and the operator. In addition, the output from the various subjobs is concentrated on one terminal instead of many, as was the case when each system program required its own terminal.

Refer to the MPB Operator's Manual in the DECsystem-10 Software Notebooks for complete information on OPSER.

Command Format

R OPSER

OPSER signifies its readiness to process commands by typing an asterisk if no subjobs are in use or subjobs are in a wait for an operator action. OPSER responds with an exclamation point when a subjob is running. Commands may be entered whenever OPSER is operating. Each command is preceded by a colon and must be typed to sufficient length to make it unique.

OPSER Commands

:AUTO/hh:mm filespec	Process the specified file as an automatic startup file. The file is terminated by an end-of-file or the typing of a line on the console by the operator. This is the normal way that the standard subjobs are started by the operator. The time argument is optional; when it is given, the AUTO file is run at the specified time.
:BATMAX n[1]	Specify the maximum number of batch jobs allowed.
:BATMIN n[1]	Specify the minimum number of batch jobs guaranteed.

(continued on next page)

[1]Not yet implemented.

OPSER program (Cont)

Command Format (cont)

:CLOSE	Close the disk log file without opening a new one.
:CURRENT	Type the number of the current subjob (the last one typed into). Output from another subjob does not affect current subjob.
:DAYTIME	Obtain the current date and time.
:DEFINE xxx=n	Associate the symbol xxx as the mnemonic for subjob number n. The symbol B is reserved for the subjob running BATCON.
:DEVICE nam:log:n	Assign the device with the physical name nam and logical name log to subjob n. The logical name is optional but a null field must be typed if the name is omitted, e.g., :DEVICE CDR::3.
:ERROR n	Report only error messages (that is, ignore nonerror messages from subjob n). Message reporting is resumed with the :REVIVE command.
:EXIT	Exit to the monitor if no subjobs are in use; otherwise give a list of those that are running. This should be used instead of ↑C, since EXIT does not return the job to monitor mode if there are any active subjobs.
:FREE	Type the first free subjob number.
:HELP	Type a text which briefly explains the commands.
:JCONT n	Continue the specified stopped job.
:KJOB, n,m,p	Kill the specified subjobs saving all files. Causes /Z:0 to be included to KJOB so spooled files are not queued.
:KILL n,m,p	Kill the specified subjobs. This is identical to :KJOB.
:KSYS hhmm	Stop all timesharing at the time specified by hhmm.
:LOGIN proj,prog	Login a new subjob. If no project-programmer number is typed, assume OPSER's project-programmer number.
:MSGLVL 0	Cause the response to the :WHAT command to include the JOBSTS bits.

(continued on next page)

Command Format (cont)

:MSGLVL 1	Cause the response to :WHAT command to eliminate the JOBSTS bits.
:QUEUE <line>	Initiate the first free subjob and send the typed-in line to the system queue manager.
:RESOURCES	Type the list of the available system resources.
:RESTRICT dev[1]	Make the specified device a restricted device (i.e., one that is assignable only by a privileged job or the MOUNT program).
:REVIVE n	Resume normal echoing of output from subjob n.
:SEND <line>	Simulate the SEND monitor command.
:SET a	Simulate a SET monitor command. Valid SET monitor commands are SET CORMAX, SET CORMIN, SET DATE, SET DAYTIME, SET LOGMAX, SET OPR TTY, SET SCHED, and SET TTY.
:SET RUN CPUn	Add CPUn to the pool of CPUs to be used for running jobs.
:SET RUN NO CPUn	Remove CPUn from the pool of CPUs to be used for running jobs.
:SILENCE n	Ignore all output from subjob n.
:SLOGIN proj,prog	LOGIN one subjob but suppress its response. If proj, prog is omitted, OPSER uses its own.
:STOP n	Put the specified subjob in monitor mode. This is equivalent to inputting two control-C's in interactive mode.
:SYSTAT xx	Run SYSTAT with optional argument xx over the first free subjob.
:TLOG filespec	Create a disk log file with the specified name. If the file already exists, a message is typed to determine whether the existing file should be superseded. If not, the file is appended to the existing one. Default for filespec is OPSER.LOG.
:TTYTST	Test this terminal by typing all the ASCII characters between octal 40 and 174, inclusive.

(continued on next page)

[1]Not yet implemented.

OPSER program (Cont)

Command Format (cont)

:UNRESTRICT dev[1]	Make the specified device a unrestricted device (i.e., one that is assignable by both privileged and non-privileged jobs).
:WHAT n,m,p	Type the status of the specified subjobs on the terminal. The status includes a SYSTAT with the time, the time of the last input and the last output, and a linear listing of the JOBSTS bits.

When a subjob number or name is required in a command string, the subjob may be specified in one of four ways. It can be omitted, in which case the last subjob typed into is used. The mnemonic ALL may be used, in which case all active subjobs are implied. A decimal number can be used from zero to the limit OPSER is generated for. Finally, a mnemonic can be assigned to the subjob with the :DEFINE command.

Examples

```
.R OPSER)
*:AUTO CTY.ATO
```

To start an automatic startup file.

```
:MSGLVL 0
:TLOG
:SLOG
:DEFINE DAE=
DAE-R DAEMON
:SLOG
:DEFINE M=
M-R OMOUNT
M-START
:SLOG
:DEFINE L=
L-R LPTSPL
L-START
:SLOG
:DEFINE B=
B-MJOB 5
B-R BATCON
B-START
```

An example of an automatic startup file.

[1]Not yet implemented.

PJOB command

Function

The PJOB command causes the monitor to respond with the job to which the user's terminal is attached.

Command Format

PJOB

Characteristics

The PJOB command:

Leaves the terminal in monitor mode.

Associated Messages

Refer to Chapter 4.

Example

```
.PJOB↵
1

.
```

PLEASE command

Function

The PLEASE command allows the user non-conflicting two-way communication with the designated station operator.

Command Format

PLEASE dev: prog! text

dev = any terminal not assigned to a job (i.e., is not a job's controlling terminal) with which the user wishes to communicate, including:

a. TTYn: directs the text to a specific terminal unit. The default is TTY0.
b. OPRnn: directs the text to the operator's terminal station nn.
c. (null argument) directs the text to TTY0 at the central station.

prog! = the name of the system program to be run automatically when the message is completed. This argument may appear before or after the device argument and must be concluded with an exclamation point. If PLEASE is entered at the CCL entry point, it reads file nnnPLS.TMP. This file is sent to the designated device. After the operator terminates the request, the specified program will be run at its CCL entry point. Neither the dev: or prog! argument can be used from a Batch control file.

text = the user's message. The argument is required. Characters are not transmitted until the RETURN, vertical tab, or form feed key is depressed, at which point the entire line is transmitted.

When the user depresses the RETURN, vertical tab, or form feed key, a message informing the operator of the caller's station number, proj-prog number or user's name if monitor job tables are available, and text message is printed on dev:. An ESCAPE or control-C on either the user's terminal or dev: causes communication to terminate and the user's TTY to be left in monitor mode. Note that when the line terminates with an ESCAPE, the line is typed but the operator response is not waited for. Messages may be typed in both directions without retyping the command.

Characteristics

The PLEASE command:

Places the terminal in user mode until ESCAPE is typed.
Runs a system program except when used with Batch.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Restrictions

For Batch users, the PLEASE command is trapped by the Batch Controller and only PLEASE text is allowed. It can be used to request operator action while in the Batch mode. The line of text can only be one line terminated with an ESCAPE.

Associated Messages

Refer to Chapter 4.

Example

```
.PLEASE TELL ME WHEN DTA3 WILL BE FREE
OPERATOR HAS BEEN NOTIFIED
IN HALF AN HOUR
THANKS
↑C

.
```

PLOT command

Function

The PLOT command is used to place entries in the plotter output queue. This command is equivalent to the following form of the QUEUE command:

QUEUE PLT:jobname = list of input specifications

Command Format

PLOT jobname = list of input specifications

jobname = name of the job being entered into the queue. The default is the name of the first file in the request, not the name of the first file given. These differ when the first file given does not yet exist.

input specifications = a single file specification or a string of file specifications, separated by commas, for the disk files being processed. A file specification is in the form dev:file.ext[proj,prog].

dev: = any file structure to which PLTSPL will have access; the default is DSK:.

file.ext = names of the files. The filename is optional. The default for the first filename is *, the default for subsequent files is the last filename used. The extension can be omitted; the default is .PLT.

[proj,prog] = a directory to which the user has access; the user's directory is assumed if none is specified.

If no arguments are given with the command (i.e., only the command name is given), the entriẽs for all jobs of all users are output. The asterisk convention can be used for the input specifications. Switches that aid in constructing the queue entry can appear as part of the input specifications. These switches are divided into three categories:

1. Queue-operation - Only one of these switches can be placed in the command string because they define the type of queue request. The switch used can appear anywhere in the command string.
2. General - Each switch in this category can appear only once in the command string because they affect the entire request. The switch used can appear anywhere in the command string.

(continued on next page)

Command Format (cont)

3. File control - Any number of these switches can appear in the command string because they are specific to individual files within the request. The switch used must be adjacent to the file to which it applies. If the switch precedes the filename, it becomes the default for subsequent files.

The following switches can be used with the PLOT command:

Switch	Explanation	Category
/AFTER:tt	Process the request after the specified time; it is either in the form of hhmm (time of day) or +hhmm (time later than the current time). The resulting AFTER time must be less than the DEADLINE time. If the switch, or the value of the switch, is omitted, no AFTER constraints are assumed.	General
/BEFORE:t	Queue only the files with creation dates before t where t is in the form dd-mmm-yy hhmm.	General
/BEGIN:n	Start the output after n feet. The default is to start output at the beginning.	File Control
/COPIES:n	Repeat the output the specified number of times. n must be less than 64. If more than 63 copies are needed, two separate requests must be made. If the switch is omitted, single copies are output.	File Control
/CREATE	Make a new entry into the plotter output queue. This switch is the default for the queue-operation switches.	Queue Operation
/DEADLINE:tt	Process the request before the specified time; tt is either in the form hhmm (time of day) or +hhmm (time later than the current time). The resulting DEADLINE time must be greater than the AFTER time. If the switch, or the value of the switch, is omitted, no DEADLINE constraints are assumed.	General
/DISPOSE:DELETE	Delete the file after spooling.	File Control
/DISPOSE:PRESERVE	Save the file after spooling. This is the default for all files except files with extensions .LST, .TMP, and, if the protection is 0xx, .PLT.	File Control

(continued on next page)

PLOT command (Cont)

Command Format (cont)

Switch	Explanation	Category
/DISPOSE:RENAME	Rename the file from the specified directory immediately, remove it from the logged-out quota and delete it after spooling. This is the default for files with extensions .LST, .TMP, and if the protection is 0xx, .PLT.	File Control
/F	List the entires in the plotter queue, but do not update the queues. Therefore, the list may not be an up-to-date listing but the listing will be faster than with /LIST.	Queue Operation
/FORMS:a	Place the output on the specified form. The argument to the switch must be six alphabetic characters. The default is that normal forms are used.	General
/KILL	Remove the specified entry from the plotter queue. This switch can be used for deleting a previously submitted request as long as the request has not been started by the spoolers.	Queue Operation
/LIMIT:n	Limit the output to the specified number of pages.	General
/LIST	List the entries in the plotter queue; if the switch, along with all other switches, is omitted, all entries for all jobs of all users are listed.	General
/MODIFY	Alter the specified parameters in the job. This switch requires that the user have access rights to the job. It can be used for altering a previously submitted request as long as the request has not been started by the spoolers.	Queue Operation
/NEW	Accept the request even if the file does not yet exist.	File Control

(continued on next page)

Command Format (cont)

Switch	Explanation	Category
/NOTE:a	Plot the specified text (a) in the output.	File Control
/NULL	Accept the request even if there is nothing in the request. No error message is given.	General
/OKNONE	Do not output message if no files match the wildcard construction. This is assumed at KJOB time.	File Control
/PHYSICAL	Suppress logical device name assignments for the device specified.	File Control
/PLOT:ASCII	Plot the file in ASCII mode. If the /PLOT: switch is omitted, the file is plotted in the data mode specified in the file.	File Control
/PLOT:BINARY	Plot the file in binary mode. If the /PLOT: switch is omitted, the file is plotted in the data mode specified in the file.	File Control
/PLOT:IMAGE	Plot the file in image mode. If /PLOT: switch is omitted, the file is plotted in the data mode specified in the file.	File Control
/PRIORITY:n	Assign the specified external priority (n=0 to 62) to the request. The larger the number, the greater priority the job has. The default is 10 if no switch is given and 20 if the switch is specified without a value.	General
/PROTECT:nnn	Assign the protection nnn (octal) to the job. If the switch or the value of the switch is omitted, the standard protection is assumed.	General
/REMOVE	Remove the file from the queue. This switch is valid only with /MODIFY and can be used to remove a previously submitted file as long as the spoolers have not started processing the request.	File Control
/SEQ:n	Specify a sequence number to help in identifying a request to be modified or deleted.	General
/SINCE:t	Queue only the files with creation dates after the specified time t where t is in the form dd-mmm-yy hhmm.	General

(continued on next page)

PLOT command (Cont)

Command Format (cont)

Switch	Explanation	Category
/START:n	Start on the nth line of the file. If the switch, or the value of the switch is omitted, the first line is assumed.	File Control
/STRS	Search for the file on all file structures in the search list and take each occurrence. The default is to take just the first occurrence.	File Control
/UNPRESERVED	Output the files only if they are not preserved (i.e., the first digit of the protection code is 0). This switch avoids redundant plotting.	General

Characteristics

The PLOT command:

Leaves the terminal in monitor mode.
Runs the QUEUE program, thereby destroying the user's core image.
Depends on FTQCOM which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

`.PLOT *.PLT/FORMS:PLAIN)`	Cause all files with the extension PLT in the user's area to be plotted. Because these are spooled files (i.e., have the extension .PLT), the files are renamed out of the user's area immediately, and deleted after plotting. The operator is asked to put PLAIN paper on the plotter.

PRESERVE command [1]

Function

The PRESERVE command renames the specified files with the standard protection inclusively ORed with 100 (usually 155 or 157). The files are then preserved and KJOB will not delete them unless requested to. This command has the same action as the P argument to the KJOB command when individually determining what to do with each file.

Command Format

PRESERVE file1.ext,file2.ext,file3.ext,...

The full wildcard construction can be used for either the filename or the extension.

Characteristics

The PRESERVE command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Example

```
.PRESERVE TEST.MAC)
.PRE PROG, COLE.F4,NAME.*)
```

[1] This command runs the COMPIL program, which interprets the command before running PIP.

PRINT command

Function

The PRINT command is used to place entries into the line printer output queue. This command is equivalent to the following form of the QUEUE command:

QUEUE LPT:jobname = list of input specifications

Command Format

PRINT jobname = list of input specifications

jobname = name of the job being entered into the queue. The default is the name of the first file in the request, not the name of the first file given. These differ when the first file given does not yet exist.

input specifications = a single file specification or a string of file specifications, separated by commas, for the disk files being processed. A file specification is in the form dev:file.ext[proj,prog].

dev: = any file structure to which LPTSPL will have access; the default is DSK:.

file.ext = names of the files. The filename is optional. The default for the first filename is *, the default for subsequent files in the last filename used. The extension can be omitted; the default is .LPT.

[proj,prog] = a directory to which the user has access; the user's directory is assumed if none is specified.

If no arguments are given with the command (i.e., only the command name is given), the entries for all jobs for all users are output.

The asterisk convention can be used for the input specifications. Switches that aid in constructing the queue entry can appear as part of the input specifications. These switches are divided into three categories:

1. Queue-operation - Only one of these switches can be placed in the command string because they define the type of queue request. The switch used can appear anywhere in the command string.

2. General - Each switch in this category can appear only once in the command string because they affect the entire request. The switch used can appear anywhere in the command string.

(continued on next page)

Command Format (cont)

3. File control - Any number of these switches can appear in the command string because they are specific to individual files within the request. The switch used must be adjacent to the file to which it applies. If the switch precedes the filename, it becomes the default for subsequent files.

The following switches can be used with the PRINT command:

Switch	Explanation	Category
/AFTER:tt	Process the request after the specified time; tt is either in the form of hhmm (time of day) or +hhmm (time later than the current time). The resulting AFTER time must be less than the DEADLINE time. If the switch, or the value of the switch, is omitted, no AFTER constraints are assumed.	General
/BEFORE:t	Queue only the files with a creation date before time t, where t is in the form dd-mmm-yy hhmm. If this switch is omitted, no BEFORE constraints are assumed.	General
/BEGIN:n	Start the output on the nth page. The default is to begin output on the first page.	File Control
/COPIES:n	Repeat the output the specified number of times. n must be less than 64. If more than 63 copies are needed, two separate requests must be made. If this switch is omitted, one copy is given.	File Control
/CREATE	Make a new entry into the line printer output queue. This switch is the default for the queue-operation switches.	Queue Operation
/DEADLINE:tt	Process the request before the specified time; tt is either in the form hhmm (time of day) or +hhmm (time later than the current time). The resulting DEADLINE time must be greater than the AFTER time. If the switch, or the value of the switch, is omitted, no DEADLINE constraints are assumed.	General
/DISPOSE:DELETE	Delete the file after spooling.	File Control
/DISPOSE:PRESERVE	Save the file after spooling. This is the default for all files except files with extensions of .LST, .TMP, and, if the protection is 0xx, .LPT.	File Control

(continued on next page)

PRINT command (Cont)

Command Format (cont)

Switch	Explanation	Category
/DISPOSE:RENAME	Rename the file from the specified directory immediately, remove it from the logged-out quota, and delete it after spooling. This is the default for files with extensions .LST, .TMP, and, if the protection is 0xx, .LPT.	File Control
/F	List the entries in the line printer queue, but do not update the queues. Therefore, the list may not be an up-to-date listing but the listing will be faster than with /LIST.	Queue Operation
/FILE:ASCII	Indicate that the input file format is to be interpreted as ASCII text. This is assumed for all files with extensions other than .DAT.	File Control
/FILE:COBOL	Indicate that the input file format is to be interpreted as COBOL SIXBIT text.	File Control
/FILE:FORTRAN	Indicate that the input file format is to be interpreted FORTRAN ASCII text (obeys FORTRAN carriage control characters). This is assumed for files with the extension of .DAT.	File Control
/FORMS:a	Place the output on the specified form. The argument to the switch must be six alphabetic characters. The default is that normal forms are used.	General
/HEADER:0 or 1	Output block headers at the beginning of the file, if 1 (default). Do not output headers, if 0.	File Control
/KILL	Remove the specified entry from the Batch input queue. This switch can be used for deleting a previously submitted request as long as the request has not been started by the spooler.	Queue Operation
/LIMIT:n	Limit the output to the specified number of pages.	General
/LIST	List the entries in the line printer queue; if the switch, along with all other switches, is omitted, all entries for all jobs of all users are listed.	Queue Operation
/LOG	Define the file that the spoolers will use to record their process. The default is jobname .LOG.	File Control

(continued on next page)

Command Format (cont)

Switch	Explanation	Category
/MODIFY	Alter the specified parameters in the job. This switch requires that the user have access rights to the job. It can be used for altering a previously submitted request as long as the request has not been started by the spooler.	Queue Operation
/NEW	Accept the request even if the file does not yet exist.	File Control
/NOTE:a	Print the specified text (a) in the output.	File Control
/NULL	Accept the request even if there is nothing in the request. No error message is given.	General
/OKBINARY	Print files whose extensions imply binary information. Normally files with extensions .SAV, .SHR, .LOW, .REL, and .HGH will not appear in the print queue.	File Control
/OKNONE	Do not output message if no files match the wildcard construction. This is assumed at KJOB time.	File Control
/PHYSICAL	Suppress logical device name assignments for the device specified.	File Control
/PRINT:ARROW	Convert all control characters to up-arrow format except 011-015 and 020-024. This is the default.	File Control
/PRINT:ASCII	Send the file to the line printer with no changes.	File Control
/PRINT:OCTAL	Perform an octal dump of the file.	File Control
/PRINT:SUPPRESS	Suppress all carriage-control characters except for ASCII code characters LF and CR; this switch implies the use of the /PRINT:ARROW and is equivalent to the operator command to the spooler (SUPPRESS).	File Control
/PRIORITY:n	Assign the specified external priority (n=0 to 62) to the request. The larger the number, the greater priority the job has. The default is 10 if no switch is given and 20 if the switch is specified without a value.	General
/PROTECT:nnn	Assign the protection nnn (octal) to the job. If the switch, or the value of the switch, is omitted, the standard protection is assumed.	General

(continued on next page)

PRINT command (Cont)

Command Format (cont)

Switch	Explanation	Category
/REMOVE	Remove the file from the queue. This switch is valid only with /MODIFY and can be used to remove a previously submitted file as long as the spooler has not started processing the request.	File Control
/REPORT:code	Print the specified report within a COBOL report file. Code can be up to 12 characters in length.	File Control
/SEQ:n	Specify a sequence number to help in identifying a request to be modified or deleted.	General
/SINCE:t	Queue only the files with creation dates after the specified time t where t is in the form dd-mmm-yy hhmm.	General
/SPACING:DOUBLE	Double-space the output lines.	File Control
/SPACING:SINGLE	Single-space the output lines. This is the default if no /SPACING switch is used.	File Control
/SPACING:TRIPLE	Triple-space the output lines.	File Control
/START:n	Start on the nth line of the file. If the switch, or the value of the switch, is omitted, the first line is assumed.	File Control
/STRS	Search for the file on all file structures in the search list and take each occurrence. The default is to take just the first occurrence.	File Control
/UNPRESERVED	Output the files only if they are not preserved (i.e., the first digit of the protection code is 0). This switch avoids redundant printing.	General

Characteristics

The PRINT command:

Leaves the terminal in monitor mode.
Runs the QUEUE program, thereby destroying the user's core image.
Depends on FTQCOM which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

`.PRINT NOTICE.TXT`	Print the file DSK:NOTICE.TXT.
`.PRINT SYSTAT.SCM/DISP:REN/COP:2`	Print two copies of the file DSK:SYSTAT.SCM from the user's default area. Rename the file out of the user's area immediately and delete it after spooling.
`.PRINT *.TXT/HEAD:0/FORMS:2PART`	Print all files in the user's area which have the extension .TXT. Do not print file headers between the files. Print the files on forms known to the operator as 2PART.
`.PRINT /SEQ:356/KILL`	Remove the request with sequence number 356 from the LPT queue. This is accepted only if the spooler has not started processing the request.
`.PRINT LOADER.SAV/OKBINARY/PRINT:SUPPRESS`	Print a file known to be a binary file and suppress all carriage control characters except CR and LF.
`.PRINT PRGMAC.REL/PRINT:OCTAL`	Print an octal dump of the file PRGMAC.REL.

PROTECT command [1]

Function

The PROTECT command renames the specified files with the requested protection. The action of this command is similar to the R switch in PIP.

The protection of a file is indicated by three octal digits. Each digit represents a particular class of user. The first digit represents the owner of the file, the second represents users with the same project number of the owner, and the third represents all of the other users. Each number in the three digit code can be one of the following:

7	No access privileges
6	Execute the file only
5	Read and execute the file
4	Append, read, and execute the file
3	Update, append, read, and execute the file
2	Write, update, append, read, and execute the file
1	Rename, write, update, append, read, and execute the file
0	Change protection, rename, write, update, append, read, and execute the file.

The standard protection is normally 057 which means the owner has all privileges (0), users in the owner's project can read and execute the file (5), and all other users cannot access the file (7). However, the system standard may be changed by the individual installations.

Command Format

```
PROTECT file1 <nnn>, file2 <nnn>, file3 <nnn>,...
```

The protection can be specified before the filename in which case it is the default for subsequent files until changed. The full wildcard construction can be used for either the filename or the extension.

Characteristics

The PROTECT command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

[1] This command runs the COMPIL program, which interprets the command before running PIP.

Associated Messages

Refer to Chapter 4.

Examples

```
.PROTECT FORM.*<157> ↵
.PRO MAIN.MAC<123>, <456>EQUIL.CBL,ADD.ALG↵
```

QUEUE command

Function

The QUEUE command allows the user to make entries in several system queues - the input queue for the Batch system, and the output spooling queues for the line printer, the card punch, the paper-tape punch, and the plotter. The QUEUE command also provides the means of obtaining listings of the entries in the queues.

Command Formats

1. QUEUE INP: jobname = control file specification, log file specification

   To make an entry in the Batch input queue, INP:.

2. QUEUE output queue name: jobname = list of input specifications

   To make an entry in an output spooling queue.

3. QUEUE listing file specifications/LIST = list of queue names

   To obtain a listing of the entries in a queue.

4. The following six commands can be substituted for the various formats of the QUEUE command:

   a. CPUNCH jobname = list of input specifications
      equivalent to QUEUE CDP: jobname = list

   b. PLOT jobname = list of input specifications
      equivalent to QUEUE PLT: jobname = list

   c. PRINT jobname = list of input specifications
      equivalent to QUEUE LPT: jobname = list

   d. PUNCH jobname = list of input specifications[1]
      equivalent to QUEUE PTP: jobname = list

   e. SUBMIT jobname = control file name, log file name
      equivalent to QUEUE INP: jobname = control file, log file

   f. TPUNCH jobname = list of input specifications
      equivalent to QUEUE PTP: jobname = list

Queue names are taken from the following list:

INP: (I:)	Batch input queue
LPT: (L:)	line printer output queue, default condition
CDP: (C:)	card punch output queue
PTP: (PT:)	paper-tape punch output queue
PLT: (PL:)	plotter output queue

[1] The PUNCH command can be redefined by the installation to be equivalent to QUEUE CDP: jobname = list.

Command Formats (cont)

Control file specification is the file specification, plus switches and keyword parameters, for the control file being submitted to the Batch input queue. This file can be on any file structure that the user has access to; the default is DSK:. The filename is required, but the extension can be omitted; the default is .CTL. The asterisk construction is legal for the filename or extension.

Log file specification is the file specification for the file that is to be used to record actions taken during the execution of the control file. This file can be on any file structure in which the user can write. The default is the same file structure in which the control file resides. If the filename is missing, the log file is given the same name as the control file. If the extension is omitted, it is .LOG.

Jobname is the name of the job being entered into the queue. The default jobname is the name of the first file in the request not the first file given. These names are different when the first file given does not yet exist.

Input specifications are the file specifications for the disk files to be processed, and the various switches and keyword parameters that aid in constructing the queue entry. The files can be on any file structure that the queue processor has access to; the default is DSK:. The files can be in any directory, provided that the user has read-access to them; the default is the user's directory. The filename is optional; the default is * for the first filename. The default for subsequent filenames is the last filename used. Note that the asterisk construction is legal only in the input specifications. The extension can be omitted because each queue has a default extension for the files to be processed. These default extensions are:

.CTL	-	Batch input queue
.LPT	-	line printer queue
.CDP	-	card punch queue
.PTP	-	paper-tape punch queue
.PLT	-	plotter queue

The listing file specification is the description of the listing file. The default for the listing file destination is TTY unless a name is specified. If no queue names are specified, all queues for all the jobs of all users are listed.

Switches - Three categories of switches are provided. The first category contains the switches that define the operation; the second contains the switches that can appear only once because they affect the entire request; the third contains the switches specific to each file. In general, switches that precede the filename become the default for all succeeding files. This is true also for a device name, an extension, or a directory name that precedes a filename.

Queue-Operation Switches - Only one of this type of switch can be placed in a command string, because these switches define the type of queue request. This switch may appear anywhere in the command string.

QUEUE command (Cont)

Command Formats (cont)

General Queue Switches - Each of these switches can appear only once in a command string. They affect the entire request, generally in terms of scheduling. These switches can appear anywhere in the command string.

File-Control Switches - These switches affect the individual files in a request and must be adjacent to the filename in the command string. In order to change the defaults for the rest of the files, however, these switches must appear before a filename.

In the table of switches below, the following conventions have been used:

ALL	- Switches that can appear for both the Batch input queue and the output queues.
INPUT	- Switches that can appear only for the Batch input queue.
LIST	- Switches that can appear only for the listing file specification.
OUTPUT	- Switches that can appear only for the output queues.

Switch	Meaning	Category	Queues
/AFTER:t	Process the request after the specified time. t is either in the form hhmm (time of day) or +hhmm (time later than the current time). The resulting AFTER time must be less than the DEADLINE time. If the switch, or the value of the switch, is omitted, no AFTER constraints are assumed.	General	ALL
/BEFORE:t	Queue only the files with creation dates before time t where t = dd-mmm-yy hhmm.	General	OUTPUT
/BEGIN	Start the output on the nth page, card, or foot. The default is to begin output on the first unit.	File Control	OUTPUT
/CARDS:n	Use n (decimal) as the maximum number of cards that can be punched by the job. If the switch is omitted, no cards are punched. If the switch is given with no value, 2000 cards is assumed as the maximum.	General	INPUT

(continued on next page)

Command Formats (cont)

Switch	Meaning	Category	Queues
/CHARGE:a[1]	Charge the run to the specified account.	General	ALL
/COPIÈS:n	Repeat the output the specified number of times (n must be less than 64). The default is one copy. If more than 63 copies are desired, two requests must be made.	File Control	OUTPUT
/CORE:n	Use n (in decimal K) as the maximum amount of core memory that the job can use. If the switch is omitted, the maximum of 25K is assumed; if the value of the switch is omitted, a maximum of 40K is assumed.	General	INPUT
/CREATE	Make a new entry in the specified queue. This switch is the default for the queue-operation switches.	Queue Operation	ALL
/DEADLINE:t	Process the request before the specified time. t is either in the form hhmm (time of day) or +hhmm (time later than the current time). The resulting DEADLINE time must be greater than the /AFTER time. If the switch, or the value of the switch is omitted, no DEADLINE constraints are assumed.	General	ALL
/DEFER[1]	Make a new entry in the specified queue, but the request is deferred until LOGOUT.	Queue Operation	ALL

(continued on next page)

[1]Not yet implemented.

QUEUE command (Cont)

Command Formats (cont)

Switch	Meaning	Category	Queues
/DEPEND:n	Specifies the initial value of the dependency count in decimal. When used with /MODIFY, this switch changes the dependency count of another job. If n is a signed number (+ or -), that number is added to or subtracted from the dependent job's count. If n is not a signed number, the dependent job's count is changed to n. If this switch is omitted, no dependency is assumed.	General	INPUT
/DISPOSE:DELETE	Delete the file after spooling.	File Control	ALL
/DISPOSE:PRESERVE	Save the file after spooling. This is the default for files with extensions of .LST, .TMP, and if protection is 0xx, .CDP, .LPT, .PLT, .PTP.	File Control	ALL
/DISPOSE:RENAME	Rename the file from the specified directory immediately, remove it from the logged-out quota, and delete it after spooling. This is the default for files with extensions of .LST, .TMP, and if protection is 0xx, .CDP, .LPT, .PLT, .PTP.	File Control	ALL
/F	List the entries in the queue, but do not update the queues. Therefore, the list may not be an up-to-date listing of the queues but the listing will be faster than with /LIST.	Queue Operation	LIST
/FEET:n	Use n (in decimal) as the maximum number of feet of paper tape that the job can punch. If the switch is omitted, no paper tape is punched. If the value is omitted, the default is 10*B+20 feet, where B is the number of blocks in the request.	General	INPUT

(continued on next page)

Command Formats (cont)

Switch	Meaning	Category	Queues
/FILE:ASCII	Specify that the input file format is to be interpreted as ASCII text. This is assumed for all files with extensions other than .DAT.	File Control	OUTPUT
/FILE:COBOL	Specify that the input file format is to be interpreted as COBOL SIXBIT text.	File Control	OUTPUT
/FILE:ELEVEN	Specify that the input file format is to be interpreted as binary format.	File Control	OUTPUT
/FILE:FORTRAN	Specify that the input file format is to be interpreted as FORTRAN ASCII text (obeys FORTRAN carriage control characters). This is assumed for files with an extension of .DAT.	File Control	OUTPUT
/FORMS:a	Place the output on the named forms. The argument to the switch must be six alphabetic characters. Normal forms (14 x 11) are used if this switch is omitted. Narrow forms are 8-1/2 x 11.	General	OUTPUT
/HEADER:0 or 1	Output block headers at beginning of the file if 1 (default); do not output headers if 0.	File Control	OUTPUT
/HELP	Print a message giving the general format of the command string and explains the dialogue that is entered if the user needs additional help.	-	-
/KILL	Remove the specified entry from the specified queue. This switch requires an output specification; it does not default to LPT:* The /KILL switch can be used for deleting a previously submitted request as long as the request has not been started.	Queue Operation	ALL

(continued on next page)

QUEUE command (Cont)

Command Formats (cont)

Switch	Meaning	Category	Queues
/LIMIT:n	Limit the output to the specified number of pages, cards, feet, or minutes.	General	OUTPUT
/LIST	List the specified entries in the queue; the default entries are those for queues for all the jobs of all users.	Queue Operation	LIST
/LOG	Define the file that the spoolers will use to record their output. The default is jobname.LOG.	File Control	OUTPUT (LPT)
/MODIFY	Alter the specified parameters in the specified jobs; this switch requires that the user have access rights to the job. It also requires a queue name; it does not default to the LPT. This switch can be used to modify a previously submitted request as long as the request has not been started.	Queue Operation	ALL
/NEW	Accept request even if file does not yet exist. This is the default for the log file of Batch input queue.	File Control	ALL
/NOTE:a	Output the specified text (a) in the output.	File Control	OUTPUT
/NULL	Accept request even if there is nothing in the request. No error message is given.	General	OUTPUT
/OKBINARY	Print files whose extensions include binary information. Normally files with extensions .SAV, .SHR, .LOW, .REL, and .HGH will not be in print queues.	File Control	OUTPUT (LPT)
/OKNONE	Do not produce message if no files match the wildcard construction.	File Control	OUTPUT

(continued on next page)

Command Formats (cont)

Switch	Meaning	Category	Queues
/OUTPUT:n	Cause job to terminate with a /Z:n to KJOB (n is from 0 to 4). N=0 Suppress all normal queuing performed at LOGOUT time. N=1 Queue only the log file. N=2 Queue only the log file and spooled output (e.g., *.LPT). N=3 Queue the log file, spooled output, and *.LST files. N=4 Queue the log file, spooled output, *.LST files, and any requests deferred to LOGOUT time.	General	INPUT
/PAGE:n	Use n (decimal) as the maximum number of pages of output that the job can print. If the switch is omitted, the maximum is 200 pages; if only the value is omitted, a maximum of 2000 pages can be printed.	General	INPUT
/PAPER:x	Identical to /PUNCH:x, /PRINT:x, /TAPE:x, or /PLOT:x.	File Control	OUTPUT
/PHYSICAL	Suppress logical device names for the specified device.	File Control	ALL
/PLOT:ASCII	Plot the file in ASCII mode. If the /PLOT switch is omitted, the file is plotted in the data mode specified in the file.	File Control	OUTPUT (PLT)
/PLOT:BINARY	Plot the file in binary mode. If the /PLOT switch is omitted, the file is plotted in the data mode specified in the file.	File Control	OUTPUT (PLT)

(continued on next page)

QUEUE command (Cont)

Command Formats (cont)

Switch	Meaning	Category	Queues
/PLOT:IMAGE	Plot the file in image mode. If the /PLOT switch is omitted, the file is plotted in the data mode specified in the file.	File Control	OUTPUT (PLT)
/PRINT:ARROW	Convert all control characters to up-arrow format except 011-015 and 020-024. This is the default.	File Control	OUTPUT (LPT)
/PRINT:ASCII	Send the file to the line printer with no changes.	File Control	OUTPUT (LPT)
/PRINT:OCTAL	Print the file in octal.	File Control	OUTPUT (LPT)
/PRINT:SUPPRESS	Suppress all character-control characters except for ASCII code characters LF and CR; this switch implies the use of the /PRINT:ARROW. Equivalent to operator command to spooler (SUPPRESS).	File Control	OUTPUT (LPT)
/PRIORITY:n	Give the specified external priority (n = 0 to 62) to the request. A larger number is greater priority. The default is 10 if no switch is given, and 20 if a switch is given without the value.	General	ALL
/PROTECT:nnn	Specify a protection nnn (in octal) for this job or queue entry. If the switch, or the value of the switch, is omitted, the standard protection is assumed.	General	ALL
/PUNCH:026	Punch files in 026 Hollerith code. If the /PUNCH switch is not given, the files are punched according to the data mode of the file.	File Control	OUTPUT (CDP)
/PUNCH:ASCII	Punch files in ASCII card code. If the /PUNCH switch is not given, the files are punched according to the data mode of the file.	File Control	OUTPUT (CDP)

(continued on next page)

Command Formats (cont)

Switch	Meaning	Category	Queues
/PUNCH:BINARY	Punch files in binary card format. If the /PUNCH switch is not given, the files are punched according to the data mode of the file.	File Control	OUTPUT (CDP)
/PUNCH:D029	Punch files in the old DEC 029 card code. If the /PUNCH switch is not given, the files are punched according to the data mode of the file.	File Control	OUTPUT (CDP)
/PUNCH:IMAGE	Punch files in image mode. If the /PUNCH switch is not given, the files are punched according to the data mode of the file.	File Control	OUTPUT (CDP)
/REMOVE	Remove the file from the queue. This switch is valid only with the /MODIFY switch and can be used to remove a previously submitted file as long as the Batch System has not started processing the job.	File Control	OUTPUT
/REPORT:code	Print the specified report within a COBOL report file. Code can be up to 12 characters in length.	File Control	OUTPUT (LPT)
/RESTART:0 or 1	A value of 0 (default) means the job cannot be requeued or restarted by the operator after a system crash. A message is sent to the job's log file. A value of 1 means the job will be requeued or restarted. The job should not be restartable if there are changes to the permanent file directory.	General	INPUT
/SEQ:n	Specify a sequence number to aid in identifying a request to be modified or deleted.	General	ALL
/SINCE:t	Queue only the files with creation dates after the specified time t where t is in the form dd-mmm-yy hhmm.	General	OUTPUT

(continued on next page)

QUEUE command (Cont)

Command Formats (cont)

Switch	Meaning	Category	Queues
/SPACING:DOUBLE	Double-space the output lines.	File Control	OUTPUT (LPT)
/SPACING:SINGLE	Single-space the printed lines (default).	File Control	OUTPUT (LPT)
/SPACING:TRIPLE	Triple-space the printed lines.	File Control	OUTPUT (LPT)
/START:n	Start on n line of the file. If the switch, or the value of the switch, is omitted, the Batch System starts with the first line.	File Control	ALL
/STRS	Search for the file on all structures in the search list and takes each occurrence. The default is to take just the first occurrence of the file.	File Control	OUTPUT
/TAPE:ASCII	Punch the tape in ASCII code. If the /TAPE switch is not given, the files are punched according to the data mode of the file.	File Control	OUTPUT (PTP)
/TAPE:BINARY	Punch the tape in binary mode. If the /TAPE switch is not given, the files are punched according to the data mode of the file.	File Control	OUTPUT (PTP)
/TAPE:IBINARY	Punch the tape in image-binary mode. If the /TAPE switch is not given, the files are punched according to the data mode of the file.	File Control	OUTPUT (PTP)
/TAPE:IMAGE	Punch the tape in image mode. If the /TAPE switch is not specified, the files are punched according to the data mode of the file.	File Control	OUTPUT (PTP)
/TIME:hhmmss	Specify the central processor time limit for the job. If no switch is specified, the limit is 5 minutes; if the switch is specified without a value, the limit is 1 hour.	General	INPUT

(continued on next page)

Command Formats (cont)

Switch	Meaning	Category	Queues
/TPLOT:n	Use n (decimal minutes) as the maximum amount of plotting time allowed for the job. If the switch is omitted, no plotter time is allowed; if the value is omitted but the switch is given, the maximum plotter time is 10 minutes.	General	INPUT
/UNIQUE: 0 or 1	Run any number of Batch jobs under this project-programmer number at the same time, if 0. Runs only one Batch job at any one time, if 1 (default).	General	INPUT
/UNPRESERVED	Output file only if not preserved.	General	OUTPUT
/ZDEFER[1]	Create a new entry in a queue and defer it until LOGOUT; however, the deferred file is zeroed first so that all previous /DEFER requests from the current job are deleted.	Queue Operation	ALL

Characteristics

The QUEUE command (and its associated variations):

Leaves the terminal in monitor mode.
Runs the QUEUE program, thereby destroying the user's core image.
Does not require LOGIN when only queue listings are desired.
Depends on FTQCOM which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

[1]Not yet implemented.

QUEUE command (Cont)

Examples

`.QUEUE FILEA,FILEB)`	Enter files FILEA.LPT and FILEB.LPT in the line-printer queue under the jobname of FILEA.
`.QUEUE INP:=TEST)`	Enter file TEST.CTL in the Batch input queue under jobname TEST and log file with name TEST.LOG.
`.QUEUE IN:PAYR=MAN)`	Enter file MAN.CTL in the Batch input queue under jobname PAYR and log file with name MAN.LOG.
`.QUEUE DSK:A.X=/LIST)`	Place a queue listing of all jobs into file A.X in the user's disk area.
`.QUEUE INP:FREED=FILEA/CREATE /PRIORITY:4/TIME:1:5)`	Place file FILEA.CTL in the Batch input queue with the jobname FREED. An external priority of 4 and CPU time limit of one minute and five seconds are set for the job. The log file is named FILEA.LOG.
`.QUEUE INP:TEST=/KILL)`	Remove the entry corresponding to TEST.CTL from the Batch input queue.
`.QUEUE INP:JOBNAM=/MODIFY/TIME:200)`	Alter the time parameter of the entry corresponding to JOBNAM.CTL in the Batch input queue.
`.QUEUE INP:=JOB.CTL/PAGES:500/TPLOT:20)`	Establish a limit of 500 pages and 20 minutes of plotting on the output generated by this job.
`.QUEUE PLT:=JOB.PLT/LIMIT:20)`	Queue a file to PLTSPL with a limit of 20 minutes of plotting time.

QUOLST program

Function

The QUOLST program informs the user of both the amount of disk space he has used and the amount he has left on each file structure in his search list. This program also returns the amount of free space that the system has left for all users of the structure. Free system space on structures not in the user's search list is not output. This information can be obtained by typing SYSTAT /F.

The output given for each file structure consists of 1) the structure name, 2) the number of blocks used, and 3) the numbers of blocks left in the logged-in quota, in the logged-out quota, and on the structure.

Command Format

R QUOLST

Characteristics

The R QUOLST command:

Leaves the terminal in monitor mode.
Runs the QUOLST program, thereby destroying the user's core image.

Examples

```
.R QUOLST

USER:    27,400
STR      USED   LEFT:(IN)  (OUT)    (SYS)
DSKA:       0     1000      100     4703
DSKB:     491     9509     4509     4240
DSKC:       0    10000     5000      396
```

```
.R QUOLST

USER:    31,50
STR      USED   LEFT:(IN)  (OUT)    (SYS)
DSKA:    1022      -22     -922     4215
DSKB:    1735    78265     8265       36
DSKC:       0     2000     1000     6378
```

The user is over quota on DSKA: and must delete files before he can logout.

R command

Function

The R command loads a core image from the system device and starts it at the location specified within the file (.JBSA). It is equivalent to RUN SYS: file.ext core and is the usual way to run a system program that does not have a direct monitor command to run it.

This command clears all of user core. However, programs should not count on this action and should explicitly clear those areas of core that are expected to contain zeroes (i.e., programs should be self-initializing). This action allows programs to be restarted by a ↑C, START sequence without having to do another R command.

On magnetic tape, if the low or high segment is missing, a null record is output before the EOF for the missing segment so that two EOFs cannot occur consecutively. Therefore, a saved null segment does not appear as a logical EOT (2 EOFs in a row).

Command Format

R file.ext core

Arguments are the same as in the RUN command except that SYS: is used as the default device. (In nondisk monitors, the default is the generic name that matches the system device.) Refer to the RUN command for a discussion of the core argument.

The extension applies to the low file, not the high file. An extension of .SHR, then .HGH, is assumed for the high file. If the user types an extension of .SHR or .HGH, the extension is treated as a null extension since .SHR and .HGH are confusing as low file extensions.

Characteristics

The R command:

Places the terminal in user mode.
Runs a system program.

Associated Messages

Refer to Chapter 4.

Examples

```
.R PIP↲
*
.R PIP 5↲
*
```

REASSIGN command

Function

The REASSIGN command allows one job to pass a device to a second job without having the device go through the monitor device pool (restricted or unrestricted). Both restricted and unrestricted devices can be reassigned. This command, applied to DECtapes, clears the copy of the directory currently in core, forcing the next directory reference to read a new copy from the tape, but does not clear the logical name assignment. If a device is INITed, a RELEASE UUO is performed unless the user issuing the command is reassigning the device to himself.

Command Format

REASSIGN dev job

dev = the physical or logical name of the device to be reassigned. This argument is required.

job = the number of the job to which the device is to be reassigned. If no job is specified, the device is reassigned to the job issuing the command. This is useful when the user wants to force the next directory reference to come from the tape instead of core.

A logical name which is also a physical name can be reassigned only if the job issuing the command and the job to which the device is to be reassigned have the same project-programmer number, or the user issuing the command has operator privileges (logged-in under [1,2] or logged-in at OPR). However, a logical name cannot be duplicated; i.e., two devices cannot have the same logical name.

Characteristics

The REASSIGN command:

Leaves the terminal in monitor mode.
Requires core.
Does not operate when the device is currently transmitting data.

Restrictions

The job's controlling terminal cannot be reassigned.

REASSIGN command (Cont)

Associated Messages

Refer to Chapter 4.

Examples

```
.REASSIGN LPT:17)                    Reassign the line printer to job 17.
.
.REASSIGN CDP:4)                     Reassign the card punch to job 4.
.
```

REATTA program

Function

The REATTA program allows a user to transfer his job from one terminal to another. Unlike the ATTACH command, REATTA does not require a password or that the terminal be of the same type that LOGIN recognizes in order to run the job. For example, usually a [1,2] job can run only on a local terminal. However, the REATTA program can be used to attach a [1,2] job from a local terminal to a remote terminal.

Before reattaching his job, the user should verify that the terminal to which he is attaching is turned on and working properly. Otherwise, it might be difficult to retrieve the job.

Command Format

```
.R REATTA ↵
```

REATTA responds by asking for the new terminal name.

```
TYPE NEW TTY NAME:
```

The user answers with either the new terminal name (e.g., CTY, TTY2) or number (e.g., 2). REATTA then responds with

```
FROM JOB n
.
```

on the old terminal, and

```
NOW ATTACHED TO JOB n
.
```

on the new terminal.

Characteristics

The R REATTA command:

Leaves the terminal in monitor mode.
Runs the REATTA program, thereby destroying the user's core image.

Restrictions

The R REATTA command is not available to Batch users.

REATTA program (Cont)

Associated Messages

Refer to Chapter 4.

Examples

```
.R REATTA)
TYPE NEW TTY NAME: TTY27)
FROM JOB 7                        ;appears on old terminal
.
NOW ATTACHED TO JOB 7             ;appears on TTY 27
.
```

REENTER command

Function

The REENTER command is similar to the DDT command. It copies the saved program counter value from .JBPC into .JBOPC and starts the program at an alternate entry point specified in .JBREN (must be set by the user or his program). If the job was executing a UUO when it was interrupted (i.e., in exec mode but not in TTY input wait or SLEEP mode), the monitor continues the job until the UUO is completed and then traps to the REENTER address in .JBREN. If the job is in TTY input wait or SLEEP mode, the trap to the REENTER address occurs immediately and .JBOPC contains the address of the UUO. If the job is in user mode, the trap also occurs immediately. Therefore, it is always possible to continue the interrupted program after trapping by executing a JRSTF@.JBOPC.

Command Format

REENTER

Characteristics

The REENTER command:

Places the terminal in user mode.
Requires core.
Requires the user to have a job number.

Associated Messages

Refer to Chapter 4.

Example

```
.REE
```

RENAME command[1]

Function

The RENAME command changes the name of one or more files on disk or DECtape.

Command Format

RENAME arg

arg = a pair of file specifications separated by an equal sign, or a string of such pairs separated by commas:

```
RENAME new1 = old1, new2 = old2,...
```

Device or file structure names can be specified only with the new filename and remain in effect until changed or until the end of command string is reached. In addition, a protection may be specified with the new filename and remains in effect only for that filename. This command accepts the full wildcard construction.

Characteristics

The RENAME command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

[1] This command runs the COMPIL program, which interprets the command before running PIP.

Example

```
.RENAME T11.MAC=T1.MAC↲
FILES RENAMED:
T1.MAC

.RENAME *.BAK=*.MAC↲
FILES RENAMED:
T11.MAC
T2.MAC
T3.MAC

.RENAME TEST.MAC<Ø57>=TEST.MAC↲
FILES RENAMED:
TEST.MAC

.
```

RESOURCES command

Function

The RESOURCES command prints the names of all available devices (except TTY's and PTY's), all file structures, and all physical units not in file structures (unless they are down or non-existent).

Command Format

RESOURCES

Characteristics

The RESOURCES command:

Leaves the terminal in monitor mode.
Does not require LOGIN.

Example

```
.RES)
DSKA,DSKB,DSKC,DPB0,DPB1,CDR0,2,PTR0,LPT0,1,2,3,DTA0,3,4,5,6,7,MTA0,1,2,
PTP0,CDP0,PLT0,DIS0
```

RESTORE program

Function

The RESTORE program enables the user to place back onto disk that which was saved on the backup medium (magnetic tape, disk, or DECtape) with the BACKUP program. This includes restoring the entire disk or a subset of the disk. The data to be returned to the disk is read from the BACKUP SET file. This file contains the data that was saved with one BACKUP command. On a restore, either the BACKUP SET file can be scanned for the desired files or the index file can be searched to determine where the requested data is stored within the BACKUP SET file. The index file contains the directories of all areas written on the backup medium along with the relative block number in the BACKUP SET file where each file begins. When the entire backup medium is being restored, the RESTORE program starts at the beginning of the index file and continues until it reaches the last file in the index.

During a restore, a command recovery file is created that contains information concerning the portion of the user's command that has been executed and the portion that is remaining. This file resides on the disk and is updated as portions of the user's request are completed. The command recovery file is valuable if the system fails because only part of the restore need be redone.

As files are restored to disk, UFDs are created for each file structure on which the user has files. These newly created UFDs are then entered into the MFD.

Command Format

R RESTORE

The following commands may be typed by the user after the RESTORE program outputs a slash. These commands are stored in core and are not processed until the START command is given. The full wildcard construction may be used to replace the filename or the extension (refer to Paragraph 1.4.2.4).

Command	Explanation
BACKSPACE FILE	Backspaces the magnetic tape to a user file header and positions the tape before the header.
BACKSPACE SET	Backspaces the magnetic tape to a BACKUP header and positions the tape either before the header or to the beginning of the tape if there is no BACKUP header (i.e., there is only one BACKUP set on the tape).
BACKSPACE UFD	Backspaces the magnetic tape to a UFD header and positions the tape immediately before the header.

(continued on next page)

RESTORE program (Cont)

Command Format (cont)

Command	Explanation
DELETE dev:file.ext	Deletes the named file from the designated device. This device must be one on which a BACKUP has been done.

DENSITY MTAn:x	Sets the magnetic tape density as specified by x.
	x = 2 200 bpi
	x = 5 556 bpi
	x = 8 800 bpi
	The default is the system standard defined at MONGEN time.

DUMP ON dev:file.ext	Dumps the contents of the BACKUP set file beginning at the present position and ending at the next file control word. All types of errors are ignored. The device on which a dump is to be written may not be a listing device.
ERROR DUMP/switch	Returns to the last file control word and dumps the file if a transmission error of the type specified has occurred. /switch = any or all of the following /CHECKSUM /PARITY /READ /WRITE
ERROR HALT/switch	Halts program execution if the type of error specified by /switch occurs during restoring. An asterisk is typed to the user so that he may type further instructions. The user may want to backspace the tape and dump it. The command recovery file is destroyed unless simply a START command is given. In this case, the current file is skipped and the next command in the recovery file is executed. /switch = any or all of the following: /CHECKSUM /EXCEPT /PARITY /READ
INDEX dev:file.ext	Reads the index file with the designated filenames from the device specified.

(continued on next page)

Command Format (cont)

Command	Explanation
LOG dev:file.ext /switch	Writes a file on the specified device so that operations of the RESTORE program can be recorded. The default is DSK:RESTOR.LOG. /switch = /ERROR Logs only the errors. If this switch is omitted, all operations are recorded.
PARITY dev: ODD or EVEN	Specifies the parity on magnetic tape as odd or even. The default is odd.
RESTORE dev1: [p,p] file.ext ← dev2/switch	Writes the specified files from dev2 to the designated area on dev1. For example, if all of the disk is to be restored from the entire BACKUP SET file, the command is RESTORE DSK: ← MTA1: /switch = /EXCEPT file descriptor Indicates the files that should not be restored.
REWIND dev:	On magnetic tape, closes BACKUP set and rewinds tape. On disk, closes BACKUP set.
SET ACCESS dd-mmm-yy	Sets the access date to be used when restoring files. The files will be restored only if accessed after this date.
SET CREATION dd-mmm-yy	Sets the creation date to be used when restoring files. The files will be restored only if created after this date.
SKIP dev: FILE	Advances to next file trailer, EOF1, and positions after it.
SKIP dev: SET	Advances to next BACKUP set trailer and positions after it, or skips to end of the tape and positions the tape between the tape marks.
SKIP dev: UFD	Advances to next UFD header, HDR1, and positions before the header.
START	Begins execution of a series of commands entered previously. Commands are not processed until a START command. If there are no commands to be processed when the START is executed, the command recovery file is searched for an executable command.
UNLOAD dev:	Performs a rewind and unload to the magnetic tape.

RESTORE program (Cont)

Command Format (cont)

The RESTORE program may be restarted by the user at any time. The user can cancel current requests and specify new ones with a ↑C ↑C START sequence. The command recovery file is deleted with a ↑C START sequence unless the next command given to RESTORE is a START. If this is the case, the command recovery file is searched and the RESTORE program proceeds according to the information in the file. A ↑C CONT sequence does not delete the command recovery file; this sequence of commands completes the current requests.

Upon completion of all requests, the RESTORE program closes the log file and types an asterisk on the user's terminal indicating its readiness for more requests. Rewinds to the magnetic tape due to it being filled are actually rewind and unload operations to ensure that the magnetic tape is not overwritten. When the RESTORE program reaches completion, the magnetic tape last written on remains in position unless a REWIND command is given.

Characteristics

The R RESTORE command:

Runs the RESTORE program, thereby destroying the user's core image.

Associated Messages

Refer to Chapter 4.

REWIND command [1]

Function

The REWIND command rewinds a magnetic tape or a DECtape. This command is equivalent to the PIP command string:

```
dev: (MW)←
```

Command Format

REWIND dev:

dev: = a magnetic tape (MTAn) or a DECtape (DTAn).

Characteristics

The REWIND command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.REW DTA4:)
.REWIND MTA1:)
```

[1] This command runs the COMPIL program, which interprets the command before running the PIP program.

RUN command

Function

The RUN command loads a core image from a retrievable storage device and starts at the location specified within the file (.JBSA).

If the program has two segments, both the low and high segments are set up. If the high file has extension .SHR (as opposed to .HGH), the high segment will be shared. Therefore, if the user has RUN (or GET) the same program, I/O will not usually be required for the high segment. A two-segment program may have a low file extension (.LOW).

The RUN command clears all of user core. However, programs should not count on this action and should explicitly clear those areas of core that are expected to contain zeroes (i.e., the programs should be self-initializing). This action allows programs to be restarted by a ↑C, START sequence without having to do another RUN command.

On magnetic tape, if the low or high segment is missing, a null record is output before the EOF for the missing segment so that two EOFs cannot occur consecutively. Therefore, a saved null segment does not appear as a logical EOT (2 EOFs in a row).

Command Format

RUN dev:file.ext [proj,prog] core

dev: = the logical or physical name of the device containing the core image. The default device name is DSK:.(In nondisk monitors, the default is the generic name that matches the system device.)

file.ext = the name of the file containing the core image; .ext applies to the low file, not the high file. An extension of .SHR, then .HGH, is assumed for the high file. If the user types an extension of .SHR or .HGH, the extension is treated as a null extension since .SHR and .HGH are confusing as low file extensions. The default filename is the job's current name as set by the last R, RUN, GET, SAVE, or SSAVE command, the last SETNAM UUO, or the last command which ran a program.

[proj,prog] = the project-programmer number; required only if core image file is located in a disk area other than the user's.

core = the amount of core to be assigned to the sum of the low and high segments if different from minimum core needed to load the program or from the core argument of the SAVE command which saved the file.

If core < the minimum low segment size, then an error message occurs.

If core ≥ the minimum low segment size and < the sum of the high segment and the minimum low segment size, then the core assignment is the low segment size.

If core ≥ the sum of the minimum low segment and the high segment size, then the core assignment is the size of both the low and high segments to be used.

Command Format (cont)

Core arguments can be specified in units of 1024 words or 512 words (a page) by following the number with K or P, respectively. For example, 2P represents 2 pages or 1024 words. If K or P is not specified, K (1024 words) is assumed.

Note that on KA10 based systems (DECsystem-1040, 1050, 1055), the minimum unit of allocation is 1024 words. Therefore, all arguments are rounded up to the nearest multiple of 1024 words (e.g., 3P is treated as 2K on a KA10 based system).

Since previous core is returned, MTA must have the core argument because there is no directory telling how much core is for the low segment. Refer to Appendix D.

Characteristics

The RUN command:

Places the terminal in user mode.

Restrictions

On systems with a large amount of core memory, the user should not specify a core argument that forces the high segment to start higher than 400000 (i.e., a core argument of greater than 128K) unless the program's high segment is location independent. If this is done, the ILLEGAL UUO error message is likely to occur.

Associated Messages

Refer to Chapter 4.

Examples

```
.RUN TEST↵
.RUN HISTST [10,63]↵
.RUN DTA3:TEST1↵
```

SAVE command

Function

The SAVE command writes out a core image of the user's core area on the specified device. It saves any user program (two-segment sharable, one-segment nonsharable, or two-segment nonsharable) as one or two files. Later, when the program is loaded by a GET, R, or RUN command, it will be nonsharable. If DDT was loaded with the program, the entire core area is written; if not, the area starting from zero up through the program break (as specified by .JBFF) is written. Refer to DECsystem-10 Monitor Calls for a description of the job data area locations referenced by this command.

The SAVE command should be used instead of the SSAVE command when debugging a two-segment program. Refer to Appendix D for additional information on the SAVE command.

On magnetic tape, if the low or high segment is missing, a null record is output before the EOF for the missing segment so that two EOFs cannot occur consecutively. Therefore, a saved null segment does not appear as a logical EOT (2 EOFs in a row).

Command Format

SAVE dev:file.ext [proj,prog] core

dev = the device on which the core image file is to be written. The default device name is DSK:. In nondisk monitors, the default is the generic name that matches the system device. The colon following the device name is required if a device is specified.

file.ext = the name to be assigned to the core image file. The default filename is the job's current name as set by the last R, RUN, GET, SAVE, or SSAVE command, the last command which ran a program (e.g., DIRECT), or the last SETNAM UUO.

ext applies to the low file, not the high file. An extension of .SHR, then .HGH, is assumed for the high file. If the user types an extension of .SHR or .HGH, the extension is treated as a null extension since .SHR and .HGH are confusing as low file extensions. If ext is omitted and the program has only one segment, the ext is assumed to be .SAV. If ext is omitted and the program has two segments, the high segment will have extension .HGH, and the low segment will have extension .LOW.

[proj,prog] = the name of the disk area on which the core image file is to be written.

core = the amount of core in which the program is to be run. This value is stored in JOBDAT as the job's core area (.JBCOR) and is used by subsequent RUN and GET commands. This argument is optional.

Core arguments can be specified in units of 1024 words or 512 words (a page) by following the number with K or P respectively. For example, 2P represents 2 pages or 1024 words. If K or P is not specified, K (1024 words) is assumed.

Command Format (cont)

Note that on KA10 based systems (DECsystem-1040, 1050, 1055), the minimum unit of allocation is 1024 words. Therefore, all arguments are rounded up to the nearest multiple of 1024 words (e.g., 3P is treated as 2K on a KA10 based system).

If core is omitted, only the number of blocks required by the core image area (as explained in the RUN command description) is assumed.

Characteristics

The SAVE command:

Leaves the terminal in monitor mode.
Requires core.
Does not operate when a device is currently transmitting data.

Associated Messages

Refer to Chapter 4.

Example

```
.SAVE ⮐
JOB SAVED

.SAVE DTA3:TEST⮐
JOB SAVED
```

SCHED command

Function

The SCHED command types out the schedule bits as set by the last privileged SET SCHED command. The schedule bits are as follows:

0	regular timesharing.
1	no further logins allowed except from CTY.
2	no further logins from remote terminals, and no answering of data sets.
4	batch jobs only.
100	device mounts can be done without operator intervention.
200	unspooling allowed.
400	no operator coverage.

Command Format

SCHED

Characteristics

The SCHED command:

Leaves the terminal in monitor mode.
Does not require LOGIN.
Depends on output from the SET SCHED command which is normally absent in the DECsystem-1040.

Example

```
.SCHED)
000400
```

Regular timesharing, but no operator coverage.

SEND command

Function

The SEND command provides a mechanism for one-way interconsole communication. (This command replaces the TALK command.) A line of information is transmitted from one terminal to another, with the identification of the terminal sending the information. With remote communications capabilities, SEND is able to differentiate between stations.

When the SEND command is sent from the central station operator's terminal (OPR) or from a terminal logged in as [1,2], it allows a broadcast of a line of information to all non-slaved terminals (including remote terminals) in the system. This allows important information to be dispersed, such as system shutdown or hardware problems. SEND ALL messages do not go to slaved terminals unless the SET TTY NO GAG bit is set to permit reception when the terminal is busy.

A busy test is made on single-destination messages before the message is sent unless the sender or the receiver of the message is OPR or a job logged-in as [1,2]. The receiver of the message is considered busy if his terminal is not at monitor command level. If the receiver is busy, the sender receives the message BUSY and the information is not sent, unless the receiving terminal has the TTY NO GAG bit set (refer to the SET TTY command). If the receiving terminal is turned off, the information appears to have been sent, since the hardware cannot detect this condition on hard-wired terminals.

Command Format

SEND dev: text

or

SEND JOB n text

dev = any physical terminal name (CTY included) or OPRnn. If OPRnn is specified, the message is sent to the operator at station nn. If OPR (nn is null) is specified, the message is sent to the operator at the user's logical station. If the terminal sending the message is the operator's terminal, the argument may be ALL to provide the broadcast operation.

n = the job number to which the message is to be sent.

The message printed on the receiving terminal appears as follows:

```
;;TTY n: - text
```

where

n is the TTY sending the message, and text is the message. A bell sounds on the receiving terminal when the message is sent.

SEND command (Cont)

Characteristics

The SEND command:

Leaves the terminal in monitor mode.
Does not require LOGIN.
Depends on FTTALK which is normally absent in the DECsystem-1040.

Restrictions

The SEND command is not available to the Batch user.

Associated Messages

Refer to Chapter 4.

Examples

```
.SEND OPR: PLEASE WRITE-ENABLE DTA3↵
.
```

SET BLOCKSIZE command

Function

The SET BLOCKSIZE command sets a default blocksize for the specified magnetic tape.

Command Format

SET BLOCKSIZE dev: nnnn

dev: = MTAn: where n is the number of the magnetic tape drive for which the blocksize is to be set, or a logical name associated with a physical magnetic tape. The user must have the magnetic tape assigned to him. This argument is required.

nnnn = a decimal number up to a maximum of 4095 designating the block size for this magnetic tape. No additional checking is done for the legality of the specified number besides the check for the maximum 4095. This argument is required.

Characteristics

The SET BLOCKSIZE command:

Leaves the terminal in monitor mode.
Depends on both FTSET and FTMTSET which are normally absent in the DECsystem-1040.

Examples

```
.SET BLOCKSIZE MTA2:3956
.

.ASSIGN MTA4:NAME:
MTA4 ASSIGNED
.SET BLOCKSIZE NAME:2000
.
```

SET CDR command

Function

The SET CDR command sets the filename for the next card-reader spooling intercept (refer to DECsystem-10 Monitor Calls). This command is generally not needed, even when the card reader is being simulated on the disk via the spooling mechanism. It is included in case the user wishes to reset or change the spooling. In addition, the Batch Controller uses this command to read spooled input card decks.

Command Format

SET CDR filename

filename = one- to three-character filename to be used on next card-reader INIT.

Characteristics

The SET CDR command:

Leaves the terminal in monitor mode.
Depends on FTSET and FTSPL which are normally absent in the DECsystem-1040.

Examples

```
.SET CDR A↲
.
.SET CDR MAS↲
.
```

SET CPU command

Function

The SET CPU command allows a privileged user to change the CPUs on which his job can run. It is used in a multiprocessing system to specify whether the programs run under the job can be processed on the primary CPU, the secondary CPU, or either CPU. The job remains with the specified CPU until (1) another SET CPU command with a different specification is given, (2) a KJOB command is issued, or (3) the user's program overrides the SET CPU command by issuing the SETUUO with a different specification. If the SETUUO overrides the command, the specification given in the UUO remains in effect until a RESET or EXIT UUO or another SETUUO with a different specification is executed. When an EXIT or RESET UUO is executed, the job reverts back to the specification given in the last SET CPU command. When the user logs in, the CPU specification is usually set to ALL. The schedulers for each CPU compete for jobs with the ALL specification so that the load is dynamically balanced between CPUs. Therefore, this command is generally not needed but is provided in case the user wishes to change the CPU specification.

Command Formats

1. SET CPU CPxn

   adds the specified CPU to the job's CPU specification.

2. SET CPU NO CPxn

   removes the specified CPU from the job's CPU specification.

3. SET CPU ALL

   adds all of the CPUs to the job's CPU specification.

4. SET CPU ONLY CPxn

   changes the CPU specification so that it includes only the specified CPU.

   x = either U designating a logical name or A or I designating physical names for a KA10 processor (DECsystem-1055) or a KI10 processor (DECsystem -1077), respectively.

   n = a decimal number from 0 to the number of processors in the system.

SET CPU command (Cont)

Characteristics

The SET CPU command:

Leaves the terminal in monitor mode.
Depends on FTSET and FTMS which are normally absent in the DECsystem-1040, 1050, and 1070.

Restrictions

The privileges required for using this command are determined by bit 5 (JP.CCC) of the privilege word, .GTPRV.

Associated Messages

Refer to Chapter 4.

Examples

```
.SET CPU ONLY CPU1)
.
.SET CPU CPA0)
.
```

SET DENSITY command

Function

The SET DENSITY command sets a default density for the specified magnetic tape.

Command Format

SET DENSITY dev: nnn

dev: = MTAn: where n is the number of the magnetic tape drive for which the density is to be set, or a logical name associated with a physical magnetic tape. The user must have the device assigned to him. This argument is required.

nnn = 200 bpi
556 bpi
800 bpi

This argument is required.

Characteristics

The SET DENSITY command:

Leaves the terminal in monitor mode.
Depends on both FTSET and FTMTSET which are normally absent in the DECsystem-1040.

Examples

```
.SET DENSITY MTA5: 556)
.
```

SET DSKPRI command

Function

The SET DSKPRI command allows a privileged user to set the priority for his job's disk operations (data transfers and head positionings). The standard priority is 0, and the range of permissible values is -3 to +3. This means that a priority lower than the standard can be specified, as well as one higher than the standard. The priority specified applies to all disk I/O channels currently open or subsequently opened whose priority has not been explicitly set with a DISK. UUO (refer to DECsystem-10 Monitor Calls). The priority specified in the SET DSKPRI command remains in effect until (1) another SET DSKPRI command is given with a different priority, (2) a KJOB command is issued, or (3) the user's program overrides the SET DSKPRI command by issuing a DISK. UUO with a different priority.

Command Format

SET DSKPRI n

n = a decimal number from -3 to +3 indicating the priority to be associated with the job's disk operations. When n = 0, the priority is the normal timesharing priority.

Characteristics

The SET DSKPRI command:

Leaves the terminal in monitor mode.
Depends on both FTSET and FTDPRI which are normally absent in the DECsystem-1040.

Restrictions

The privileges required for using this command are determined by bits 1 and 2 of the privilege word, .GTPRV. These two bits specify an octal number from 0-3. The user is always allowed a 0 priority.

Examples

```
.SET DSKPRI 2)

.
```

SET HPQ command

Function

The SET HPQ command allows a privileged user to place his job in a high-priority scheduler run queue. With this command, the user obtains a faster response and CPU time than in the normal timesharing queues. The job remains in the specified high-priority queue until (1) another SET HPQ command to a different high-priority queue is given, (2) a KJOB command is issued, or (3) the user's program overrides the SET HPQ command by issuing an HPQ UUO with a different value. If an HPQ UUO overrides the command, the level specified in the UUO remains in effect until a RESET or EXIT UUO or another HPQ UUO with a different value is executed. When an EXIT or RESET UUO is executed, the job is returned to the high-priority queue specified in the SET HPQ command.

Command Format

SET HPQ n

n = a decimal number from 0 to 15 indicating the high-priority queue to be entered. When n = 0, the queue is the normal timesharing run queue. Queue numbers from 1 to 15 are high-priority queues. The number of high-priority queues is an installation parameter and may be less than 15.

Characteristics

The SET HPQ command:

Leaves the terminal in monitor mode.
Depends on both FTSET and FTHPQ which are normally absent in the DECsystem-1040.

Restrictions

The privileges required for using this command are determined by bits 6 through 9 of the privilege word, .GTPRV. These four bits specify an octal number from 0-17, which is the highest priority queue attainable by the user.

Examples

```
.SET HPQ 4)
.
```

SET SPOOL command

Function

The SET SPOOL command adds devices to or deletes devices from the current list of devices being spooled for this job. Spooling is the mechanism by which I/O to or from slow-speed devices is simulated on disk. Devices capable of being spooled are: the line printer, the card punch, the card reader, the paper tape punch, and the plotter.

Command Formats

1. SET SPOOL dev1, dev2, ...devn

   adds the specified devices to the job's spool list.

2. SET SPOOL ALL

   places all spooling devices into the spool list.

3. SET SPOOL NONE

   clears the entire spool list.

4. SET SPOOL NO dev1, dev2, ...devn

   removes the specified devices from the job's spool list.

   dev1, dev2, ... devn = names of one or more devices to be added to or deleted from the current spool list.

Characteristics

The SET SPOOL command:

Leaves the terminal in monitor mode.
Depends on both FTSET and FTSPL which are normally absent in the DECsystem-1040.

Restrictions

To unspool devices, the job must have (1) the privilege bit set in .GTPRV, (2) bit 28 (200 octal) set in the STATES word by the operator SET SCHED command, or (3) the user must be logged-in under [1,2].

Associated Messages

Refer to Chapter 4.

Examples

```
.SET SPOOL CDP:)
.SET SPOOL NO LPT:)
.SET SPOOL NONE)
.
```

SETSRC program

Function

The SETSRC program is used to manipulate the job's search list or the system's search list. A search list is defined to be the order of file structures that are to be searched whenever generic device DSK: is explicitly or implicitly specified by the user. This search list is originally defined by the system manager to include the file structures which the user can access. With the SETSRC program, the user can alter the search list defined for him by adding or deleting file structures.

The search list is in the form

fs1/s/s, fs2/s/s, ..., FENCE, ..., fs9/s/s

where fs is the name of the file structure and /s is a switch modifying the file structure. The file structures on the left of the FENCE comprise the active search list and represent the generic device DSK for this job. The files to the right of the FENCE comprise the passive search list and represent file structures that were once in the active search list. File structures are kept in the passive search list in order that quotas can be checked on a DISMOUNT or KJOB command. The FENCE represents the boundary between the active and passive search list.

Note that the MOUNT and DISMOUNT commands can also change the job's search list by adding or deleting a file structure. Since the SETSRC program does not create a UFD if one does not exist, the MOUNT command should be used to create a UFD. The name of the new file structure is placed at the end of the search list.

Refer to the SETSRC specification in the DECsystem-10 Software Notebooks for a complete description of the SETSRC program.

Command Format

R SETSRC

The user can then respond with any of the following commands:

Command	Explanation
A	Add one or more file structures to the existing search list. The file structures (with any switches) are appended to the beginning or the end of the active search list according to the following specifications: 1. If no asterisk appears in the specification (e.g., fs1, fs2) or if an asterisk appears before the file structure names (e.g., *, fs1, fs2), the file structures are added to the end of the search list.

(continued on next page)

Command Format (cont)

Command	Explanation
A (cont)	2. If the asterisk follows the file structure names (e.g., fs1, fs2, *), the file structures are added to the beginning of the search list. 3. If the asterisk appears in the middle of the file structures (e.g., fs1, *, fs2), the file structures before the asterisk are added to the beginning of the search list and the file structures after the asterisk are added to the end. If the specified file structure is currently in the search list, it is removed and then added in the desired position. Therefore, this command can be used to reorder the search list.
C	Create a new search list for this job. Any file structures in the current search list which are not in the new list are moved to the passive search list.
CP	Create a new default directory path.
CS	Create a new system search list (i.e., the file structure search list for device SYS:). The user must be logged in under [1,2] to use this command.
H	Obtain information about the available commands.
M	Modify the current search list and DSK specification by altering the switch settings for individual file structures. This command does not add or remove file structures from the search list.
R	Remove file structures from the search list. They are placed on the right side of the FENCE (passive search list) so that on subsequent LOGOUTs or DISMOUNTs quota limits can be checked.
T	Type the search list of the job.
TP	Type the default directory path.
TS	Type the system search list.

The following switches can be used in the SETSRC command string. Switches that modify file structures must appear immediately after the file structure that they modify. Other switches can appear anywhere in the command string. The switches can be abbreviated as long as the abbreviation is unique. The minimum number of characters is underlined below.

Switches that modify file structures

/CREATE	Allow new files to be created on the file structure.

(continued on next page)

SETSRC program (Cont)

Command Format (cont)

/NOCREATE	Do not allow new files to be created on the file structure when DSK is specified, but allow files to be superseded. Files can be created on the file structure if the user specifies the file structure name explicitly.
/NOWRITE	Do not allow writing on the file structure for this job (i.e., the file structure is read only).
/WRITE	Allow writing on the file structure.

If no switches are specified, /CREATE and /WRITE are assumed. For compatibility with previous versions of SETSRC, /N is equivalent to /NOCREATE and /R equivalent to /NOWRITE.

Switches that modify the directory path (used only with the CP command)

These switches can be typed in directly as commands by omitting the CP command and the slash (i.e., /SCAN is equivalent to CP/SCAN).

/NOSCAN	Cancel the scan switch for the directory path.
/SCAN	Set the scan switch for the directory path.

Switches that modify the DSK or SYS specification (used only with the C and M commands)

These switches can be typed in directly as commands by omitting the C or M command and the slash (i.e., NOSYS is equivalent to M/NOSYS).

/LIB: [proj, prog]	Set the job's library directory to the UFD [proj,prog] and add it to the user's DSK specification. This means that if a file is not found in the user's directories in his search list, the library directory will then be searched for the file.
/NOLIB	Remove the library directory from the user's DSK specification.
/NOSYS	Remove the SYS specification from the user's DSK specification.
/NONEW	Remove the [1,5] directory from the user's SYS specification.
/SYS	Add the SYS specification to the user's DSK specification. This means that if a file cannot be found in the user's directories in his search list or in his library directory (if /LIB: [proj,prog] has been specified), the system directory [1,4] will then be searched for the file.
/NEW	Add the directory [1,5] to the user's SYS specification. This means that when the system directory is searched, the directory [1,5] will be searched before the directory [1,4].

Characteristics

The R SETSRC command

Places the terminal in user mode.
Runs the SETSRC program, thereby destroying the user's core image.

Restrictions

The user must be logged in under [1,2] to create a new system search list. The directory path commands (CP and TP) are meaningful only with the 5.04 and later monitors and only if FTSFD is on.

Examples

```
.R SETSRC ↵

*T ↵
DSKB:, FENCE                          The user's search list is defined as DSKB.

*A DSKA: ↵                            Add DSKA to the end of the search list.
*T
DSKB:,DSKA:,FENCE                     The user's search list is defined as DSKB,DSKA.

*A DSKC:,* ↵                          Add DSKC to the beginning of the search list.
*T ↵

DSKC:,DSKB:,DSKA:,FENCE
*R DSKA: ↵                            Remove DSKA from the search list.
*T ↵

DSKC:,DSKB:,FENCE,DSKA:               The user's search list is defined as DSKC,DSKB.

*M DSKB:/NOWRITE ↵                    Do not allow writing on DSKB.

*M /LIB:[27,500] ↵                    Set the user's library directory to [27,500] and add
                                      it to the user's DSK specification.
*SYS ↵                                Add SYS: to the user's DSK specification.

*T↵
/LIB:[27,500]/SYS  DSKC:,DSKB:/NOWRITE,FENCE,DSKA:

                                      The user's DSK and SYS specifications are first
                                      followed by the user's search list.

*TS ↵

DSKA:,DSKB:,DSKC:,FENCE               The system search list is defined as DSKA, DSKB,
                                      DSKC.
```

SET TIME command

Function

The SET TIME command sets a central processor time limit for a job. When the time limit is reached, the job is stopped and a message is typed. A timesharing job may be continued by typing CONT, but no time limit is in effect unless it is reset. A Batch job cannot be continued.

Command Format

SET TIME n

n = number of seconds of central processor time to which the job is limited. An argument of 0 cancels the time remaining.

Characteristics

The SET TIME command:

Leaves the terminal in monitor mode.
Depends on both FTSET and FTTLIM which are normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.MAKE LOOP.F4)                        Create a program with an indefinite loop.

*I10     CONTINUE
         GOTO 10
         END
$$
*EX$$

.TYPE LOOP.F4)                        Type the program.
10       CONTINUE
         GOTO 10
         END
```

(continued on next page)

Examples (cont)

```
.LOAD LOOP)
FORTRAN:  LOOP.F4
LOADING

LOOP 2K CORE

EXIT
.SET TIME 5)
```

Compile and load the program.

Set the time limit to 5 seconds.

```
.TIME)
3.50
5.57
KILO-CORE-SEC=32
```

Clear the incremental run time, so that the SET TIME command can be checked.

```
.START)
```

Start the loop.

```
?
?TIME LIMIT EXCEEDED
```

As expected, the time limit was exceeded.

```
.TIME)
5.00
10.57
KILO-CORE-SEC=67

.
```

SET TTY or TTY command

Function

The SET TTY command (or TTY command) declares properties of the terminal line on which the command is typed to the scanner service. With hardwired TTYs, the system manager can set the default conditions, so that this command is usually not needed. However, the user is likely to use this command on data sets, where the terminal cannot be predicted.

Command Formats

1. SET TTY NO word

   equivalent to TTY NO word

2. SET TTY word

   equivalent to TTY word

   NO = the argument that determines whether a bit is to be set or cleared. This argument is optional.

   word = the various words representing bits that may be modified by this command. The words are as follows:

SET TTY ALTMODE	Converts the ALTmode codes of 175 and 176 to the ASCII standard escape character 033.
SET TTY NO ALTMODE	Restores the individual identity of the codes 175 and 176.
SET TTY BLANKS	Restores multiple carriage return-line feeds and form feeds.
SET TTY NO BLANKS	Suppresses blank lines (consecutive carriage return-line feeds after the first) and outputs form feeds and vertical tabs as 2 carriage return-line feeds. This is used for the display terminal in order to prevent the output from moving up off the screen.
SET TTY CRLF	Restores the carriage return.
SET TTY NO CRLF	The carriage return normally output at the end of a line exceeding the carriage width is suppressed.
SET TTY ECHO	Restores the normal echoing of each character typed in.

(continued on next page)

Command Formats (cont)

SET TTY NO ECHO	The terminal line has local copy and the computer should not echo characters typed in.
SET TTY FILL n	The filler class n is assigned to this terminal. The filler character is always DEL (RUBOUT, 377 octal). No fillers are supplied for image mode output.
SET TTY NO FILL	Equivalent to TTY FILL 0. Fillers for output and echoing are determined from the following:

Character Name	Octal	Number of Fillers for Filler Class			
		0	1	2	3
BS	010	0	2	6	6
HT	011	0	1 or 2	1 or 2	1 or 2†
LF	012	0	1	6	6
VT	013	0	2	6	6
FF	014	0	12	21	21
CR	015	0	1 or 2	2 or 4	2 or 4††
XON	021	0	1	1	1
TAPE	022	0	1	1	1
XOFF	023	0	1	1	1
NTAP	024	0	1	1	1

†1 if 0-3 spaces to tab stop; 2 if 4-7 spaces to tab stop.
††1 or 2 if CR is typed; 2 or 4 if CR is supplied because the line is too long.

SET TTY FORM	This terminal has hardware FORM (PAGE) and VT (vertical tab) characters.
SET TTY NO FORM	The monitor sends eight line feeds for a FORM and four line feeds for a VT.
SET TTY GAG	The SEND command cannot be received at this terminal unless the terminal is at command level (initial state).
SET TTY NO GAG	The SEND command can be received at this terminal even though it is not at command level.
SET TTY LC	The translation of lower-case characters input to upper case is suppressed.

(continued on next page)

SET TTY command (Cont)

Command Formats (cont)

SET TTY NO LC	The monitor translates lower-case characters to upper case as they are received. In either case, the echo sent back by the monitor matches the case of the characters after translation. By looking at the printout, the user can determine what translation was performed by the monitor.
SET TTY PAGE	The user has the ability to temporarily suspend system typeout without losing it. The XOFF key (↑S) suspends the typeout, and the XON key (↑Q) restores it. The XOFF and XON keys are not echoed and are not sent to the user's program. This command is useful for display terminals where the user may want to read a page of text before it disappears from the screen. Note that this preempts the use of ↑S and ↑Q for reading paper tape (see SET TTY TAPE).
SET TTY NO PAGE	The typeout control ability of the XOFF and XON keys is disabled. The current interpretation of these keys depends on the last SET TTY TAPE command.
SET TTY SLAVE	The terminal becomes slaved, i.e., no commands may be typed on the terminal, and the terminal may be ASSIGNed by another user. The user can slave only his own terminal and must contact the operator in order to unslave it.
SET TTY TAB	This terminal has hardware TAB stops every eight columns.
SET TTY NO TAB	The monitor simulates TAB output from programs by sending the necessary number of SPACE characters.
SET TTY TAPE	The XON key (↑Q) causes the terminal to read paper tape. The XOFF key (↑S) causes the terminal to stop reading paper tape.
SET TTY NO TAPE	The XON key (↑Q) and the XOFF key (↑S) have no special paper tape function. They may have a PAGE function.
SET TTY WIDTH n	The carriage width (the point at which a free carriage return is inserted) is set to n. The range of n is 17 (two TAB stops) to 200 decimal.

Characteristics

The SET TTY command:

Leaves the terminal in monitor mode.
Does not require LOGIN.
Depends on FTSET which is normally absent in the DECsystem-1040. However, the TTY command format can always be used.

SET WATCH command

Function

The SET WATCH command sets the system to print incremental job statistics automatically. This command provides the user with a tool for measuring the performance of his programs.

Command Formats

1. SET WATCH arg_1, arg_2,...,arg_5

   prints the specified WATCH statistics.

2. SET WATCH ALL

   prints all the WATCH statistics.

3. SET WATCH NONE

   eliminates the printing of all WATCH statistics.

4. SET WATCH NO arg_1, arg_2,...,arg_5

   eliminates the printing of the specified WATCH statistics.

   The following arguments enable printing whenever a monitor command switches the console from monitor to user mode.

   arg = DAY prints the time of day, as [HH:MM.SS]

   arg = VERSION prints the version of the program in standard format (refer to the VERSION command).

   The following arguments enable printing whenever the console is returned to monitor mode via the ↑C, EXIT, HALT, ERROR IN JOB n, or DEVICE xxx OPR zz ACTION REQUESTED messages.

   arg = READ prints the incremental number of disk blocks read modulo 4096.

   arg = RUN prints the incremental run time.

   arg = WAIT prints the wait time (time elapsed since the user started or continued the program).

   arg = WRITE prints the incremental number of disk blocks written modulo 4096.

Command Formats (cont)

Any combination of the arguments may be specified in any order. Statistics are not printed for commands that do not run programs, such as ASSIGN or PJOB. When a user logs in, his job is set to WATCH the statistics of which he has notified the system manager. The information on what statistics to WATCH is kept in ACCT.SYS.

The order of the error message is the same as the order of output. Therefore, a user who forgets either the argument or the significance of the statistics can find these out by examining the message. A single space is always typed between each statistic, whether the statistic appears or not; therefore, it is possible to tell which statistics are being typed.

NOTE

Enabling WATCH output interacts with the incremental data typed by the TIME and DSK commands.

Characteristics

The SET WATCH command:

Leaves the terminal in monitor mode.
Depends on FTSET and FTWATCH which are normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

SET WATCH command (Cont)

Examples

1.
```
.SET WATCH P
?ARGS ARE: DAY,RUN,WAIT,READ,WRITE,VERSION,ALL,NONE

.SET WATCH
?ARGS ARE: DAY,RUN,WAIT,READ,WRITE,VERSION,ALL,NONE

.SET WATCH DAY RUN WAIT READ WRITE

.R PIP
[22:38:19]
*↑C
[0.10 2.95 457 243]
```

2.
```
.SET WATCH VERSION DAY

.R TECO
[9:44:30]

[S:TECO 22(64) + ]
*↑C
```

SKIP command [1]

Function

The SKIP command spaces a magnetic tape forward a specified number of files or records or to the logical end of tape. This command, depending on its arguments, is equivalent to the following PIP command strings:

```
MTAn: (M #nA) ←
MTAn: (M #nD) ←
MTAn: (M #nT) ←
```

Command Formats

1. SKIP MTAn: x FILES

   advances forward x files.

2. SKIP MTAn: x RECORDS

   advances forward x records.

3. SKIP MTAn: EOT

   advances forward to the logical end of tape.

The words FILES, RECORDS, and EOT can be abbreviated to F, R, and E, respectively.

Characteristics

The SKIP command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

[1] This command runs the COMPIL program, which interprets the command before running the PIP program.

SKIP command (Cont)

Examples

```
.SKIP MTA0: 4 FILES )
.SKIP MTA1: EOT)
.SKIP MTA2: 20 RECORDS )
```

SSAVE command

Function

The SSAVE command is the same as the SAVE command except that the high segment, if present, will be sharable when it is loaded with the GET command. To indicate this sharability, the high segment is written with extension .SHR instead of .HGH. A subsequent GET will cause the high segment to be sharable. Because an error message is not given if the program does not have a high segment, a user can use this command to save system programs without having to know which are sharable.

On magnetic tape, if the low or high segment is missing, a null record is output before the EOF for the missing segment so that two EOFs cannot occur consecutively. Therefore, a saved null segment does not appear as a logical EOT (2 EOFs in a row).

The SAVE command rather than the SSAVE command, should be used when debugging the program. This is because a GET command after a SSAVE command does not reinitialize the original high segment from the file after the user modifies it with the D command or the DDT program. Refer to Appendix D for more information on the SSAVE command.

Command Format

SSAVE dev:file.ext [proj,prog] core

Arguments and defaults are the same as in the SAVE command.

Characteristics

The SSAVE command:

Leaves the terminal in monitor mode.
Requires core.
Does not operate when a device is currently transmitting data.

Associated Messages

Refer to Chapter 4.

Example

```
.SSAVE DSK:TEST↵
JOB SAVED

.
```

(continued on next page)

SSAVE command (Cont)

Example (Cont)

```
        .LOAD FILE1↵                    Compile and load program.
        MACRO: FILE1
        LOADING

        LOADER 1K CORE
        EXIT

        .SSAVE↵                         Save a sharable copy.  The filename is taken from
        JOB SAVED                       the routine that contained the starting address.

        .GET↵                           Get a sharable copy.
        JOB SETUP
```

START command

Function

The START command begins execution of a program either previously loaded with the GET command or interrupted (e.g., ↑C). The old program counter is copied from .JBPC to .JBOPC. An explicit start address is optional, and, if omitted, the address supplied in the file (.JBSA) is used. If an address argument is specified and the job was executing a UUO when interrupted (i.e., it was in exec mode but not in TTY input wait or SLEEP mode), the monitor sets a status bit (UTRP) and continues the job at the location at which it was interrupted before trapping to the specified START address. When the UUO processing is completed, the monitor clears the status bit, sets .JBOPC to the address following the UUO, and then traps to the START address. If the job is in TTY input wait or SLEEP mode, the trap to the program occurs immediately, and .JBOPC contains the address of the UUO. If the job is in user mode, the trap also occurs immediately.

Command Format

START adr

adr = the address at which execution is to begin if other than the location specified within the file (.JBSA). This argument is optional. If adr is not specified, the address comes from .JBSA. A starting address of 0 may be specified.

Characteristics

The START command:

Places the terminal in user mode.
Does not operate when a device is currently transmitting data.
Requires core.
Requires LOGIN if an address argument is specified.

Associated Messages

Refer to Chapter 4.

Example

```
.START↲
```

SUBMIT command

Function

The SUBMIT command is used to place entries into the input queue for the Batch system. This command is equivalent to the following form of the QUEUE command:

QUEUE INP: jobname = control file, log file

Command Format

SUBMIT jobname = control file, log file

jobname = name of the job being entered into the queue.

control file = name of the control file. This file contains all monitor-level and user-level commands for processing by the Batch Controller (BATCON).

log file = name of the log file. This file is used by the Batch Controller to record its processing of the job.

Only the two files mentioned above can be specified in a request to the Batch input queue. The name of the control file is required; the log file name is optional and, if omitted, is taken from the control file. If the jobname is ommited, it is the name of the first file in the request, not the name of the first file given. If an extension is omitted, the following are assumed:

.CTL for the control file
.LOG for the log file.

Three categories of switches can be used in the command string:

1. Queue-operation - Only one of these switches can be placed in the command string because they define the type of queue request. The switch used can appear anywhere in the command string.

2. General - Each switch in this category can appear only once in the command string because they affect the entire request. The switch used can appear anywhere in the command string.

3. File control - Any number of these switches can appear in the command string because they are specific to individual files within the request. The switch used must be adjacent to the file to which it applies. If the switch precedes the filename, it becomes the default for subsequent files.

Command Format (cont)

The following switches can be used with the SUBMIT command.

Switch	Explanation	Category
/AFTER:tt	Process the request after the specified time; tt is either in the form of hhmm (time of day) or +hhmm (time later than the current time). The resulting AFTER time must be less than the DEADLINE time. If the switch, or the value of the switch, is omitted, no AFTER constraints are assumed.	General
/CARDS:n	Use n (decimal) as the maximum number of cards that can be punched by the job. If the switch is omitted, no cards are punched. If the switch is given with no value, 2000 cards is assumed as the default.	General
/CORE:n	Use n (decimal K) as the maximum amount of core memory that the job can use. If the switch is omitted, 25K is the maximum. If the switch is specified, but the value is omitted, the default maximum is 40K.	General
/CREATE	Make a new entry into the Batch input queue. This switch is the default for the queue-operation switches.	Queue Operation
/DEADLINE:tt	Process the request before the specified time; tt is either in the form hhmm (time of day) or +hhmm (time later than the current time). The resulting DEADLINE time must be greater than the AFTER time. If the switch, or the value of the switch, is omitted, no DEADLINE constraints are assumed.	General
/DEPEND:n	Specify the initial value of the dependency count (in decimal). When used with /MODIFY, this switch changes the dependency count of another job. If n is a signed number (+ or -), that number is added to or subtracted from the dependent job's count. If n is not a signed number, the dependent job's count is changed to n. If this switch is omitted, no dependency is assumed.	General
/DISPOSE:DELETE	Delete the file after processing.	File Control

(continued on next page)

SUBMIT command (Cont)

Command Format (cont)

Switch	Explanation	Category
/DISPOSE:PRESERVE	Save the file after processing. This is the default for all files except those with extensions .TMP, .LST, .CDP, .LPT, .PLT, and .PTP.	File Control
/DISPOSE:RENAME	Rename the file from the specified directory immediately, remove it from the logged-out quota, and delete it after processing. This is the default for files with extensions .TMP, .LST, .CDP, .LPT, .PLT, and .PTP.	File Control
/F	List the entries in the input queue, but do not update the queues. Therefore, the list may not be an up-to-date listing, but the listing will be faster than with /LIST.	Queue Operation
/FEET:n	Use n (decimal) as the maximum number of feet of paper tape that the job can punch. If the switch is omitted, no paper tape is punched. If the value is omitted, the default is 10*B+20 feet, where B is the number of blocks in the request.	General
/KILL	Remove the specified entry from the Batch input queue. This switch can be used for deleting a previously-submitted request as long as the request has not been started by the Batch Controller.	Queue Operation
/LIST	List the entries in the input queue; the default is all entries for all jobs of all users.	Queue Operation
/MODIFY	Alter the specified parameters in the job. This switch requires that the user have access rights to the job. It can be used for altering a previously submitted request as long as the request has not been started by the Batch Controller.	Queue Operation
/NEW	Accept the request although the file does not yet exist. This is the default for the log file. When placing this switch with the control file, the user can submit his job and then create the control file.	File Control

(continued on next page)

Command Format (cont)

Switch	Explanation	Category
/OUTPUT:n	Cause job to terminate with a /Z:n to KJOB (n is from 0 to 4). N=0 Suppress all normal queuing performed at LOGOUT time. N=1 Queue only the log file. N=2 Queue only the log file and spooled output (e.g., *.LPT). N=3 Queue the log file, spooled output, and *.LST files. N=4 Queue the log file, spooled output, *.LST files, and any requests deferred to LOGOUT time (default).	General
/PAGE:n	Use n (decimal) as the maximum number of pages of output that the job can print. If the entire switch is omitted, the maximum is 200 pages; if only the value is omitted, the maximum is 2000 pages.	General
/PHYSICAL	Suppress logical device name assignments for the device specified.	File Control
/PRIORITY:n	Assign the specified external priority (n=0 to 62) to the request. The larger the number, the greater priority the job has. The default is 10 if no switch is given and 20 if the switch is specified without a value.	General
/PROTECT:nnn	Assign the protection nnn (octal) to the job. If the switch, or the value of the switch, is omitted, the standard protection is assumed.	General
/RESTART:0 or 1	A value of 0 means the job is not requeued or restarted by the Batch Controller after a system crash (default). A message is sent to the job's log file. A value of 1 means the job is restarted by the Batch Controller.	General
/SEQ:n	Specify a sequence number to help in identifying a request to be modified or deleted.	General

(continued on next page)

SUBMIT command (Cont)

Command Format (cont)

Switch	Explanation	Category
/START:n	Begin on the nth line of the control file. If the switch, or the value of the switch, is omitted, the Batch Controller starts with the first line.	File Control
/START:xxx	Start at the statement labelled xxx (up to 5 characters) of the control file.	File Control
/TIME:hhmmss	Specify the central processor time limit for the job. If no switch is specified, the limit is 5 minutes; if the switch is given without a value, the limit is 1 hour.	General
/TPLOT:n	Use n (decimal minutes) as the maximum amount of plotting time allowed for the job. If the switch is omitted, no plotter time is allowed; if the value is omitted, but the switch is given, the maximum is 10 minutes.	General
/UNIQUE:0 or 1	Run any number of Batch jobs under this project-programmer number at the same time, if 0. Run only one Batch job at any one time, if 1 (default).	General

Characteristics

The SUBMIT command:

Leaves the terminal in monitor mode.
Runs the QUEUE program.
Depends on FTQCOM which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.SUBMIT USRJOB=CONTRL,LOGFIL)
```

The defaults are as follows:

1. control file name is CONTRL.CTL
2. log file name is LOGFIL.LOG
3. no cards punched (/CARDS:0)
4. maximum core of 25K (/CORE:25)
5. no dependency (/DEPEND:0)
6. control and log files are saved after spooling (/DISPOSE:PRESERVE)
7. no paper tape punched (/FEET:0)
8. all line printer output is spooled with the maximum pages being 200 (/OUTPUT:4, /PAGE:200)
9. priority is 10 (/PRIORITY:10)
10. standard protection is assumed (/PROTECT:nnn (standard))
11. job is not restarted after a crash (/RESTART:0)
12. control file is begun on the first line (/START:1)
13. maximum CPU time is 5 minutes (/TIME:0:05)
14. no plotter time allowed (/TPLOT:0)
15. only one job at a time under a given project-programmer number is run (UNIQUE:1).

```
.SUBMIT USRJOB=/MODIFY/FEET:35/CORE)
```

Modify the original request to include 35 feet as the maximum number of feet of paper tape that the job can punch and 40K of core as the maximum amount of core that the job can use. This command is valid only if the job has not been started yet by the Batch system.

```
.SUBMIT USRJOB=/KILL)
```

Kill the job only if it has not been started by the Batch system.

SYSTAT command

Function

The SYSTAT command runs a system program which prints status information about the system. This information allows a user to determine the load on the system before logging-in.

To write the output on the disk as a file with name SYSTAT.TXT, assign device DSK with logical name SYSTAT.

The SYSTAT command types the status of the system: system name, time of day, date, uptime, percent null time (idle plus lost time), number of jobs in use.

It types the status of each job logged-in: job number; project-programmer number (**,** = detached, [OPR] = the project-programmer number of the operator, [SELF] = user's project-programmer number); terminal line number (CTY = console terminal, DET = detached, Pn = PTY number); program name being run; program size; job and swapped state (refer to DECsystem-10 Monitor Calls); run time since logged-in.

It types the status of high segments being used: name (PRIV = nonsharable, OBS = superseded); device or file structure name from which the segment came; directory name (**,** if detached); size (SW = swapped out, SWF = swapped out and fragmented, F = in core and fragmented on disk, SPY = user is executing the SPY UUO); number of users in core or on the disk.

The command types swapping space used, virtual core used, swapping ratio, active swapping ratio, virtual core saved by sharing, average job size.

It types status of busy devices: device name, job number, how device is assigned (AS = ASSIGN command, INIT = INIT or OPEN UUO, AS+INIT = both ways).

It types system file structures: free blocks, mount count, single-access job.

It types remote stations: number of station, status of station.

It types dataset control: number of the TTY, status of TTY.

Command Format

SYSTAT arg

arg = one or more single letters (in any order) used to type any subset of the SYSTAT output. This argument is optional. The following message, produced by typing SYSTAT /H, lists the various arguments to the SYSTAT command.

```
.SYSTAT/H
SYSTAT V467(5)
SYSTAT INSTRUCTIONS:
TYPE  "SYS<C.RET.>" TO LIST THE ENTIRE STATUS, OR
TYPE  "SYS " FOLLOWED BY ONE OR MORE LETTERS AS FOLLOWS--
B  BUSY DEVICE STATUS
D  DORMANT SEGMENT STATUS
E  NON-DISK ERROR REPORT
F  FILE STRUCTURE STATUS
H  THIS MESSAGE
J  JOB STATUS
L  OUTPUT TO LPT
N  NON-JOB STATUS (ALL BUT J)
O  OTHER SYSTEM STATUS
P  DISK PERFORMANCE
R  REMOTE STATION STATUS
S  SHORT JOB STATUS
T  DATASET STATUS
X  READ DSK:CRASH.XPN
NNN PRINTS JUST JOB NNN (. DOES THIS JOB)
[P,PN] PRINTS JUST JOBS WITH THAT PROJ-PROG (P AND/OR PN MAY BE *)
#NNN PRINTS JUST JOBS FROM TERMINAL NNN
        (ALSO, C=CTY, PNN=PTYNN, TNN=TTYNN, .-THIS TTY)
```

Characteristics

The SYSTAT command:

Leaves the terminal in monitor mode.
Runs the SYSTAT program, thereby destroying the user's core image.
Does not require LOGIN.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

SYSTAT command (Cont)

Examples

```
.SYSTAT)

STATUS OF B5Ø4ØØ-Ø5 #4Ø. AT 16:15:17 ON 11-APR-72

UPTIME 7:ØØ:59, 1Ø8% NULL TIME = 1Ø5% IDLE + 3% LOST
53 JOBS IN USE OUT OF 64.  5Ø LOGGED IN, 2 DETACHED

JOB     WHO       LINE#     WHAT    SIZE(K)   STATE    RUN TIME

 1     [OPR]      DET      DAEMON   5+SPY     SL  SW        2Ø
 2     4Ø,64      131      TECO     2+3       ↑C  SW         7
 3     [OPR]       2       OPSER    1+3       HB  SW      3:43
 4     [OPR]       4       UMOUNT   2+3       ↑C  SW      3:38
 5      2,111      3       PIP      1+4       ↑C  SW        26
 6    274,1431    6Ø       COBDDT   14+8      RN         12:15
 7     [OPR]      CTY      PIP      1+4       ↑C  SW        43
 8     16,35      2Ø       FDSYS    22        TI  SW      1:37
 9    347,1246    65       PIP      1+4       ↑C  SW         5
1Ø    271,1131    64       QUEUE    2+4       ↑C  SW        29
11     [OPR]      PØ       BATCON   2+3       RN          2:45
12    271,7Ø1     61       SEEK     2+8       TI  SW      5:Ø8
13    337,1113    114      KJOB     1+3       ↑C  SW        13
14     4Ø,64      15Ø      TECO     2+3       TI  SW      4:37
15    122,216     124      KJOB     1+4       CB             2
16     16,1Ø7     P2       SYSTAT   6+SPY     SL  SW        25
17    142,1243    62       TECO     2+3       TI          1:ØØ
18    146,5ØØ     112      TECO     2+3       TI          2:Ø9
19     4Ø,65      147      EDITS    2+6       TI  SW      1:24
2Ø    24Ø,353     155      TECO     2+3       TI  SW         1
21     [OPR]      P1       LPTSPL   2         IO         13:47
22    345,1417    74       EDITS    2+6       ↑C  SW        43
23     [OPR]      P3       LPTSPL   2         IO         18:39
24     [OPR]      P4       PTPSPL   2+3       HB  SW      1:25
25     [OPR]       1       OPSER    1+3       HB  SW        46
26     [OPR]      P6       OPROMO   2+5       TI  SW      1:38
27     [OPR]      P5       OPROMO   2+5       SL  SW      1:31
28    36Ø,1436    137      UMOUNT   2+3       CB          1:Ø1
29    125,2Ø7     15       EDITS    2+6       TI  SW         1
3Ø    11Ø,1412    127      TECO     2+3       DI             7
31    4Ø2,57Ø     12       PIP      1+4       CB  SW      1:26
32    4Ø5,5Ø5     25       PIP      1+4       ↑C  SW      1:13
33     4Ø,633     146      TECO     2+3       TI          5:33
34    271,7Ø1     14       SEEK     2+8       TI  SW         4
35     1Ø,2Ø      142      D        13+34     TI         33:14
36    413,521     21       TECO     2+3       TI            2Ø
37    4Ø2,57Ø     P7       DCTDE    4         ↑C  SW        12
```

(continued on next page)

Examples (cont)

```
38    122,1202    136    EDITS    2+6    TI SW         1
39     60,60      153    TECO     2+3    TO         6:33
40    406,104     35     PLEASE   1      SL SW         3
41    402,574     36     SPACE    2      TI SW        10
42    271,701     30     RAFCO'   6+8    TI         2:11
43     14,1145    P10    UMOUNT   2+3    SL SW         1
44     14,1145    11     DIRECT   1+3    ↑C SW        45
45      2,5       135    QUEUE    3+3    CB SW         0
46     11,176     144    COMPCT   5      TO         2:16
47     10,33      DET    UMOUNT   2+3    TO SW         1
48     10,34       6     TECO     4+3    TI         3:02
49      2,5       13     SYSTAT   6+SPY  RN            0
50    110,1367    157    TECO     2+3    TI SW        16
51    340,1107    26     AID      2+9    RN SW         1
52      2,5        7     SYSTAT   4      CB            0
53    122,1007    122    TECO     2+3    TI SW        24
PNN CORRESPONDS TO TTY172+NN

HIGH SEGMENTS:
PROGRAM DEVICE  OWNER    HIGH(K) USERS

OPSER   DSKB    SYS       3 SW    2
DIRECT  DSKB    SYS       3 SW    1
QMANGR  DSKB    SYS       3       2
PIP     DSKB    SYS       4  SW   5
TECO    DSKB    SYS       3      12
EDITS   DSKB    SYS       6 SW    4
(PRIV)          JOB 35   34       1
QUEUE   DSKB    SYS       4       2
LIBOL   DSKB    SYS       8       4
UMOUNT  DSKB    SYS       5 SW    2
PTPSPL  DSKB    SYS       3 SW    1
UMOUNT  DSKB    SYS       3       4
KJOB    DSKB    SYS       3 SW    1
AID     DSKB    SYS       9  SW   1

SWAPPING SPACE USED = 156/635 = 25%
VIRT. CORE USED = 254/635 = 40%
SWAPPING RATIO = 254/144 = 1.8
ACTIVE SWAPPING RATIO = 75/144 = 0.5
VIRT. CORE SAVED BY SHARING = 115/(115+254) = 31%
AVERAGE JOB SIZE = 163/53 = 3.1 + 206/53 = 3.9  TOTAL = 369/53 = 7.0
```

(continued on next page)

SYSTAT command (Cont)

Examples (cont)

```
BUSY DEVICES:
DEVICE   JOB      WHY      LOGICAL

LPTØ     23       AS+INIT
LPT1     21       AS+INIT
DTA1     14       AS
DTA2     19       AS
DTA3     33       AS
DTA4      7       AS
DTA6     27       AS+INIT
DTA7     26       AS+INIT
MTA1     37       AS
MTA2     35       AS
MTA3     31       AS
PTPØ     24       INIT
72 DISK DDBS

SYSTEM FILE STRUCTURES:
NAME     FREE     MOUNT
DSKA     7Ø5      12
DSKB     1722Ø    54
DSKC     35895    5
DIAG     2835     3
TOTAL FREE 56655

DATASET CONTROL
TTY#     STATUS

6Ø       IN USE
61       IN USE
62       IN USE
64       IN USE
65       IN USE
72       IN USE
74       IN USE

.KJOB
```

(continued on next page)

Examples (cont)

```
.SYSTAT P

STATUS OF B50400-05 #40 AT 16:27:49 ON 11-APR-72


DISK PERFORMANCE STATISTICS:
UNIT OR F/S
          BR        BW        DR        DW        XR        XW        MR        MW
DSKA      716 FREE
FHA1(1RD002): 716 FREE, 0 SEEKS
          6616      177       59861     8811      0         0         30062     27935
 ERRORS: 1DAT:1 RETRIES:1 2CONI:4000,15 1CONI:4000,4015 2DATAI:21 1DATAI
:140071
DSKB      17215 FREE
DPA0(102446): 4445 FREE, 54617 SEEKS
        . 30498     39483     12648     1155      0         0         41622     10489
 MSB  ERRORS: 1DAT:11 RETRIES:1 2CONI:15 1CONI:5,4015 2DATAI:54661,40000
 1DATAI:54661,40000
DPA1(117986): 4345 FREE, 42791 SEEKS
          37609     41724     17037     2790      0         0         20563     9505
 MSB  ERRORS: 1DEV:8 1DAT:240 RETRIES:1 2CONI:15 1CONI:5,4015 2DATAI:102
261,240000 1DATAI:102261,240000
DPA2(102376): 4170 FREE, 33215 SEEKS
          35205     35413     14911     1236      0         0         15503     7687
 MSB
DPA4(102490): 4310 FREE, 32078 SEEKS
          38472     32920     12599     2215      0         0         9711      5934
 MSB  ERRORS: 1DEV:6 RETRIES:3 2CONI:15 1CONI:40015 2DATAI:456661,400000
 1DATAI:456661,0
DSKC      35925 FREE
DPA5(151669): 35925 FREE, 2863 SEEKS
          1705      337       250       1         0         0         4236      1014
 MSB
DIAG      2945 FREE
DPA6(DIAG01): 2945 FREE, 6808 SEEKS
          2133      2532      8423      533       0         0         8199      2602
 MSB

ACTIVE SWAPPING STATISTICS:
UNIT      R         W         USED(K)
FHA0      809992    366056    281/335 = 84%
FHA1      397720    351376    181/300 = 60%

.KJOB
```

SYSTAT command (Cont)

Examples (Cont)

```
.SYS [10,*]
SYSTAT V467(5)
10      [SELF]   10      SYSTAT   5+SPY    RN                3
14      10,133    0      TECO     2+3      TI               47



.SYS /O
SYSTAT V467(5)

STATUS OF        50416A SYSTEM #2 AT 1:16:11 P.M. ON 24-FEB-72

UPTIME 45:47, 60% NULL TIME = 55% IDLE + 5% LOST
15 JOBS IN USE OUT OF 37.  15 LOGGED IN, 1 DETACHED

SWAPPING SPACE USED = 68/350 = 19%
VIRT. CORE USED = 85/350 = 24%
SWAPPING RATIO = 85/18 = 4.7
ACTIVE SWAPPING RATIO = 5/18 = 0.3
VIRT. CORE SAVED BY SHARING = 5/(5+85) = 6%
AVERAGE JOB SIZE = 49/15 = 3.3 + 41/15 = 2.7  TOTAL = 90/15 = 6.0
```

TECO command [1]

Function

The TECO command runs TECO and opens an already existing file on disk for editing. Refer to the TECO manual in the DECsystem-10 Software Notebooks.

Command Format

TECO dev:file.ext [proj,prog]

dev: = the device or file structure name containing the existing file. If omitted, DSK: is assumed.

file.ext = the filename and filename extension of the existing file. If omitted, the arguments of the last EDIT-class command are used.

[proj,prog] = the directory name in which the file appears. If omitted, the user's directory is assumed.

Characteristics

The TECO command:

Places the terminal in user mode.
Runs the TECO program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Example

```
.TECO TEST1.MAC ⤶

*↑C

.TECO DSKB:FILNAM.CBL [100,27] ⤶
```

[1]This command runs the COMPIL program, which interprets the commands before running TECO.

TIME command

Function

The TIME command causes typeout of the total running time since the last TIME command, followed by the total running time used by the job since it was initialized (logged-in), followed by the integrated product of running time and core size (KILO-CORE-SEC=). Time is typed in the following format:

hh:mm:ss.hh

where

hh = hours
mm = minutes
ss.hh = seconds to nearest hundredth.

Interrupt level and job scheduling times are charged to the user who was running when the interrupt or rescheduling occurred.

NOTE

If automatic runtime is enabled using the SET WATCH command, the incremental runtime is usually 0.

Command Format

TIME job

job = the job number of the job whose timing is desired. If job is omitted, the job to which the terminal is attached is assumed. In this case, monitor types out the incremental running time (running time since last TIME command) as well as the total running time since the job was initialized.

Characteristics

The TIME command:

Leaves the terminal in monitor mode.
Does not require LOGIN.

Associated Messages

Refer to Chapter 4.

Example

```
.TIME
0.38
0.38
KILO-CORE-SEC=1
```

The command is given for the first time after LOGIN; therefore, the incremental time equals the total time since LOGIN.

```
.TI
0.00
0.38
KILO-CORE-SEC=1

.DIR/F

NEW505  BAK      DSKB:    [10,770]
NEW505  RNO
FOO     SFD
C'      MAC      DSKC:
PETALS  F4
PETALS  SAV
CSHELL  MAC
CIO     MAC
TRYCIO  MAC
METER   RNO
SURFIT  ALG

.TI
0.40
0.78
KILO-CORE-SEC=6
```

The DIRECT command took .40 seconds of runtime and 5 kilo-core-seconds.

TPUNCH command

Function

The TPUNCH command is used to place entries into the paper-tape punch output queue. This command is equivalent to the following form of the QUEUE command:

    QUEUE PTP: jobname = list of input specifications.

The TPUNCH command can be further abbreviated to

    PUNCH jobname = list of input specifications.

However, individual installations may redefine PUNCH to mean output to the card-punch queue instead of the paper-tape punch queue.

Command Format

TPUNCH jobname = list of input specifications

jobname = name of the job being entered into the queue. The default is the name of the first file in the request not the name of the first file given. These differ when the first file given does not yet exist.

input specifications = a single file specification or a string of file specifications, separated by commas, for the disk files being processed. A file specification is in the form dev:file.ext [proj,prog].

dev: = any file structure to which PTPSPL will have access; the default is DSK:.

file.ext = names of the files. The filename is optional. The default for the first filename is *, the default for subsequent files is the last filename used. The extension can be omitted; the default is .PTP.

[proj,prog] = a directory to which the user has access; the user's directory is assumed if none is specified.

The wildcard construction can be used for the input specifications.

If no arguments appear in the command string (i.e., only the command name is given), all entries in the paper-tape punch queue for all jobs are listed.

Switches that aid in constructing the queue entry can also appear as part of the input specifications. These switches are divided into three categories:

1. Queue-operation - Only one of these switches can be placed in the command string because they define the type of queue request. The switch used can appear anywhere in the command string.

(continued on next page)

Command Format (cont)

2. General - Each switch in this category can appear only once in the command string because they affect the entire request. The switch used can appear anywhere in the command string.

3. File control - Any number of these switches can appear in the command string because they are specific to individual files within the request. The switch used must be adjacent to the file to which it applies. If the switch precedes the filename, it becomes the default for subsequent files.

The following switches can be used with the TPUNCH command.

Switch	Explanation	Category
/AFTER:tt	Process the request after the specified time; tt is either in the form of hhmm (time of day) or +hhmm (time later than the current time). The resulting AFTER time must be less than the DEADLINE time. If the switch, or the value of the switch, is omitted, no AFTER constraints are assumed.	General
/BEFORE:t	Queue only the files with a creation date before time t where t is in the form dd-mmm-yy.	General
/BEGIN:n	Start the output on the nth foot of tape. The default is to begin output on the first foot.	File Control
/COPIES:n	Repeat the output the specified number of times. N must be less than 64. If more than 63 copies are needed, two separate requests must be made. If this switch is omitted, one copy is made.	File Control
/CREATE	Make a new entry into the paper-tape output queue. This switch is the default for the queue-operation switches.	Queue Operation
/DEADLINE:tt	Process the request before the specified time; tt is either in the form hhmm (time of day) or +hhmm (time later than the current time). The resulting DEADLINE time must be greater than the AFTER time. If the switch, or the value of the switch, is omitted, no DEADLINE constraints are assumed.	General

(continued on next page)

TPUNCH command (Cont)

Command Format (cont)

Switch	Explanation	Category
/DISPOSE:DELETE	Delete the file after spooling.	File Control
/DISPOSE:PRESERVE	Save the file after spooling. This is the default of all files except files with extensions of .LST, .TMP, and if the protection is 0xx, .PTP.	File Control
/DISPOSE:RENAME	Rename the file from the specified directory immediately, remove it from the logged-out quota, and delete it after spooling. If omitted, this is the default for files with extensions .LST, .TMP, and if the protection is 0xx, .PTP.	File Control
/F	List the entries in the paper-tape punch queue, but do not update the queues. Therefore, the list may not be an up-to-date listing, but the listing will be faster than with /LIST.	Queue Operation
/FILE:ASCII	Indicate that the file format is ASCII text. This is the default.	File Control
/FILE:ELEVEN	Indicate that the file format is MACX11 binary format.	File Control
/KILL	Remove the specified entry from the paper-tape punch queue. This switch can be used for deleting a previously submitted request as long as the request has not been started by the paper-tape punch spooler.	Queue Operation

(continued on next page)

Command Format (cont)

Switch	Explanation	Category
/LIMIT:n	Limit the output to the specified number of feet. The default is 10*B+20 feet, where B is the number of blocks in the request.	General
/LIST	List the entries in the paper-tape punch queue; if the switch, along with all other switches, is omitted, all entries for all jobs of all users are listed.	Queue Operation
/MODIFY	Alter the specified parameters in the job. This switch requires that the user have access rights to the job. It can be used for altering a previously submitted request as long as the request has not been started by the spooler.	Queue Operation
/NEW	Accept the request even if the file does not yet exist.	File Control
/NOTE:a	Punch the specified text (a) in the output.	File Control
/NULL	Accept the request even if there is nothing in the request. No error message is given.	General
/OKNONE	Do not output message if no files match the wild-card construction. This is assumed at KJOB time.	File Control
/PHYSICAL	Suppress logical device name assignments for the device specified.	File Control
/PRIORITY:n	Assign the specified external priority (n = 0 to 62) to the request. The larger the number, the greater priority the job has. The default is 10 if no switch is given and 20 if the switch is specified without a value.	General
/PROTECT:nnn	Assign the protection nnn (octal) to the job. If the switch, or the value of the switch, is omitted, the standard protection is assumed.	General
/REMOVE	Remove the file from the queue. This switch is valid only with the /MODIFY switch and can be used to remove a previously submitted file as long as the spooler has not started processing the request.	File Control

(continued on next page)

TPUNCH command (Cont)

Command Format (cont)

Switch	Explanation	Category
/SEQ:n	Specify a sequence number to help identify a request to be modified or deleted.	General
/SINCE:t	Queue only the files with creation dates after the specified time t where t is in the form dd-mmm-yy.	General
/START:n	Begin on the nth line of the file. If the switch, or the value of the switch, is omitted, the spooler starts with the first line.	File Control
/STRS	Search for the file on all file structures in the search list and take each occurrence. The default is to take just the first occurrence.	File Control
/TAPE:ASCII	Punch the tape in ASCII code. If the /TAPE switch is not specified, the file is punched according to the data mode of the file.	File Control
/TAPE:BINARY	Punch the tape in binary mode. If the /TAPE switch is not specified, the file is punched according to the data mode of the file.	File Control
/TAPE:IBINARY	Punch the tape in image-binary mode. If the /TAPE switch is not specified, the file is punched according to the data mode of the file.	File Control
/TAPE:IMAGE	Punch the tape in image mode. If the /TAPE switch is not specified, the file is punched according to the data mode of the file.	File Control
/UNPRESERVED	Output the files only if they are not preserved (i.e., the first digit is 0). This switch avoids redundant printing.	General

Characteristics

The TPUNCH command:

Leaves the terminal in monitor mode.
Runs the QUEUE program, thereby destroying the user's core image.
Depends on FTQCOM which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.TPUNCH TENDMP.REL/TAPE:BINARY/COPIES:5 ⤶
```

Punch 5 copies, in binary mode, of the file DSK:TENDMP.REL.

TYPE command [1]

Function

The TYPE command directs PIP to type the contents of the named source file(s) on the user's terminal.

To stop the typing, type ↑C twice.

Command Format

TYPE list

list = a single file specification or a string of file specifications separated by commas. The filename is required. The extension is required if the filename has an extension.

In addition, the full wildcard construction can be used.

Switches can be passed to PIP by enclosing them in parentheses in the TYPE command string. When COMPIL interprets the command string, it passes the switches on to PIP.

Characteristics

The TYPE command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core area.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.TYPE FILEA,DTA0:FILEB.MAC↵
.TYPE *.TMP,DTA4:C↵
```

[1] This command runs the COMPIL program, which interprets the command before running PIP.

UNLOAD command [1]

Function

The UNLOAD command rewinds and unloads a magnetic tape or a DECtape. This command is equivalent to the following PIP command string:

```
dev:(MU) ←
```

Command Format

UNLOAD dev:

dev: = a magnetic tape (MTAn) or a DECtape (DTAn).

Characteristics

The UNLOAD command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.UNLOAD DTA7:)

.UNL MTA3:)
```

[1] This command runs the COMPIL program, which interprets the command before running the PIP program.

VERSION command

Function

The VERSION command prints the version number of the program in the user's core area (i.e., the last program run implicitly or explicitly). The version number is obtained from .JBVER and .JBHVR in the job data area and is printed in standard format. The output from this command is in one of the following representations:

low + high	The low and high segments are different.
low	There is only a low segment.
low +	The low and high segments are the same.
+[1]	A GETSEG UUO has been done to a high segment which matches the low segment.
+ high[1]	A GETSEG UUO has been done to a high segment which does not match the low segment.
blank[1]	The high segment has been released.

With the VERSION command, the low and high segments are represented in the format

name version

With the SET WATCH VERSION command, the low and high segments are represented in one of three formats:

name version	The program is not from SYS:
:name version	The output is the result of a SETNAM UUO (e.g., at the end of loading).
S:name version	The program is a system program (not logical device SYS:).

The name is a SIXBIT name and the version is in standard format. When printing the version number, the standard format is:

major version minor version (edit) - group who modified program last

[1]Output only from the SET WATCH VERSION command.

Function (cont)

The major version is octal; the minor version is alphabetic; the edit is octal and enclosed in parentheses; and the group who last modified the program is octal and preceded by a hyphen (0 = DEC development, 1 = all other DEC personnel, and 2-7 = customer use). There are no spaces separating the items, and if an item is zero, it does not appear in print. The parentheses and hyphen also do not appear in print if the corresponding item is zero. The following are examples of version numbers output in standard format.

10B(335)-1	major version 10, minor version B, edit number 335, group that modified program last 1.
7(5)	major version 7, minor version 0, edit number 5, group that modified program last 0.
54A	major version 54, minor version A, edit number 0, group that modified program last 0.

Command Format

VERSION

Characteristics

The VERSION command:

Leaves the terminal in monitor mode.
Depends on FTVERS which is normally absent in the DECsystem-1040.

Examples

```
.R TECO

*↑C

.VERSION
TECO 22(64) +

.TYPE SAMPL.TXT
THIS IS A TEXT FILE

.VERSION
PIP 31(35) +
```

WHERE command

Function

The WHERE command enables the user to determine the station at which a specific peripheral device is located. If the station of a particular terminal is requested, the number returned is the physical location of the terminal which may or may not be the location of the controlling job. This depends on whether the user changed his job's logical location with the LOCATE command.

Command Format

WHERE devn

dev = any physical device name and n is the unit number.

Characteristics

The WHERE command:

Leaves the terminal in monitor mode.
Does not require LOGIN.
Depends on FTREM which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.WHERE CDR2:)
1                                   ;station of CDR2.

.WH TTY:)
2                                   ;station of job's terminal.

.WHE OPR:)
4                                   ;station of job issuing command.

.WHERE CTY:)
1                                   ;central station.
```

ZERO command [1]

Function

The ZERO command clears the directory of the output device. This command is equivalent to the following PIP command string:

```
dev:/Z ←
```

Command Format

ZERO dev:

dev: = a DECtape (DTAn) or a disk (DSK). This argument is required.

A directory name can be specified with ZERO DSK: and if the user has access to the specified directory, the directory is zeroed. If no directory is specified, the user's directory is assumed.

Characteristics

The ZERO command:

Leaves the terminal in monitor mode.
Runs the PIP program, thereby destroying the user's core image.
Depends on FTCCLX which is normally absent in the DECsystem-1040.

Associated Messages

Refer to Chapter 4.

Examples

```
.ZER DTA4:)

.ZERO DSK:)

.ZER DSK: [27,40])
```

[1]This command runs the COMPIL program, which interprets the command before running the PIP program.

CHAPTER 3
BATCH SYSTEM COMMANDS

The Batch System, operating under the control of the DECsystem-10 Operating System, increases system throughput by processing jobs that do not require human interaction. Types of jobs best suited for a batch environment are: large and long-running jobs, jobs that require large amounts of data, frequently run production jobs, and jobs that require little or no interaction with the user. Up to 14 Batch jobs can be processed concurrently without adversely affecting the running of timesharing jobs. Batch jobs may be entered from

1. Local devices
2. Remote devices
3. Interactive terminals.

3.1 BATCH COMPONENTS

The Batch System consists of a group of programs; some are used for Batch operations only, others are available for various operations of the total computing system.

The individual Batch components are: the Stacker, CDRSTK; the Queue Manager, QMANGR; the Batch Controller, BATCON; and the output spoolers, LPTSPL (line printer), CDPSPL (card punch), PLTSPL (plotter), and PTPSPL (paper-tape punch).

3.1.1 The Stacker

The Stacker, CDRSTK, is responsible for

1. reading a sequential input stream from an input device,
2. separating the input by placing it in files according to the control cards contained in the input stream,
3. creating the job's log file and entering a report of its processing, and
4. entering the job into the Batch input queue.

When input is from the card reader, CDRSTK accepts ASCII, binary, 026, and DEC-029 Hollerith code. (Refer to Appendix B for tables of card codes.) The input is read in image mode, and CDRSTK converts it to one of the mentioned codes. If input is from any other device, only ASCII code is accepted.

CDRSTK creates three types of files during its copying of the input data: the user's data files, the Batch control file, and the job's log file. The data files are created according to the control cards in the input and are placed into the user's disk area. Programs and data are copied into these files and are passed to the job, while it is running, by the Batch Controller. Refer to Paragraph 3.3 for the description of the control cards that cause CDRSTK to copy information into these files.

A user control file is created for each valid job and is subsequently processed by the Batch Controller. This file contains all monitor level and user level commands encountered in the input. CDRSTK also enters commands resulting from the processing of certain control cards and any information that does not follow specific control card format. The control file is placed in the user's disk area. Refer to Paragraph 3.4 for a description of the Batch Controller commands that can be entered into the control file.

The job's log file contains a report of the CDRSTK's processing, along with a record of any operator intervention during its operation. This file is in the user's disk area along with the other CDRSTK-created files and is deleted after it is printed by the line printer spooler.

3.1.2 The Queue Manager

The Queue Manager, QMANGR, is the program that schedules jobs and maintains system queues. When CDRSTK finishes processing a job, it makes an entry into the Batch input queue. The Queue Manager computes and dynamically revises priorities for the job and notifies the Batch Controller when the job is to be run. Jobs are scheduled for running according to the parameters pertaining to each job and to the priorities established by the system. While the job is running, its queue entry is flagged to show it is in use, but the entry is not deleted from the queue until the job terminates. When the job is logged off the system, an output queue entry is usually made and the entry in the input queue is deleted. The Queue Manager again schedules the job's output and deletes the job's output queue entry only when the output is completely finished.

3.1.3 The Batch Controller

The Batch Controller, BATCON, controls all jobs entered into the Batch System. It reads the control file created by CDRSTK or the user and initiates and controls the running of the job by passing data and system program commands directly to it.

Monitor commands are examined by the Batch Controller and passed to the monitor for action. The Controller determines the destination of commands by interpreting the character in column 1 in each line of the control file. If column 1 contains a space or a tab, the spaces are ignored until a nonspace character is encountered. If column 1 contains an alphabetic or numeric character, the line is either at monitor command level or at user command level. If column 1 contains a special character, the Batch Controller interprets the line as follows:

$ (dollar sign) - The interpretation depends upon the character in column 2.

If column 2 contains an alphabetic character, the line is copied to the log file as a comment because it is a Stacker control line and has already been processed.

If column 2 contains a numeric or special character, the line is treated as data.

If column 2 contains a dollar sign ($), the initial dollar sign is suppressed and the line is treated as data.

If column 2 contains a line feed, vertical tab, or form feed, a blank line is entered into the log file.

(period) - The interpretation depends upon the character in column 2.

If column 2 contains an alphabetic character, the line is treated as a monitor command and the period is suppressed.

If column 2 contains a nonalphabetic character, the line is treated as data with the period as part of the data.

* (asterisk) - The line is treated as a user-level command or program data and the asterisk is suppressed. This is the standard input data method for most system programs.

= (equal sign) - The line is treated as a user level command or program data. The equal sign is suppressed and final spaces and the end of the line are suppressed (i.e., not passed to the program). This line normally indicates a DDT or TECO command because these commands terminate with special characters rather than the end of the line and would not function properly if the end of the line were passed.

; (semicolon) - The line is treated as a comment to the log file.

% (percent sign) - The line is treated as part of a command level statement label. The percent sign is normally reserved for DEC use. If % is encountered when the job has had no error, the control file is advanced, unless a %FIN is encountered. In this case, the %FIN is executed. Refer to the discussion of the .IF command in Paragraph 3.4.5.

The Batch Controller does not examine the contents of any lines in the control file other than those destined for the monitor. However, when it encounters an up-arrow (↑), it converts the up-arrow as follows:

If the character following the up-arrow is a numeric character, the up-arrow and the digit are passed to the job.

If the character following the up-arrow is an alphabetic character, the up-arrow and the character are translated to a control character; e.g., ↑A is translated to CTRL-A.

(continued on next page)

If the character following the up-arrow is another up-arrow, the first up-arrow is ignored and the second up-arrow is treated as an up-arrow; e.g., ↑↑A is treated as ↑A (up-arrow A) and ↑↑↑A is treated as ↑↑A (up-arrow up-arrow A).

If the job is requesting input and is at monitor level, the control file is read until a command or intermediate level line is found. If a job is requesting input at data level and the next line is a monitor command, the Batch Controller inserts a control-C.

A Batch user may not issue the following monitor commands when his job is operating in batch mode: ATTACH, DETACH, CCONT, CSTART, and SEND. If these commands are used, the line is suppressed and flagged at BATERR in the log file and the job is continued. All other monitor commands and system program commands may be used by a job operating in batch mode.

The Batch Controller makes entries to the log file to record its processing of the control file and the job.

3.1.4 The Output Spoolers

The output spoolers receive job output that has been placed into the output queues by the Queue Manager. Usually a job's output is placed in a line printer queue to be printed at a later time by the LPTSPL spooling program at the same station from which the input was received. The output filenames are in the form QxxSnn.LPT, where xx is a random number, and nn is the station number of the printer where the job is currently located. However, the user can also specify other output devices either in his programs within his job or by means of the QUEUE monitor command in his job. The first method causes output to the card punch, paper-tape punch, or plotter to be automatically spooled by the system. The second specifies nonstandard output spooling to any of the spooling devices.

3.2 SUBMITTING JOBS

A job is a unit that consists of one step or a group of steps. It can contain (1) a single program and its related data, or several programs and their data, and (2) the monitor and user-level commands that are required to control the programs.

The Batch system allows the user to submit his job by one of the following three methods:

1. The user punches his job on cards, inserts control cards to CDRSTK, and leaves his cards at the designated place for the operator to run (refer to Paragraph 3.2.1).
2. The user creates his job as a file for input to CDRSTK (instead of having his job on cards) and then runs CDRSTK himself (refer to Paragraph 3.2.2).
3. The user bypasses CDRSTK by creating his own control file on disk for the Batch Controller and then enters his job into the Batch input queue from his terminal (refer to Paragraph 3.2.3).

3.2.1 Submitting a Job with Cards

With this method, a job is submitted via a deck of cards, bounded by the control cards that mark its beginning and end. Other control cards to CDRSTK are interspersed among the card deck to direct CDRSTK's processing. Figure 3-1 shows a job containing the appropriate control cards to CDRSTK. This job compiles, loads, executes, and lists a FORTRAN program.

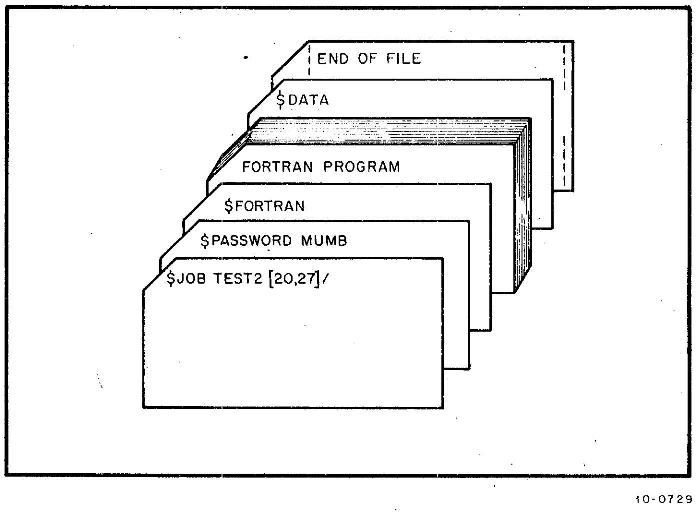


Figure 3-1 Typical Job on Cards

3.2.1.1 The $JOB Card - This card notifies CDRSTK that a job is to be processed. CDRSTK creates a control file into which commands are placed for the Batch Controller and a log file on the disk. The first argument (TEST2) shown on this card is the user-assigned name for the job; the second argument ([20,27]) is the project-programmer number of the user. For a description of switches which can be used on this card, refer to Paragraph 3.3.9.

3.2.1.2 The $PASSWORD Card - This card contains the PASSWORD associated with the project-programmer number specified on the $JOB card. In Figure 3-1, the PASSWORD is MUMB, which was assigned to the user by the system manager. Refer to Paragraph 3.3.12 for more information on the $PASSWORD card.

3.2.1.3 The $FORTRAN Card - This card causes CDRSTK to insert a COMPILE monitor command (refer to Chapter 2) into the control file in order to cause the program to be compiled. Immediately following the $FORTRAN card is the FORTRAN source program to be compiled. The source program is read into a disk file with the specified filename (or a default name if a filename is not given) and with an extension of .F4. Refer to Paragraph 3.3.8 for more information on the $FORTRAN card.

3.2.1.4 The $DATA Card – The card after the FORTRAN program is the $DATA card. This card causes CDRSTK to insert an EXECUTE monitor command (refer to Chapter 2) into the control file in order to load and then execute the previously compiled program. Refer to Paragraph 3.3.3 for additional information on this card.

3.2.1.5 The End of File Card – The last card shown in the example is the end-of-file card. This card signals the end of the job. The card is recognized by CDRSTK as the end of the file because of the punches in rows 12, 11, 0, 1, 6, 7, 8, 9 in columns 1 and 80 of the card. Refer to Paragraph 3.3 for more information about this card.

3.2.1.6 Output – Once the program is punched on cards, the card deck is submitted to the operator, who in turn stacks the job in the card reader. The user receives his output in the form of line printer listings. Refer to Paragraph 3.5 for an explanation of the job output.

The CDRSTK control cards shown in Figure 3-1 are just a few of the control cards available to the user. For a complete description of all the CDRSTK control cards, refer to Paragraph 3.3.

3.2.2 Submitting a Job with a File

With this method, a job is submitted via a file contained on any input device that supports ASCII code. This file contains the program and data with card images of the control cards for CDRSTK. The following example shows the creation of a disk file containing a FORTRAN program and card images of CDRSTK control commands. Note that it corresponds to the card example in Paragraph 3.2.1.

```
.LOGIN 2Ø,27↲
JOB17 5SØ4 TTY11
PASSWORD:
1Ø2Ø 15-MAR-72 WED
.MAKE JOB↲
*I $JOB TEST2, [2Ø,27]↲
$FORTRAN↲
C FORTRAN PROGRAM GOES HERE
$EOD↲
$DATA↲
$$
*EX$$
.
```

3.2.2.1 Image of the $JOB Card – The first line of the file is an image of the $JOB card. Note that the $ character must be the first character of the line in order for CDRSTK to recognize it as a control command. This line causes a control file and a log file to be created on the disk when CDRSTK is run. The first argument (TEST2) is the user-assigned name for the job; the second ([20,27]) is the project-programmer number of the user. The $PASSWORD card image is not needed because the user is already logged-in when creating the input file. For additional information on the $JOB card, refer to Paragraph 3.3.9.

3.2.2.2 Image of the $FORTRAN Card - This line causes CDRSTK to insert a COMPILE monitor command (refer to Chapter 2) into the control file in order to compile the program. The source program follows immediately and is read into a disk file with the specified filename (or a default name if a filename is not given) and with an extension of .F4. Refer to Paragraph 3.3.8 for more information on the $FORTRAN card.

3.2.2.3 Image of the $EOD Card - This line indicates to CDRSTK the end of the FORTRAN program. Refer to Paragraph 3.3.6 for more information.

3.2.2.4 Image of the $DATA Card - This line causes CDRSTK to insert an EXECUTE monitor command (refer to Chapter 2) into the control file in order to load and execute the program.

3.2.2.5 Running CDRSTK - Once the file is created and CDRSTK is run by the user, it processes the user-created file in the same manner as it processes input files of jobs entered directly by the operator. The user runs CDRSTK by typing

```
        .R CDRSTK )
```

CDRSTK responds with an asterisk, and then the user types in the following command

```
        *START dev:file.ext)
```

where dev: is the name of the device containing the input file for CDRSTK and file.ext is the name of the file. Using the above file, the command is

```
        *START DSK:JOB )
```

and CDRSTK responds with

```
        !
```

When CDRSTK has completed its processing (i.e., when it has created the control and log files and has entered the job into the Batch input queue), it responds with

```
        READY
        *
```

indicating its readiness to accept another file. At this point, the user can enter another file or return to monitor mode with a ↑C.

The card images shown in the preceding example are only a few of the CDRSTK control card images available. Refer to Paragraph 3.3 for a complete description of all of the control cards.

3.2.3 Submitting a Job with a Control File to the Batch Controller

With this method, a job is submitted via the steps within a control file to the Batch Controller. The file must be a disk file and is created with a system editor. Since this file is processed directly by the Batch Controller, control card images are not used. The control file consists of monitor commands, user program commands, comments, and sequence control statements. Refer to Paragraph 3.4 for a description of control file commands. The following is an example of creating a control file. It assumes that a file named DATA.F4 already exists on disk.

```
.MAKE JOB.CTL ↵
*I.EXECUTE /COMPILE DATA.F4 /LIST ↵
$$
EX$$
```

Once the control file is created, the user can enter the job into the Batch input queue one of three ways:

1. SUBMIT jobname = control file, log file
   refer to the SUBMIT command in Chapter 2.
2. QUEUE INP: jobname = control file, log file
   refer to the QUEUE command in Chapter 2.
3. R QUEUE
   refer to the QUEUE specification in Notebook 7 of the DECsystem-10 Software Notebooks.

3.2.4 Interjob Dependency

Jobs are not necessarily run in the order that they are read into the Batch System. Priorities stipulated by the user on the $JOB card (refer to Paragraph 3.3.9) and additional parameters set by the Batch System are dynamically computed by the Queue Manager to determine in what order the jobs are run. However, it is often useful to submit several jobs that must be run in a specific order, for example, one job updates a master file and another job processes it. Therefore, the running of one job is dependent upon the running of the other. Although these jobs could be combined into one large job, it is sometimes necessary to keep them distinct; i.e., they might be submitted by different people at different times. Because the jobs in the Batch System are run in order of priority, the user specifies an additional priority, an initial dependency count, on the $JOB card of the dependent job. This dependency count becomes part of the queue entry. Any input queue entry that has a dependency count greater than zero cannot be scheduled. When the count becomes zero, the job is scheduled, based upon the time it was submitted and the time that the dependency count became zero. If the dependency count becomes negative, an advisory message is sent to the issuing job and to the dependent job. The dependency count can be altered by including the QUEUE command as part of any job upon which the dependent job is waiting. (Refer to the QUEUE monitor command.) The QUEUE command switch that allows the user to change the dependency count of another job is the /MODIFY/DEPEND:nn switch. If the user specifies a plus or minus sign before the count (nn), that number is subtracted from or added to the dependent job's count. If the user does not specify a sign, the dependent job's count is changed to the count specified in the /MODIFY/DEPEND: switch.

3.3 CDRSTK CONTROL CARDS

Control cards are interspersed among the input stream to aid CDRSTK in separating the input into the appropriate files, either the user's data files or the control file processed by the Batch Controller. The control cards contain a dollar sign ($) in column 1 and an alphabetic character in column 2. These are the only cards read and interpreted by CDRSTK; the remainder of the input is separated and placed into the appropriate file. Note that if the user creates his own control file, he bypasses CDRSTK, and, therefore, does not use these control cards.

Only the first part of the command name or switch need be specified; as long as the name is unique, it is accepted. The first three characters of a command name are generally sufficient to ensure uniqueness. The standard comment and continuation conventions for the system can be used on the control card. A comment is preceded by a semicolon; characters after the semicolon to the end of the card are treated as comments. A card may be continued by placing a hyphen as the last non-TAB or non-space character before the end of the card. Comments beginning with a semicolon, TABs, and spaces can follow. All defaults for control card parameters are installation parameters.

The end-of-file is used to signal the end of the job. When input is from the card reader, an end-of-file card is used. Column 1 of this card has rows 4 and 5 blank and rows 6, 7, 8, and 9 punched. The recommended form of this card has columns 1 and 80 containing punches in rows 12, 11, 0, 1, 6, 7, 8, 9, with rows 2, 3, 4, and 5 blank, so that the card can be recognized in any orientation. When devices other than the card reader are used for input, the standard end-of-file for each device is treated by CDRSTK as the end of the job.

3.3.1

$ALGOL

Function

This card causes CDRSTK to copy the named ALGOL program onto disk and to insert a COMPILE monitor command into the control file. The card is placed at the beginning of the source program. When the job is run, the specified program is compiled and temporary relocatable binary and listing files are created. The binary and listing files can be made permanent if the user renames them to change their protection. The source file can be preserved by means of the /PROTECT switch. The listing file is printed as part of the job's output.

Processor switches can be passed to the ALGOL compiler by including them in the command string. The position of these switches in the command determines their position in the COMPILE command generated by CDRSTK. For example

```
$ALGOL /NOLIST (E,,N)
```

results in the following COMPILE command

```
.COMPILE /COMPILE DECKAA.ALG /NOLIST (E,,N)
```

Refer to Paragraph 1.5.7 for a description of the ALGOL processor switches.

Card Format

$ALGOL dev:name.ext [proj,prog] (processor switches) $/S_1/S_2.../S_n$

dev: = a file structure name. If omitted, DSK is assumed.

name.ext = the name of the file to be created on disk. If omitted, CDRSTK assigns a unique filename of the form DECKaa (where aa = AA through ZZ) with the extension .ALG. It is recommended, however, that the user select a distinct filename for each job that is in the Batch system simultaneously.

[proj,prog] = a directory name other than that specified on the $JOB card. If omitted, the project-programmer number on the $JOB card is used.

(processor switches) = the switches to be passed to the ALGOL compiler. They must be enclosed in parentheses and the slash cannot appear in connection with these switches.

$/S_1/S_2.../S_n$ = the switches that control the mode of input interpretation and the listing of the compiled program.

Switch	Meaning	Default
/ASCII	The input is read in ASCII mode.	on
/D029	The card deck is read in the old DEC-029 format. This format is similar to ASCII and is available only in those installations that use DEC-029 format.	off
/LIST	A temporary listing file of the program is created.	on

(continued on next page)

Card Format (cont)

Switch	Meaning	Default
/NOLIST	No listing file of the program is created.	off
/PROTECT:nnn	The protection to be set for the file (in octal).	The file is preserved only until KJOB for the job.
/SUPPRESS: ON or OFF	When ON is specified, trailing blanks are suppressed. They are not suppressed when OFF is specified.	on
/WIDTH:nn	The maximum number of columns to be read. If the specified width is less than 80, only that number of columns is read. The remaining columns are treated as blank. Normally this switch is only used when the /SUPPRESS switch is on.	80
/026	The card deck is read in 026 card code.	off

Restrictions

The /026 and /D029 switches apply only to card reader input. Input from other devices must be read in ASCII code; otherwise an error message is written in the log file and the job is terminated.

3.3.2 $COBOL

Function

This card causes CDRSTK to copy the specified COBOL program onto disk. The card is placed at the beginning of the source program, and when CDRSTK reads the card, it inserts a COMPILE monitor command into the control file and copies the COBOL program into the file on the specified disk area. When the job is run, the program is compiled and a temporary relocatable binary file and a temporary listing file are created. The binary and listing files can be made permanent if the user renames them to change their protection. The source file can be preserved if the user specifies the /PROTECT switch. The listing file is printed as part of the job's output.

Processor switches can be passed to the COBOL compiler by including them in the command string. The position of these switches in the command determines their position in the COMPILE command generated by CDRSTK.

For example

```
$COBOL (A,M,C) /PROTECT:057
```

results in the following COMPILE command

```
.COMPILE /COMPILE DECKAB.CBL (A,M,C) /LIST
```

Refer to Paragraph 1.5.7 for a description of the COBOL processor switches.

Card Format

$COBOL dev:name.ext [proj,prog] (processor switches) /S_1/S_2.../S_n

dev: = a file structure name. If omitted, DSK is assumed.

name.ext = the name of the file to be created on disk. If omitted, CDRSTK assigns a unique filename of the form DECKaa (where aa = AA through ZZ) with the extension .CBL. It is recommended, however, that the user select a distinct name for each job in the Batch system simultaneously.

[proj,prog] = a directory name other than that specified on the $JOB card. If omitted, the project-programmer number on the $JOB card is used.

(processor switches) = the switches to be passed to the COBOL compiler. They must be enclosed in parentheses and the slash cannot appear in connection with these switches.

/S_1/S_2.../S_n = the switches that control the mode of input interpretation and the listing of the compiled program.

Switch	Meaning	Default
/ASCII	The input is read in ASCII mode.	on
/D029	The card deck is read in the old DEC-029 format. This format is similar to ASCII format and is available only in those installations that use DEC-029 format.	off
/LIST	A temporary listing file of the program is created.	on
/PROTECT:nnn	Specifies the protection to be set for the job (in octal).	The file is preserved only until KJOB for the job.
/SEQUENCE	The program is read in conventional COBOL format with sequence numbers in columns 1 through 6, and comments beginning in column 73. When this switch is specified, the default width is 72.	on
/SUPPRESS: ON or OFF	When ON is specified, trailing blanks are suppressed. They are not suppressed when OFF is specified.	on
/WIDTH:nn	The maximum number of columns to be read. If the specified width is less than 80, only that number of columns is read, the remaining columns are treated as blank. Normally this switch is used only when the /SUPPRESS switch is on.	80
/026	The card deck is read in 026 card code.	off

Restrictions

The /026 and /D029 switches apply only to card reader input. Input from other devices must be read in ASCII code; otherwise, an error message is written in the log file and the job is terminated.

$DATA

Function

This card causes CDRSTK to copy data into a file on the user's disk area and to insert an EXECUTE monitor command into the control file.

CDRSTK maintains a list of filenames of all source or relocatable programs that have been processed since the beginning of the job or the last $DATA card read. Each time a program is copied by the CDRSTK, its name is placed in the list and given an extension of .REL. When the $DATA card is read, CDRSTK places an EXECUTE command into the control file and copies the filenames of the programs into the EXECUTE command string. On the next $language card, CDRSTK clears the list of filenames so that the next entries into the list reflect only those filenames copied since the last $DATA command was read. When the job is run, the programs are loaded and executed. No compilation is performed because the programs are either in relocatable binary form or have been previously compiled because of the $language card. If two $DATA cards appear in a row, the same programs are reloaded and executed again.

Loader switches can be passed to the LOADER by placing them in the command string. When CDRSTK places these switches in the EXECUTE command, it converts them to the standard LOADER switch format (i.e., % switch). For instance,

```
$DATA (S,F)
```

causes the following EXECUTE command to be generated

```
.EXECUTE ... %S %F
```

Card Format

$DATA dev:name.ext [proj,prog] $/S_1/S_2.../S_n$

dev: = a file structure name. If omitted, DSK is assumed.

name.ext = the filename of the file to be created. If omitted, CDRSTK creates a unique filename of the form Qaa (aa = AA through ZZ) with the extension .CDR.

It is recommended, however, that the user select a distinct name for each job that is in the Batch system simultaneously, so that he can distinguish the various output listings.

[proj,prog] = the directory name if different from the one specified on the $JOB card. If omitted, the project-programmer number specified on the $JOB card is used.

$/S_1/S_2.../S_n$ = switches that control the mode of reading and interpreting of the input media.

Card Format (cont)

Switch	Meaning	Default
/ASCII	The input stream is read in ASCII mode.	on
/BINARY	The card deck is read in binary card form. This switch is ordinarily not necessary because the first column of each card is checked for punches in rows 7 and 9. If these rows are punched, the card is read in binary.	off
/D029	The card deck is read in the old DEC-029 format. This format is similar to ASCII format and is available only for compatability with old decks.	off
/IMAGE:n	The card deck is read in image mode. The switch must be followed by a decimal number in the range 2 through 80. This causes ensuing cards to be read in image mode until either end-of-file is reached or a card is read that contains punches in all rows of column 1 and all rows in column n. The CDRSTK control commands are not recognized when cards are read in image mode.	off
/PROTECT:nnn	A protection of nnn (octal) is set for the file; if not specified, the file is preserved only until a KJOB command for the job.	
/SUPPRESS: ON or OFF	When ON is specified, trailing spaces are suppressed. They are not suppressed when OFF is specified.	on
/WIDTH:nn	The maximum number of columns to be read. If the specified width is less than 80, only that number of columns is read. The remaining columns are treated as blanks. Normally, this switch is only used when the /SUPPRESS switch is on. For example, /WIDTH:72 causes the CDRSTK to disregard columns 73 through 80 and to suppress any trailing spaces up to column 72.	80
/026	The card deck is read in 026 card code.	off

The modes ASCII, 026, IMAGE, and D029 are mutually exclusive modes for interpreting Hollerith punches. When one of those modes is set, it remains in effect until changed (refer to the $MODE card) or the end of file is reached.

The defaults for all modes are reset by the next $MODE card or by individual switches in other control cards such as in the $DECK card.

Requirements

If the data is contained within the programs instead of being a separate file, a $DATA card or an EXECUTE command must be placed after the programs. The program will not be executed otherwise.

Restrictions

This card can be used only when the programs in the job have been entered with a $language card or $RELOCATABLE card, since CDRSTK maintains a list of the filenames of programs that are input with these commands. If the user wishes only to have the programs compiled, no $DATA card or EXECUTE command should appear in the job.

3.3.4

$DECK

Function

This card causes CDRSTK to copy all statements up to the next control card into a data file.

Card Format

$DECK dev:name.ext [proj,prog] /S_1/S_2.../S_n

dev: = a file structure name. The default is normally DSK.

name.ext = the user-assigned name and extension of the file to be created. If omitted, a unique filename in the form DECKaa (aa = AA through ZZ) is created by CDRSTK with the extension .CDR. It is recommended, however, that the user select a distinct name for each job that is in the Batch system simultaneously.

[proj,prog] = a disk area other than the one supplied on the $JOB card. If omitted, the project-programmer number specified on the $JOB card is used.

/S_1/S_2.../S_n = switches that control the mode of reading and interpreting of the input media. The switches are identical to the switches described for the $DATA card.

Restrictions

The /BINARY, /026, /IMAGE, and /D029 switches apply only to card reader input. Input from other devices must be read in ASCII code; otherwise an error message is written in the log file and the job is terminated.

3.3.5

$DUMP

Function

This card causes CDRSTK to insert a DUMP monitor command into the control file which invokes a dump, according to the arguments specified, when an error is detected by the Batch Controller.

Card Format

$DUMP /command arg/command arg...

The commands and their arguments are the same as described for the DUMP program. (Refer to Chapter 2). Two of the commands useful to a Batch job are duplicated below.

Command	Argument	Meaning
ALL		Dumps the entire file.
DUMP	dump descriptor, dump descriptor,...	Dumps the specified bytes in the current modes.

3.3.6

$EOD

Function

This card terminates the input that is being copied into a data file by CDRSTK because of a preceding $DECK card. All control cards with the exception of $MODE perform this action, i.e., terminate the copying of input. If input is not being copied and this card is read, CDRSTK ignores it. $EOD is only necessary when the user wishes to place a line of input which is not a CDRSTK control card after input that is being copied into a data file.

Card Format

$EOD

3.3.7

$ERROR
$NOERROR

Function

These cards are used to aid the Batch Controller in processing errors. They cause CDRSTK to insert an .IF statement into the control file; e.g., .IF (ERROR) or .IF (NOERROR). Refer to Paragraph 3.4.5 for an explanation of the .IF statement. These cards must appear at the point at which the error occurs.

Card Formats

$ERROR statement

$NOERROR statement

statement = an executable monitor or batch command preceded by a period. If the statement directs the Batch Controller to go to a statement label, the statement label line and any related lines must be included in the sequence of commands at the place the user wants it executed. For example,

```
$FORTRAN TEST1
      .
      .
      .
$ERROR .GOTO A
$DATA TEST1DA
      .
      .
      .
$ERROR .GOTO A
A: CONT
$FORTRAN TEST2
      .
      .
      .
```

3.3.8

$FORTRAN or $F40

Function

This card causes CDRSTK to copy the named FORTRAN program onto disk and to insert a COMPILE monitor command into the control file. The card is placed at the beginning of the source program. When the job is run, the specified program is compiled and temporary relocatable binary and listing files are created. The binary and listing files can be made permanent if the user renames them to change their protection. The source file can be preserved by means of the /PROTECT switch. The listing file is printed as part of the job's output.

Processor switches can be passed to the FORTRAN compiler by including them in the command string. Their position in the command string determines their position in the COMPILE command generated by CDRSTK. For example,

```
$FORTRAN (A,S,D) /NOLIST
```

results in the following COMPILE command

```
.COMPILE /COMPILE DECKII.F4 (A,S,D) /NOLIST
```

Refer to Paragraph 1.5.7 for a description of the FORTRAN processor switches.

Card Format

$FORTRAN dev:name.ext [proj,prog] (processor switches) $/S_1/S_2.../S_n$

$F40 dev:name.ext [proj,prog] (processor switches) $/S_1/S_2.../S_n$

dev: = a file structure name. If omitted, DSK is assumed.

name.ext = the name of the file to be created on disk. If omitted, CDRSTK assigns a unique filename of the form DECKaa (where aa = AA through ZZ) with the extension .F4. It is recommended, however, that the user select a distinct name for each job in the Batch system simultaneously.

[proj,prog] = a directory name other than that specified on the $JOB card. If omitted, the project-programmer number on the $JOB card is used.

(processor switches) = the switches to be passed to the FORTRAN compiler. They must be enclosed in parentheses and the slash cannot appear in connection with these switches.

$/S_1/S_2.../S_n$ = the switches that control the mode of input interpretation and the listing of the compiled program.

Switch	Meaning	Default
/ASCII	The input is read in ASCII mode.	on
/CREF	A cross-referenced listing file is created to be processed by the CREF program.	off
/D029	The card deck is read in the old DEC-029 format. This format is similar to ASCII and is available only in those installations that use DEC-029 format.	off
/LIST	A temporary listing file of the program is created.	on
/M	The MACRO coding is included in the output listing.	off
/NOLIST	No listing file of the program is created.	off
/PROTECT:nnn	The protection to be set for the file (in octal).	The file is preserved only until KJOB for the job.
/SUPPRESS: ON or OFF	When ON is specified, trailing blanks are suppressed. They are not suppressed when OFF is specified.	
/WIDTH:nn	The maximum number of columns to be read. If the specified width is less than 80, only that number of columns is read. The remaining columns are treated as blank. Normally this switch is only used when the /SUPPRESS switch is on.	72
/026	The card deck is read in 026 card code.	off

Restrictions

The /026 and /D029 switches apply only to card reader input. Input from other devices must be read in ASCII code; otherwise, an error message is written in the log file and the job is terminated.

$JOB

Function

This card, in conjunction with the $PASSWORD card (if required), causes CDRSTK to create a control file and a log file on disk into which commands are placed for the Batch Controller. The filename of the control file is the name of the job specified in the command string, and the extension is .CTL. CDRSTK also uses this name as the filename of the log file with an extension of .LOG. If the jobname is omitted from the command string, CDRSTK creates a unique name for the control file and log file. It is recommended, however, that the user select a distinct name for each job that is in the Batch system simultaneously, so that he can distinguish the various output listings. In general, the jobname used on input appears in the output queues. CDRSTK adds the control and log files to the directory of the specified project-programmer number.

The user may specify a wildcard designation (#) for the programmer number in the $JOB card, for example,

```
$JOB FLEX[4,#] or $JOB FLEX [4]
```

This specification causes CDRSTK to look at ACCT.SYS (an administrative file) in order to determine if the wildcard option is allowed for this project. If it is, CDRSTK provides a unique programmer number within the project. If it is not allowed, CDRSTK returns an error message and continues with the next job.

Card Format

$JOB name [proj,prog] /$S_1$/$S_2$.../$S_n$

name = the user-assigned name for the job; if omitted, CDRSTK creates a unique name of the form JOBaaa (aaa = AAA through ZZZ) for the control and log files.

[proj,prog] = the project-programmer number of the user who submitted the job. This argument is required. A space or comma can separate this argument from the jobname.

/S_1/S_2.../S_n = switches taken from the following group. These switches are optional.

Switch	Meaning	Default
/AFTER:dd-mmm-yy hhmm	The job cannot be run until after the specified date and time. The resulting AFTER time must be less than the DEADLINE time.	None
/AFTER:+tt	The job cannot be run until after the input time plus the number of minutes indicated by tt.	None
/CARDS:nnk	The maximum number of cards (up to 10K) that can be punched by the job (in decimal). K is optional.	0

(continued on next page)

Card Format (cont)

Switch	Meaning	Default
/CHARGE:aa	The job is charged to a user-specified account (aa = name of the account).	None
/CORE:nnk	Maximum amount of core (in decimal) that can be used by the job up to the maximum allowed by the installation. K is optional.	25K
/DEADLINE:dd-mmm-yy hhmm	The job must be completed by the specified date and time. The resulting DEADLINE time must be greater than the AFTER time.	None
/DEADLINE:+tt	The job must be started by the indicated number of minutes after it is input.	None
/DEPEND:nn	Initial interjob dependency count (in decimal).	0
/FEET:nn	The number of feet of paper tape that will be punched by the job.	0
/LOCATE:Snn	Specifies the remote station of the job and where the output is to be sent.	The station where the cards were input.
/NAME:aa	The user's name in up to 12 characters.	None
/PAGES:nn	The maximum number of pages in decimal to be printed by the job, including the log file and compilation listing.	100
/PRIORITY:nn	The external priority of the job; the highest priority that can be specified is 62 (decimal).	0
/PROTECT:nnn	The protection (in octal) for the job, the control file, and the log file.	Preserved only until KJOB is given for the job.
/RESTART:0 or 1	If 0, the job cannot be restarted by the operator. The job can be restarted if 1 is specified. The job should not be restartable if there are changes to the permanent file directory.	0
/TIME:hh:mm:ss	The limit placed on the amount of CPU time used by the job.	5.0 (5 minutes)
/TPLOT:mm	The number of minutes of plotter time that the job will use.	0
/UNIQUE:0 or 1	If 1, only one Batch job at a time is run using the specified directory. If 0, any number of Batch jobs can be run at the same time using the specified directory.	1

Requirements

The $JOB card must immediately follow the $SEQUENCE card, or be the first card if the $SEQUENCE card is not required.

$MACRO

Function

This card causes the CDRSTK to copy the designated MACRO program onto disk and is placed at the beginning of the source program. When CDRSTK reads the card, it inserts a COMPILE monitor command into the control file and copies the MACRO program into the file on the specified disk area. When the job is run, the program is assembled and a temporary relocatable binary file and listing files are created. The binary and listing files can be made permanent if the user renames them to change their protection. The source file is preserved by means of the /PROTECT switch. The listing file is printed as part of the job's output.

Processor switches can be passed to the MACRO assembler by including them in the command string. Their position in the command string determines their position in the COMPILE command generated by CDRSTK. For example

```
        $MACRO /PROTECT:055 (W,S,Q)
```

results in

```
        .COMPILE /COMPILE DECKCB.MAC /PROTECT:055 (W,S,Q) /LIST
```

Card Format

$MACRO dev:name.ext [proj,prog] (processor switches) $/S_1/S_2.../S_n$

dev: = a file structure name. If omitted, DSK is assumed.

name.ext = the name of the file to be created on disk. If omitted, CDRSTK assigns a unique filename in the form DECKaa (aa = AA through ZZ) with the extension .MAC. However, it is recommended that the user select a distinct name for each job in the Batch system simultaneously.

[proj,prog] = a directory name other than that specified on the $JOB card. If omitted, the project-programmer number on the $JOB card is used.

(processor switches) = the switches to be passed to the MACRO assembler. They must be enclosed in parentheses and the slash cannot appear in connection with these switches.

$/S_1/S_2.../S_n$ = the switches that control the mode of input interpretation and the listing of the assembled program.

Switch	Meaning	Default
/ASCII	The input is read in ASCII mode.	on
/CREF	A cross-referenced listing file is created to be processed by the CREF program.	off

(continued on next page)

Card Format (cont)

Switch	Meaning	Default
/D029	The card deck is read in the old DEC-029 format. This format is similar to ASCII and available only in those installations that use DEC-029 format.	off
/LIST	A temporary listing file of the program is created.	on
/NOLIST	No listing file of the program is created.	off
/PROTECT:nnn	The protection to be set for the file (in octal).	The file is preserved only until KJOB for the job.
/SUPPRESS: ON or OFF	When ON is specified, trailing blanks are suppressed. When OFF is specified, they are not suppressed.	on
/WIDTH:nn	The maximum number of columns to be typed. If the specified width is less than 80, only that number of columns is read. The remaining columns are treated as blank. Normally, this switch is used only when the /SUPPRESS switch is on.	80
/026	The card deck is read in 026 card code.	off

Restrictions

The /026 and /D029 switches apply only to card reader input. Input from other devices must be read in ASCII code; otherwise, an error message is written in the log file and the job is terminated.

3.3.11

$\boxed{\text{\$MODE}}$

Function

This card causes CDRSTK to change the mode in which it is interpreting the input stream. The $MODE card can be placed anywhere after the $PASSWORD card in the command sequence and is terminated by another $MODE card or the end-of-file (which terminates the job). This command does not terminate the copying of input preceded by a $DECK card.

Card Format

$MODE /S$_1$/S$_2$.../S$_n$

/S$_1$/S$_2$.../S$_n$ = switches that control the mode of reading and interpreting of the input media. These switches are identical to the switches described for the $DATA card.

Restrictions

The mode switches /026, /IMAGE, /D029, and /BINARY can be used only for card input. Input from other devices is always read as ASCII code. Thus, the only switches that can be used with the $MODE card for devices other than the card reader are /SUPPRESS and /WIDTH.

3.3.12

$PASSWORD

Function

This card contains the password associated with the project-programmer number specified in the $JOB card. If the password does not match the password stored in the system for the specified project-programmer number, CDRSTK does not create any files and aborts the job. Use of this command is an installation option.

Card Format

$PASSWORD password

password = 1 to 6 character password.

Requirements

If the $PASSWORD card is required, it must immediately follow the $JOB card.

3.3.13

$RELOCATABLE

Function

This card causes CDRSTK to copy a relocatable binary program from cards to a file on the user's disk area. The cards are read in binary mode.

Card Format

$RELOCATABLE dev:name.ext [proj,prog] /S_1

dev: = a file structure name. If omitted, DSK is assumed.

name.ext = the name of the file into which the program is copied. If the filename is omitted, CDRSTK creates a unique name in the form DECKaa (aa = AA through ZZ). It is recommended that the user select a distinct name for each job in the Batch system simultaneously. If the extension is omitted, .REL is assumed.

(continued on next page)

Card Format (cont)

[proj,prog] = the disk directory if different from the one specified on the $JOB card. If omitted, the project-programmer number on the $JOB card is assumed.

$/S_1$ = /PROTECT:nnn (octal)

The protection for the file to be created. If not specified, the file is preserved only until a KJOB command for the job is executed.

Restrictions

Relocatable binary programs can only be read when the input is from cards.

The program following this command must be read in binary; the mode cannot be changed until a nonbinary file is copied. If an attempt is made to change the mode, an error message will be issued and the job will be aborted.

3.3.14 $SEQUENCE

Function

This card specifies the job's unique sequence number. The use of this card depends on the requirements of the particular installation.

Card Format

$SEQUENCE n

n = a decimal number

Requirements

If the installation requires this command, it must be the first card in the input stream.

3.4 BATCON CONTROL FILE COMMANDS

Ordinarily the Batch Controller reads the control file in a sequential manner. The commands described in this section can appear in the control file to interrupt the sequential processing of the control file in order to specify error recovery. If an error occurs in the job, the Batch Controller is notified of the error; the user has the option of including several methods of error recovery.

The user may include an .IF command in the control file. When the error occurs, the Batch Controller examines the next monitor level line in the control file for an .IF command to determine what action to take on the error. It does not search past the next executable monitor line in the control file for the .IF command; therefore, if this command is used, it must be the next monitor command in the control file.

If the user does not wish to include an .IF command, he may include two types of error recovery routines in the control file, one type labeled %ERR (error processing for non-system programs) and the other labeled %CERR (error processing for compilers and system programs). A system program is one found on a device specified in the SYS search list in [1,4]. If SYS is assigned as a logical device name, the programs are considered user programs, not system programs. After an error occurs in the job and the next executable monitor line in the control file is not an .IF statement, the Batch Controller searches for the labeled error recovery control lines and processes the statements following these labels. These routines may be placed anywhere in the control file. Once the Batch Controller has processed the routine, it continues from that point in the control file; it does not read backwards over sections of the control file skipped in searching for the error routines. The following example shows the use of a %ERR error recovery routine.

```
.COMPILE SAMPLE /LIST↵
.MOUNT MTA:3 /VID:42936↵
.EXECUTE↵
.DISMOUNT 3↵
.R SORT↵
*MUMP.SRT←FOR04.DAT/R80/K1.10↵
.QUEUE MUMP.SRT↵
%ERR:.CLOSE↵
.DUMP↵
.DISMOUNT 3↵
%FIN:.DELETE FOR04.DAT↵
```

Depending on the type of error found, the following operations are performed. If a compilation error occurs, only the compilation and the listing result. No tape is mounted. If an execution error results,

1. the program is compiled,
2. the tape is mounted,
3. the program begins execution,
4. the output is closed,
5. a quick dump of core is taken,
6. the tape is dismounted, and
7. the file FOR04.DAT is deleted.

If a SORT error occurs, the program compiles, the tape is mounted, the program is executed, and the file FOR04.DAT is deleted. Finally, if no errors result,

1. the program is compiled,
2. the tape is mounted,
3. the program is executed,
4. the tape is dismounted,
5. the sort is performed,
6. MUMP.SRT is printed, and
7. the file FOR04.DAT is deleted.

When the user is bypassing CDRSTK and creating his own control file, he may place a %FIN at the end of the control file. (CDRSTK, in creating the control file, automatically places a %FIN at the end.) This label is used for cleanup purposes, e.g., deleting the input files. In creating the control file, the user may place other %FIN's at various points in the file for periodic cleanup of his job. For example, this label is used in a special kind of error recovery. If the time allocated to the job runs to the maximum limit specified in the $JOB command (refer to Paragraph 3.3.9) or by the Batch system, the user is given an additional 10% of his allocated time to cleanup his job before it is aborted. Because the user includes a %FIN, cleanup is performed and the results of the job's processing are not lost when the job is aborted. The user should be careful in using the %FIN in the control file because if the Batch Controller is searching for an error recovery routine and %FIN is placed before a %ERR or %CERR, the %FIN is executed and the Batch Controller assumes the error recovery routine has been satisfied and does not search any longer for %ERR or %CERR. Furthermore, a .GOTO label cannot bypass a %FIN label. Therefore, the best place to put a %FIN is as the last line in the control file.

If an error occurs in the job and the user either was not running a system program or has not included an .IF command or error recovery control lines, the Batch Controller initiates a standard quick dump of the user's core area and terminates the job (refer to the DUMP command in Chapter 2). The Batch Controller also initiates a dump if it is searching for a %ERR and reads a %FIN instead.

3.4.1 .BACKTO

Function

The .BACKTO command is used by Batch users to interrupt the sequential reading of the control file by the Batch Controller. Control is transferred in a backward direction. This command can be used with a .IF command to specify transfer of control to an error routine.

Command Format

.BACKTO label

label = label of a statement in the control file. This label is from one to six alphanumeric characters terminated with a colon and must not begin with a % character.

When the .BACKTO command is encountered, the Batch Controller searches for the labeled statement and transfers control to it. If the statement is not found, the job is terminated.

3.4.2 .CHKPNT

Function

The .CHKPNT command is used to aid in error recovery when a Batch job is terminated abnormally by a system failure. As many .CHKPNT commands as desired can be placed in the control file. When the job is restarted after the failure, the program begins at the location of the last .CHKPNT command instead of at the beginning of the program.

Command Format

.CHKPNT label

label = label of a statement in the control file. This label is from one to five alphanumeric characters. When the label appears with the statement in the control file, it must be followed by a double colon instead of the usual single colon (e.g., label :: statement).

3.4.3 .ERROR

Function

The .ERROR command causes the Batch Controller to recognize a message beginning with the specified character as an error in the job.

Command Format

.ERROR character

character = the beginning character of the line that is to be recognized as an error (e.g., %). If this argument is not specified, a ? at the beginning of a line is considered as an error.

3.4.4 .GOTO

Function

The .GOTO command is used by Batch users to interrupt the sequential reading of the control file by the Batch Controller. Control is transferred in a forward direction. This command may be used with a .IF command to specify transfer of control to an error routine.

<u>Command Format</u>

.GOTO label

label = label of a statement in the control file. The label appearing in the control file is from one to six alpahnumeric characters terminated with a colon and must not begin with a % character.

When the .GOTO command is encountered, the Batch Controller searches for the labeled statement and transfers control to it. If the statement is not found before the end of the control file is reached, the job is terminated.

<u>Examples</u>

```
.EX TEST.MAC/L
.IF (ERROR) .GOTO A
.GOTO B
A::.QUEUE LPT::=TEST.MAC

.GOTO B
B::;
```

3.4.5

.IF

<u>Function</u>

The .IF command is used by Batch users to aid the Batch Controller in processing errors. The Batch Controller recognizes the existence of an error when it encounters a line beginning with a question mark that is output from the job to the log file or a line that begins with the character specified in the .ERROR command. When the error occurs, this command must be the next monitor level command in the control file.

<u>Command Format</u>

.IF (condition) statement

(condition) = ERROR or NOERROR. The parentheses must be included.

statement = an executable monitor or batch command preceded by a period.

If the specified condition is true, the statement is executed. If the specified condition is not true, the Batch Controller processes the next line in the control file.

3.4.6

.NOERROR

Function

The .NOERROR command instructs the Batch Controller to ignore all errors (including messages beginning with a question mark) in the job. This is especially useful in TECO searches. However, the message

```
?TIME LIMIT EXCEEDED
```

always indicates that an error exists.

Command Format

```
.NOERROR
```

3.4.7

.NOOPERATOR

Function

The .NOOPERATOR command designates that no messages from the job are to be output to the controlling terminal.

Command Format

```
.NOOPERATOR
```

3.4.8

.OPERATOR

Function

The .OPERATOR command makes it possible for the job, or a program within the job, to communicate with the operator. Any message from the job, starting with the specified character (refer to Chapter 4), is typed on the controlling terminal. The job then waits for operator intervention and the operator's answer restarts the job.

(continued on next page)

Function (cont)

When the .OPERATOR command is in effect, the Batch Controller ignores an .IF statement unless the .NOOPERATOR command is given first, and proceeds to search for an error recovery routine labeled with either %ERR: or %CERR: (refer to Paragraph 3.4). This action is taken in order to minimize output to the operator in case of an unexpected transfer of control. However, when an error occurs, the Batch Controller preserves the error status across the .NOOPERATOR command and looks for the .IF statement as the next monitor-level command. In other words, an .IF statement following a .NOOPERATOR command will be executed. Refer to the following examples.

In the example below, the .IF statement will be ignored.

```
.OPERATOR %
.RUN TESPRG
.IF (ERROR) .GOTO TAG
```

However, in the following example, the .IF statement will be executed.

```
.OPERATOR %
.RUN TESPRG
.NOOPERATOR
.IF (ERROR) .GOTO TAG
```

Command Format

.OPERATOR character

character = the beginning character of the line that is to be sent to the operator (e.g., %). If this argument is not specified, $ at the beginning of the line is assumed.

3.4.9 .REQUEUE

Function

The .REQUEUE command indicates to the Batch Controller that the job is to be requeued, instead of terminated, after an error. It is normally used with the .IF (ERROR) command (e.g., .IF (ERROR) .REQUEUE). The job is restarted after a default requeue time at the specified label in the control file.

Command Format

.REQUEUE label

.REVIVE

Function

The .REVIVE command causes all output from the job to be placed in the log file.

Command Format

.REVIVE

.SILENCE

Function

The .SILENCE command suppresses all output from the job except error messages to the log file. This means that the only lines appearing in the log file will be those that begin with a question mark.

Command Format

.SILENCE

3.5 JOB OUTPUT

The output from a user's job is normally in the form of printed listings containing the user's job output, compilation listings, any memory dumps requested by the user or initiated by the Batch Controller, and the log file indicating the processing performed by the programs in the Batch system. The results from the job and the log file are automatically placed in the queue for the line printer spooler, LPTSPL, unless the job was submitted with the /OUTPUT:0 switch. However, the user can output to any device in the system. When a user program specifies a slow-speed spooling device, the Batch system places the output into a queue for the appropriate spooler. If the user wishes specific files to be output to particular spooled devices outside of his programs, he can include the QUEUE monitor commands in his control file to specify the output device and any additional parameters that he wishes.

Compilation listings are produced from the $language control cards unless the user specifies otherwise. These listings are automatically spooled to the line printer. The user can also include the COMPILE monitor command in his job with switches to produce listings.

The user can include any of the monitor DUMP commands or the CDRSTK card $DUMP to request memory dumps during program testing. Under normal error conditions, the Batch Controller performs an automatic two-page dump for the user (refer to Paragraph 3.4).

3.5.1 The Log File

As part of its processing, CDRSTK creates a log file for each job so that the user can examine the processing performed by the CDRSTK and BATCON programs. The log file is the first part of the job's output. CDRSTK enters a record of its own processing, any errors detected, and any operator interventions. When the job is run, the Batch Controller places additional messages into the log file, including each line of the control file as it is passed to the job, any error conditions, and any operator actions. The LOGOUT program appends an accounting summary message to the log file when the job terminates. This message is similar to the message received when an interactive user logs off the system (refer to the KJOB command in Chapter 2). Note that the log file is appended to for jobs of the same name; thus it may be necessary to delete this file before running another job with the same name.

3.5.1.1 CDRSTK Messages - CDRSTK places six kinds of messages into the log file. The first line of each message is identified by the time that CDRSTK placed the message into the file and by an identifying word in columns 1 through 16. The identifier for each kind of message is taken from the following group:

DATE -- gives the date, system name, CDRSTK version, and the input device.
STACK -- identifies any CDRSTK nonerror message.
STERR -- identifies any CDRSTK error message.
CARD -- describes any card image not in an error message.
STSUM -- identifies the summary message at the end of the CDRSTK's processing.
STOPR -- describes any operator actions that occurred during the CDRSTK's processing.

The first entry in the log file always contains the identifier DATE and a message giving the date, the system name, the current version of CDRSTK, and the input device; for example,

```
10:20:06 DATE 13-MAY-71 5S03C System 40 CDRSTK version 7 device CDR1
```

The $SEQUENCE and the $JOB commands are the next two lines printed. The $PASSWORD command is never printed for reasons of security. When the end-of-file is read, CDRSTK prints a summary message giving the number of cards read, the number of files and blocks written, and the number of each type of error that occurred. The summary is also placed in the system accounting file. An example of the job summary is given below.

```
11:25:38 STSUM End-of-File after 423 cards, 3 files (40 blocks) written
         4 Hollerith errors (nonfatal)
         2 Binary Sequence errors (fatal)
         Job Aborted by CDRSTK
```

Between the beginning and ending messages, CDRSTK prints any operator actions as they occur, some nonerror messages, and reports of errors it has detected. The following are examples of nonerror messages from CDRSTK.

```
CARD     $JOB TESTA, [10,225]
STOPR     JOB STOPPED BY OPERATOR
STOPR     CONTINUED BY OPERATOR
```

3.5.1.2 CDRSTK Error Reporting - CDRSTK places messages in the log file that describe errors that have occurred during its processing. The following errors are detected, and their degree of severity is as specified:

Fatal Errors

a. Error on the $JOB card.
b. Error on the $PASSWORD card.
c. Unrecognizable command on a CDRSTK control card.
d. Error in a parameter on a CDRSTK control card.
e. Binary sequence error - issued a maximum of five times per deck.
f. Improper code (binary rather than Hollerith, or vice versa).

Nonfatal Errors

a. Hollerith error (invalid punch).
b. Missing end-of-file card.

Error messages are issued by CDRSTK to the log file either up to the first fatal error, or, for nonfatal errors, up to a maximum of 200 errors or errors on 10% of the total card count, whichever is greater. However, CDRSTK continues processing the job up to the end-of-file. The following are examples of error messages placed in the log file by CDRSTK.

```
JOB ABORTED BY OPERATOR
JOB ABORTED - HOLLERITH ERRORS
CARD #nnn    FATAL CARD
CARD #nnn    COL #nnn
CARD #nnn    CARD SEQUENCE ERROR
CARD #nnn    SWITCH ERROR
CARD #nnn    MODE ERROR
NON-BINARY CARD IN BINARY DECK
```

Each card-reading error results in a message which includes the first card column in error, the deck number and columns 1 through 30 of the $DECK card, and the card number within the deck and within the job. The faulty card image appears on the next line with a backward slash (\) indicating the column in error.

```
11:15:05 STERR Hollerith error at col. 7 of card 241, card 73 in deck 2 ($FORTRAN MAIN)
        3 \ORMAT ('FOO')
```

3.5.1.3 Batch Controller Messages - The Batch Controller messages are similar to those of the Stacker. The times followed by an identifying notation are placed in columns 1 through 16 of the first line of each message. The identifiers for the Batch Controller messages are described in the list below:

BVERS -- denotes the version of BATCON.

BDATE -- identifies the date BATCON processed the job.

BATCH -- identifies any Batch Controller nonerror message.

BAOPR -- describes any operator action.

BAERR -- denotes any Batch Controller error message.

MONTR -- identifies a line input or output at monitor level.

USER -- describes any line input or output at user level.

BASUM -- gives the Batch Controller summary message.

The first line in the log file printed by the Batch Controller is the version number. As each line in the control file is read, it is printed in the log file as well as being passed to the user program or to the monitor. Any time that the operator performs some action that affects the job, the Batch Controller records it in the log file. The BATCON program enters a message in the log file every time it generates a monitor command. For example, if a fatal error occurs in the job and the user has not included an .IF statement, a %ERR routine or a %CERR routine in the control file, the Batch Controller generates a DUMP command. It also generates a LOGIN and a KJOB monitor command for each job.

Any errors in the input that are detected by the Batch Controller are printed in the log file.

3.5.1.4 Batch Controller Error Reporting - The Batch Controller places the identifier BAERR on any line that it detects as being an error. The errors that are detected are listed below; the first three are fatal errors.

a. Missing condition (ERROR or NOERROR) or missing statement in an .IF statement.

b. Missing statement label in the .GOTO or BACKTO command.

c. The labeled statement in a .GOTO command cannot be found after the .GOTO or before the .BACKTO command in the control file.

d. Use of the ATTACH, DETACH, SEND, CCONT, and CSTART monitor commands.

Most user error conditions are not flagged by the Batch Controller, they are passed to the monitor where they are flagged as errors.

3.6 SAMPLE JOBS

The following sample job setups illustrate the versatility of the Batch System.

The first example, Figure 3-2, shows a setup to list a card deck with the QUEUE monitor command.

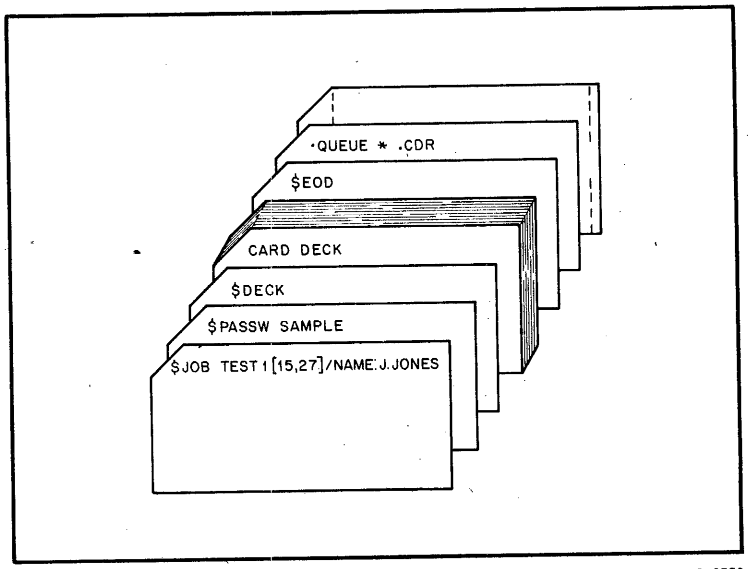
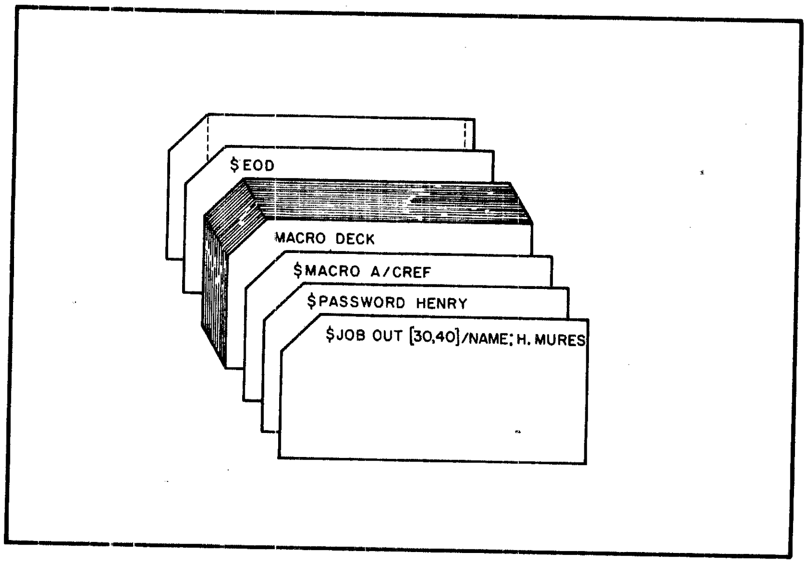


Figure 3-2 Sample Job #1

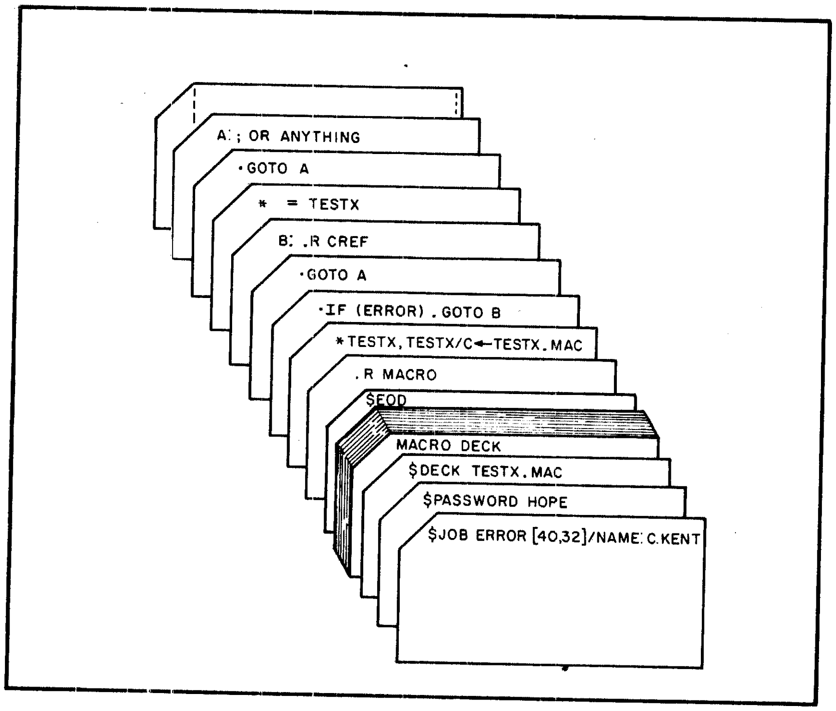
The second example, Figure 3-3, produces a CREF listing of a MACRO deck whether or not errors occur in the program.

The third example, Figure 3-4, illustrates the use of error processing commands.



10-0728

Figure 3-3 Sample Job #2



10-0727

Figure 3-4 Sample Job #3

Figure 3-5 illustrates a MACRO assembly, two FORTRAN compilations, and execution of all three programs, and shows how monitor commands are entered along with the programs and the Stacker control cards.

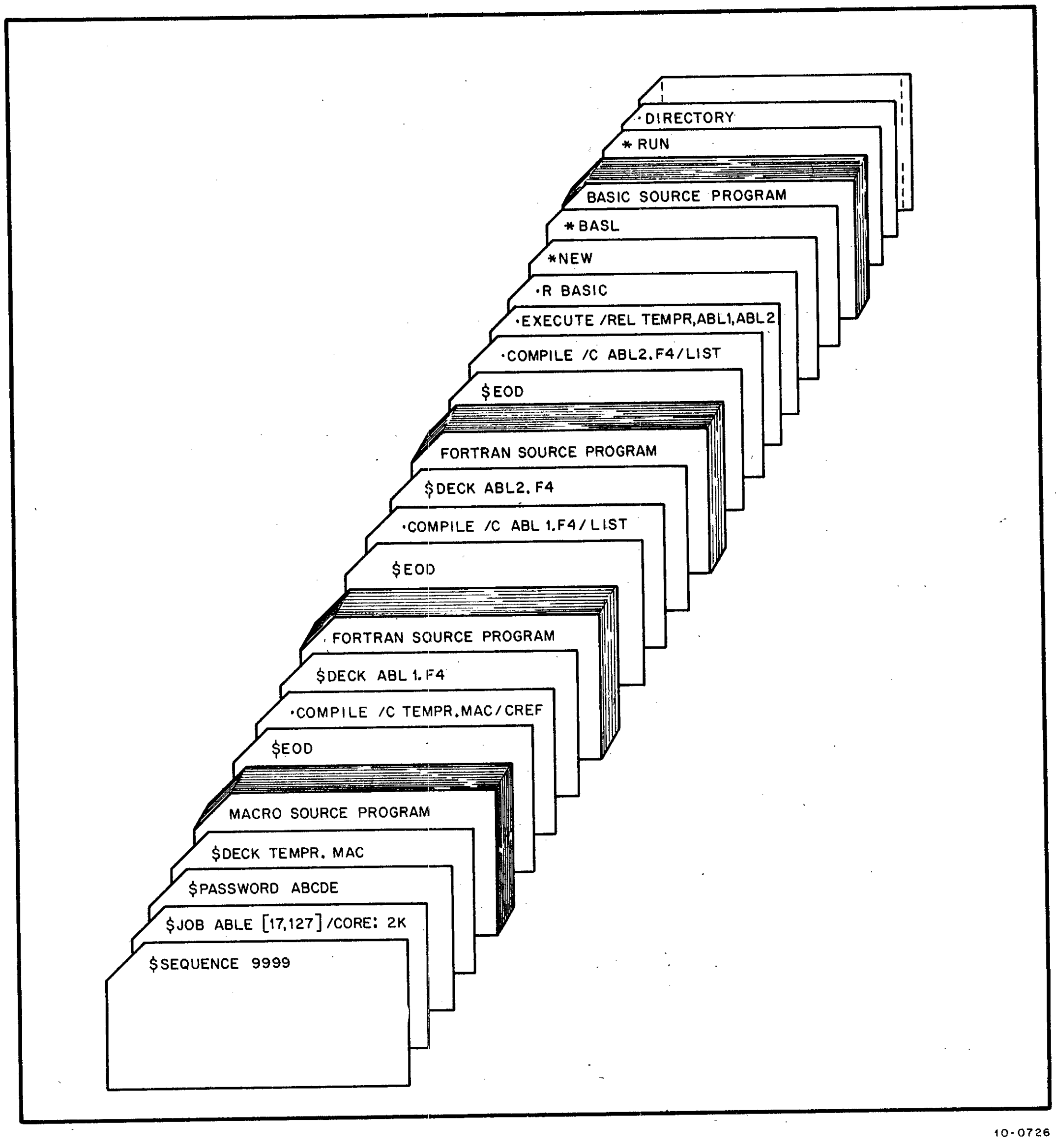
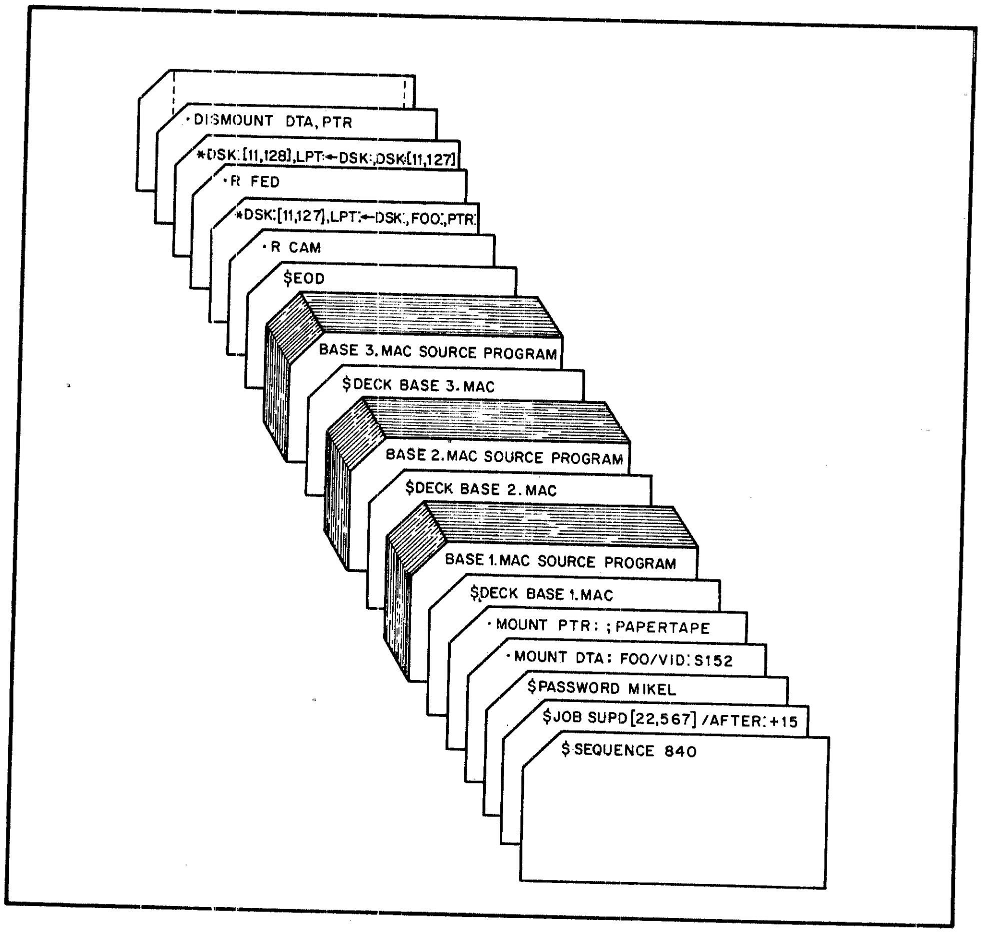


Figure 3-5 Sample Job #4

Figure 3-6 shows a simple SOUP update. Three base files are copied from cards to disk. The user files are on DECtape and the correction from DEC is on paper tape.



```
.DISMOUNT DTA,PTR
*DSK:[11,128],LPT:←DSK:,DSK:[11,127]
.R FED
*DSK:[11,127],LPT:←DSK:,FOO:,PTR:
.R CAM
$EOD
BASE 3.MAC SOURCE PROGRAM
$DECK BASE 3.MAC
BASE 2.MAC SOURCE PROGRAM
$DECK BASE 2.MAC
BASE 1.MAC SOURCE PROGRAM
$DECK BASE 1.MAC
.MOUNT PTR: ;PAPERTAPE
.MOUNT DTA: FOO/VID:S152
$PASSWORD MIKEL
$JOB SUPD[22,567]/AFTER:+15
$SEQUENCE 840
```

10-0725

Figure 3-6 Sample Job #5

CHAPTER 4
SYSTEM DIAGNOSTIC MESSAGES AND ERROR CODES

The following conventions are used in describing the system diagnostic messages:

dev	represents a legal device name.
file structure name	represents a legal file structure name.
file.ext	represents a legal filename and extension.
adr	represents a user address.
n	represents a number.
abc	represents a disk unit or drive.
x	represents an alphabetic character
switch	represents a switch.

Most messages returned to the user fall in one of five categories. These categories are determined by the beginning character of the message.

? at the start of the message indicates a fatal error message.

% at the start of the message represents an advisory or warning message.

[at the beginning of the message indicates a comment line.

$ at the beginning of the message represents an operator/job communication line. A response is expected.

' (quote) at the beginning of the message represents a comment to the operator. No response is expected.

Programs and/or commands causing the error message are given in parentheses. (Note that the ONCE-only messages have been removed and placed in ONCE.RNO in the DECsystem-10 Software Notebooks.) The descriptive text given with the message indicates what action the user should take when he receives the message. He can, if necessary, notify the operator of any problems that he is having by issuing the SEND, PLEASE, or R GRIPE command.

4.1 SYSTEM DIAGNOSTIC MESSAGES

The typein is typed back preceded and followed by ?

The monitor encountered an incorrect character (e.g., a letter in a numeric argument). The incorrect character appears immediately before the second ?.

For example:

```
.CORE ABC
?CORE A?
```

ACCOUNTING SYSTEM FAILURE...

A program could not append an entry to the accounting file. Notify the operator. (LOGIN, LOGOUT).

?ADDRESS CHECK FOR DEVICE dev

(1) The monitor checked a user address on a UUO and found it to be too large (>C(.JBREL)) or too small (≤C(.JBPFI)); in other words, the address lies outside the bounds of the user program (2) The SAVed file is too large for the core assigned, or the file is not a core image file. (GET).

$ALL AREAS ON BACKUP

The BACKUP program has processed all of the project-programmer numbers specified and is now closing the associated files. (BACKUP).

?ALREADY ASSIGNED TO JOB n

The device is already assigned to another user's job (job n).

?AMBIGUOUS ABBREVIATION

A command or switch has been abbreviated to the point that it is not unique. (COMPIL).

?ARGS ARE: DAY, RUN, WAIT, READ, WRITE, VERSION, ALL, NONE

The user either did not type an argument or typed an illegal argument in the SET WATCH command string.

dev: ASSIGNED

The device has been successfully assigned to the user's job.

?ASSIGNED TO JOB n_1, n_2, ...

If there is more than one device of the type specified, the numbers of the other jobs that have the same type of device are output, unless the user assigning the device has all the devices of the specified type. In this case, ?DEVICE ASSIGNED TO JOB is output.

?ATTACH TO USER JOB FAILED

DAEMON could not attach to the user's job. (DAEMON).

$BACKUP COMPLETED AT time

The BACKUP program has successfully completed. (BACKUP).

?BAD DENSITY

The value given with the DENSITY command was not valid. (RESTORE).

?BAD DIRECTORY FOR DEVICE DTAn

The system cannot read or write the DECtape directory without getting some kind of error. This error often occurs when the user tries to write on a write-locked tape or use a DECtape that has never been written on.

?BATCH ONLY

The command issued can only be given by a batch job.

BLOCK NOT FREE

M specifies a unit or file structure logical block that is not free. (ALCFIL).

n BLOCKS ALREADY ALLOCATED

The file already exists. The new specification replaces, rather than updates, the old specification. (ALCFIL).

?n1K BLOCKS OF CORE NEEDED

The user's current core allocation is less than the contents of .JBFF.

?BOMB OUT

The location within INITIA that detected the error will be in AC 15 and the console lights. (INITIA).

?BOOTSTRAP LOADER IS NOT IN COPY; TRY /L

An attempt was made to write the bootstrap loader onto a DECtape via the /T switch before the loader was loaded into a core buffer and preserved with the COPY core image. (COPY program).

?BOOTSTRAP LOADER WILL NOT FIT IN 3 BLOCKS

The user's bootstrap loader is too big to fit into blocks 0, 1, and 2 of the output DECtape. (COPY program).

?BUFFER CAPACITY EXCEEDED AND NO CORE AVAILABLE

The buffer is not large enough to handle the number of lines required for looking ahead for matches, and additional core is not available. (FILCOM).

?BUSY

The terminal addressed is not communicating with the monitor (i.e., it is accepting a command or returning output from a command). The operator's terminal is never busy. (SEND, JCONT).

?CANNOT DO I/O AS REQUESTED

Input (or output) cannot be performed on one of the devices specified for input (output). For example, input may have been requested for a device that can only do output. (FUDGE2).

?CANNOT DO OUTPUT TO DEVICE dev

Output was attempted to a device that can only do input, or to a device assigned a logical name. (QUEUE).

?CANNOT PROCESS EXTERNAL SYMBOLS

External symbols were encountered while loading the bootstrap loader with the /L switch. (COPY program).

?CANNOT PROCESS HIGH SEG'S

While loading the bootstrap loader with the /L switch, high segment code was encountered. (COPY program).

?CANNOT REATTACH FROM A BATCH SUBJOB

Batch jobs are not allowed to reattach their jobs. (REATTA).

$%CANT ACCESS COMMAND FILE - CONTINUING

The command recovery file is not being created. This file contains information as to how much of the user's command has been processed and how much is remaining. Without this file, the user must start at the beginning if the system crashes. (BACKUP, RESTORE).

?CANT ACCESS DEVICE dev

The device specified cannot be INITed. The device is either in use or has an error, such as, being off-line. The user should request another device, or check this device for errors. (BACKUP, RESTORE).

$%CANT ACCESS INDEX DEVICE - CONTINUING dev

The device specified for the index file cannot be INITed and an index file is not being created. The user can start over if he wants to create an index file. (BACKUP, RESTORE).

?CANT ACCESS SYSTEM FILES

ACCT.SYS could not be read. Only the operator may LOGIN until ACCT.SYS is ready. Consult the operator. (LOGIN).

?CANT ADD TO YOUR FILE STRUCTURE SEARCH LIST n

n is the error code from STRUUO when trying to add a file structure to search list. (LOGIN).

?CANT ATT TO JOB

The project-programmer number specified is not that of the owner of the desired job, the project-programmer number was not given when it was required, or the PASSWORD given was incorrect. (ATTACH).

?dev CANT BE REASSIGNED

(1) The job's controlling terminal cannot be reassigned, or (2) the logical name would be duplicated, or (3) the logical name is a physical device name in the system and the job reassigning the device is either logged-in under a different project-programmer number or does not have operator privileges. (REASSIGN).

?CANT CONTINUE

The job was terminated due to (1) all ERROR IN JOB messages (except for HALT), (2) the EXIT UUO, (3) the CLOSE command, or (4) the REA command when the device was INITed, and the user attempted to continue his program at the point at which I/O was terminated. The job cannot be continued.

CANT CREATE NEW FILE STRUCTURE SEARCH LIST

The monitor cannot create a new file structure search list.

?CANT DECIPHER COMMAND

The command typed is not recognized by the BACKUP program. (BACKUP, RESTORE).

?CANT DECIPHER THAT

There is a syntax error in the command string. (MOUNT, DISMOUNT, FILE).

?CANT DET DEV

The user is not logged-in under [1,2].

?CANT ENTER OUTPUT FILE n file descriptor

The ENTER to write the output file failed; n is the disk error code. (DUMP).

?CANT EXPAND TABLE xxxx

The DUMP program ran out of core in attempting to expand the indicated table. (DUMP).

?CANT FIND INPUT FILE n file descriptor

DUMP cannot locate the file specified as the input file; n is the disk error code. (DUMP).

?CANT FIND FILE file.ext

The specified file could not be found.

?CANT GET SWAPPING PARAMETERS

DAEMON tried to obtain the job's swapping parameters and failed. (DAEMON).

?CANT GET SWAPPING POINTER FOR JOB

DAEMON tried to obtain the pointer to the user's job on the swapping space and could not because the GETTAB UUO failed. (DAEMON).

?CANT GET USERS PPN

DAEMON tried to obtain the user's project-programmer number and could not because a GETTAB UUO failed. (DAEMON).

?CANT OPEN file structure name

The file structure is mounted but cannot be opened. No UFD is created, though one may already exist. (LOGIN).

?CANT OPEN CHANNEL FOR DEVICE dev

The OPEN on the channel for the named device failed. (BACKUP, RESTORE).

?CANT OPEN DEVICE dev

The specified device does not exist or it is assigned to another user . (DAEMON).

?CANT OPEN INDEX FILE

The OPEN failed for the index file. (BACKUP, RESTORE).

?CANT OPEN SWAP UNIT abc

DAEMON attempted to use the indicated swapping unit and failed. (DAEMON).

?CANT RELEASE UFD INTERLOCK FOR dev [p,p]

The UFD interlock cannot be released for the named device. (BACKUP).

?CANT RENAME-FILE PRESERVED

An attempt was made via the /DISPOSE:RENAME switch to delete a preserved file (i.e., a file whose owner's field is greater than 0). (QUEUE).

?CANT SET OUR SEARCH LIST

DAEMON tried to set its search list and failed in its attempt. (DAEMON).

?CANT SET SEARCH LIST = USER'S

DAEMON attempted to set its file structure search list to be the same as the user's search list. (DAEMON).

?COMMAND ERROR

General catch-all error response for most commands. The syntax of the command is in error, and the command cannot be deciphered.

In FILCOM, one of the following errors occurred in the last command string typed.

1. There is no separator (← or =) between the output and input specifications.
2. The input specification is completely null.
3. The two input files are not separated by a comma.
4. A file descriptor consists of characters other than alphanumeric characters.

(continued on next page)

5. FILCOM does not recognize the specified switch.
6. The project-programmer number is not in standard format, i.e., [proj,prog].
7. The value of the specified switch is not octal.
8. The first input file is followed by a comma but the second input file is null.

?COMMAND SYNTAX ERROR
TYPE /H FOR HELP

An illegal command string was entered. (GLOB).

?COMMA REQUIRED IN DIRECTORY

A project-programmer number has been specified without the separating comma. (DUMP, QUEUE, BACKUP, RESTORE).

CONT BY OPR

The job has been continued by the operator. This message appears on the console of the job being continued. (JCONT).

?CONTROL AND LOG FILES MUST BE DISTINCT

The control file cannot be the same file as the log file. (QUEUE).

?2K CORE NEEDED AND NOT AVAILABLE

FILCOM needs 2K of core to initialize I/O devices and this core is not available from the monitor. (FILCOM).

%CPUn OPR1 ACTION REQUESTED

The Job's CPU specification includes a CPU which is not running or is not scheduling jobs. The monitor remembers the specification and uses the CPU as soon as it is started. If at least one CPU is running, the message is printed only once, since the job can run on another CPU.

?DAEMON FILE MUST BE WRITTEN ON A DISK

The device specified was a nondisk device. (DAEMON).

?DAEMON NOT RUNNING

The DAEMON program has not been initialized. It must be started by the operator to allow the DUMP and DCORE commands to operate. (DUMP, DCORE).

?DETACH UUO FAILED

DAEMON could not detach itself from the TTY. Note that DAEMON does not detach itself if it is loaded with DDT. (DAEMON).

?DATA ERROR ON DEVICE PTR

A read error has occurred on the paper-tape reader. (COPY program).

?DESTINATION DEVICE ERROR

An I/O error occurred on the output device. (GLOB).

?DEVICE CANT BE REASSIGNED

(1) The job's controlling terminal cannot be reassigned, (2) the logical name would be duplicated, or (3) the logical name is a physical device name and the job reassigning the device is either logged in under a different project-programmer number or is not the operator.

?DEVICE ERROR ON OUTPUT DEVICE

A write error has occurred on the output file. (FUDGE2).

?DEVICE INIT FAILURE

The specified device has been assigned to another job or does not exist. (COPY program).

?DEVICE MTAn NEEDS A WRITE RING, INSERT ONE AND TYPE <CR>

This message is returned by the BACKUP and RESTORE programs.

?DEVICE MUST BE A DECTAPE

The only device that can be specified in the COPY command string is the DECtape. (COPY program).

?DEVICE dev NOT A DIRECTORY DEVICE

This message is returned by the BACKUP program.

?DEVICE NOT ASSIGNABLE

A non-privileged user cannot assign the requested device because it belongs to the restricted pool of devices. The user should try to assign the device with the MOUNT command. (ASSIGN).

?DEVICE NOT AVAILABLE

Specified device cannot be initialized because another user is using it or because it does not exist.

?DEVICE WILDCARD ILLEGAL

The wildcard construction cannot be used in the device specification. (DUMP, QUEUE, BACKUP, RESTORE).

?DIALOG MODE NOT SUPPORTED

The capability of interactive dialogue with the user has not been implemented. (QUEUE).

?DIRECTORY FULL ON OUTPUT DEVICE

There is no room in the file directory on the output device to add the updated file (nondisk devices only). (FUDGE2).

device name DISMOUNTED

The DISMOUNT command has completed.

?device name DISMOUNT INCOMPLETE

The DISMOUNT command was unsuccessful. In most cases, the reasons for failure have already been listed by nonerror messages.

DONT KNOW CTY LINE NUMBER

The DCORE command cannot be typed on CTY. (DAEMON).

?DOUBLE DEVICE ILLEGAL

Two device names appeared in a row without an intervening filename, or two colons appeared in a row, e.g., LPT:PTP: or DSKA ::FILEX. (DUMP, QUEUE).

?DOUBLE DIRECTORY ILLEGAL

Two directory names cannot appear without an intervening filename. (DUMP, QUEUE).

?DOUBLE EXTENSION ILLEGAL

Two extensions cannot appear without an intervening filename or comma. (DUMP, QUEUE).

?DOUBLE FILENAME ILLEGAL

Two filenames appeared in a row, or two periods appeared in a row; e.g., Q TEST1 TEST2 or TEXTX..MAC. (DUMP, QUEUE).

DPAn NO DRIVE AVAILABLE ON THIS CONTROLLER

The drives on the specified controller are all in use. (MOUNT).

?DSK CANT BE REASSIGNED

An attempt was made to reassign the prototype disk device data block (DDB).

?DSKCHR FAILURE n ON UNIT abc

The DSKCHR UUO gave an unexpected error return; n is the disk error code. Notify the operator. (DAEMON, KJOB).

%END OF TAPE id ON dev
PLEASE MOUNT TAPE id+1 AND TYPE <CR> TO CONTINUE:

This message is sent to the operator. (RESTORE).

%END OF TAPE id ON dev
PLEASE MOUNT NEXT TAPE AND TYPE <CR> TO CONTINUE

This message is sent to the operator. (BACKUP).

%END OF TAPE n ON MTAn AT time

The end of the tape has been reached. (RESTORE).

\$END OF TAPE n ON $\left\{\begin{matrix}\text{DPA}\\\text{MTA}\\\text{DSK}\end{matrix}\right\}$ x AT time

This message appears in the log file. (BACKUP).

?ENTER ERROR n
?DIRECTORY FULL

No additional files can be added to the directory of the output device; n is the disk error code. (GLOB).

?ENTER FAILURE

The DECtape directory is full (i.e., there is no room for the file to be written on the DECtape).

?ENTER FAILURE n

The output filename is null; n is the error code for an illegal filename (nondisk devices only). (FUDGE2).

?ENTER FAILURE FOR INDEX FILE

The ENTER failed for the index file. (BACKUP).

?ENTER FAILURE IN QUEUE MANAGER

QUEUE was unable to enter the files into the output queue. (QUEUE).

?ENTER FAILURE n ON $\left\{\begin{matrix}\text{CCL}\\\text{DAEMON}\end{matrix}\right\}$ FILE

The ENTER to write the file failed; n is the disk error code.

?ENTRY BLOCK TOO LARGE PROGRAM name

The entry block of the named program is too large for the FUDGE2 entry table, which allows for 100 entry names. FUDGE2 can be reassembled with a larger table. (FUDGE2).

?ERROR CLOSING OUTPUT, STATUS = n

An I/O error occurred while closing the file on disk; n is the disk error code. (DUMP).

?ERROR IN JOB n

A fatal error occurred in the job or in the monitor while servicing the job. This typeout usually precedes a one-line description of the error.

?EXCEED LOG-OUT m QUOTA BY n BLOCKS

The total number of blocks for all the user's files exceeds the maximum permitted value (m) by the indicated amount n. The user may use PIP or the DELETE command to remove files. Until the user is under the limit, he cannot dismount the file structure. (DISMOUNT).

?EXECUTION DELETED

A program is prevented from being executed because of errors detected during assembly, compilation, or loading. Loading is performed, but the loader exits to the monitor without starting execution. (LOADER).

?EXPECTED FORMAT IS "NNNK" = 16K to 256K

The core-bank specified while processing the /T switch is not within the acceptable range or does not terminate with the letter K; e.g., 32 is not acceptable; 32K is. (COPY program).

%FAILURE ON $\left\{\begin{matrix}\text{ENTER}\\ \text{OPEN}\end{matrix}\right\}$ FOR ERROR FILE--CONTINUING

The error file could not be generated. The BACKUP program is continuing without one. (BACKUP).

?FAILURE ON $\left\{\begin{matrix}\text{INIT}\\ \text{OPEN}\end{matrix}\right\}$ FOR LOG FILE

The log file could not be generated. (BACKUP, RESTORE).

%FAILURE OUTPUTTING ERROR FILE--CONTINUING

The error file could not be output. The BACKUP program is continuing its processing. (BACKUP, RESTORE).

%FAILURE $\left\{\begin{matrix}\text{READING}\\ \text{CREATING}\end{matrix}\right\}$ UFD FOR dev [proj,prog]

The UFD for the named device could not be read (BACKUP) or created (RESTORE).

%FAILURE TO INTERLOCK UFD FOR dev [proj,prog]

The UFD interlock for the named device failed. (BACKUP).

file structure name FILE ERRORS EXIST

One of the files in a file structure has an error status, as flagged in the UFD of that file structure. (LOGIN).

?FILENAME ALREADY IN USE

The specified file already exists. (COMPIL).

?FILENAME REQUIRED FOR INPUT QUEUE

A file cannot be entered into the Batch input queue without a filename. (QUEUE).

?FILE n NOT IN SAV FORMAT

The user indicated via the /X switch that the file is to be expanded but the specified file is not in compressed file format. N is either 1 or 2 indicating the first file or the second file. (FILCOM).

?FILE n READ ERROR

An error has occurred on either the first or second input device. (FILCOM).

?FILE SWITCHES ILLEGAL IN OUTPUT FILE

File switches cannot appear on the left of the equal sign, i.e., in the output specification. (QUEUE).

?(3) FILE WAS BEING MODIFIED-file.ext

Another user is modifying the file. (COMPIL).

?(0) FILE WAS NOT FOUND-file.ext

The named file could not be located. (COMPIL).

?FORMAT OR READ ERROR IN AUXACC.SYS

LOGIN unexpectedly found an end-of-file or an error in AUXACC.SYS. Notify the operator. (LOGIN).

file.ext FOUND BAD BY FAILSAFE READING MTA

The file in the file structure has an error status as flagged in the UFD of the file structure. (LOGIN).

FROM JOB n

An informative message telling the user the job number to which the console was attached or from which the console is detaching. (ATTACH, DETACH).

?FUDGE2 SYNTAX ERROR

An illegal command string was entered; for example, the left arrow was omitted or a program name was specified for the output file. (FUDGE2).

?GIVING BACK TOO MUCH CORE

An internal problem in the DUMP program. Notify your system programmer or software specialist. (DUMP).

?HALT AT USER adr

The user's program executed a HALT instruction at adr. Typing CONTINUE resumes execution at the effective address of the HALT instruction.

file.ext HARDWARE DATA READ ERROR DETECTED

The file has a hardware data read error flagged in the UFD of the file structure. (LOGIN).

file.ext HARDWARE DATA WRITE ERROR DETECTED

The file has a hardware data write error flagged in the UFD of the file structure. (LOGIN).

?HUNG DEVICE dev

If a device does not respond within a certain period after it is referenced, the system decides that the device is not functioning and outputs this message.

?ILLEGAL BACKUP DEVICE

The BACKUP operations can be done only on disk, magnetic tape, and DECtape. (BACKUP).

?ILLEGAL BLOCK TYPE

While loading the bootstrap loader with the /L switch, an unrecognizable block type was encountered by COPY. (COPY program).

?ILLEGAL COMMAND SYNTAX CHARACTER x

The character x is used incorrectly in the command string. (QUEUE, BACKUP).

?ILLEGAL DATA MODE FOR DEVICE dev AT USER adr

The data mode specified for a device in the user's program is illegal, such as dump mode for the terminal.

?drive ILLEGAL DRIVE NAME

The drive specified by the user is in conflict with the unit or controller type required by the units of the file structure. (MOUNT).

?ILLEGAL IN BATCH JOB

The ATTACH, DETACH, SEND, CCONT, and CSTART monitor commands cannot be used by a batch job.

?ILLEGAL JOB NUMBER

The job number is too large or is not defined in this configuration.

?ILLEGAL QUEUE DEVICE

The queue name specified cannot be used with the given switch. (QUEUE).

?ILLEGAL QUEUE NAME xxx

The queue is not one of the system queues, or the queue is a logical name. (QUEUE).

?ILLEGAL TO CREATE REQUEST FOR SOMEONE ELSE

Only the operator logged in under 1,2 can create queueing request for other users. (QUEUE).

?ILLEGAL UUO AT USER adr

An illegal UUO was executed at user location adr.

?ILL INST. AT USER adr

An illegal operation code was encountered in the user's program.

?ILL MEM REF AT USER adr

An illegal memory reference was made by the user's program. If this message occurred on a memory write, the error is at adr-1 since the program counter has been advanced. If it occurred on a memory read, then the illegal instruction is probably in location adr. The user should use the E command to first examine location adr-1 and then location adr in order to determine the illegal instruction. The index registers may also have to be examined.

?INDEX FILE CANNOT GO TO A LISTING DEVICE

This message is returned by the BACKUP and RESTORE programs.

?INPUT AND OUTPUT DECTAPES MAY NOT BE THE SAME DEVICE

The COPY program performs its operations on an input DECtape and an output DECtape. These DECtapes cannot be the same. (COPY program).

?INPUT (or OUTPUT) BLOCK TOO LARGE

A DECtape block number greater than 1101_8 was encountered. (COPY program).

?INPUT (or OUTPUT) CHECKSUM OR PARITY ERROR

A read (or write) error has been detected. (COPY program).

?INPUT DEVICE dev CANNOT DO OUTPUT AT USER adr

Output was attempted on a device that can only do input (e.g., the card reader).

?INPUT (or OUTPUT) DEVICE ERROR

The DECtape control unit has detected the loss of data or a missed block. (COPY program).

?INPUT DEVICE NOT A DISK

The input specifications in a QUEUE command must be disk files. (QUEUE).

?INPUT ERROR

An I/O error occurred while reading a temporary command file from the disk. File should be rewritten. (COMPIL).

?INPUT ERROR - file.ext FILE NOT FOUND

The specified file could not be found on the input device. (FILCOM).

%INPUT ERROR DSKn $\left\{\begin{array}{l}\text{file.UFD}\\ \text{file.MFD}\\ \text{file.SFD}\end{array}\right\}$ [proj,prog]

The BACKUP program cannot access the entries in the named directory. These entries will not be saved on the BACKUP medium. The BACKUP program continues by advancing to the next directory. (BACKUP).

?INPUT ERROR, STATUS = n

An I/O error occurred while reading the file from disk; n is the disk error code. A new INPUT command causes a new LOOKUP to be done. (DUMP, DAEMON).

?INPUT FAILED FOR FILE DSKn file.ext [proj,prog]

The INPUT failed for the specified file. (BACKUP, RESTORE).

?INPUT (or OUTPUT) PREMATURE END OF FILE

When copying a DECtape, COPY encountered the end of file before it expected it. This may happen when copying a PDP-9 DECtape to a PDP-10 DECtape. (COPY program).

?INSUFFICIENT CORE FOR QUEUE

There is not enough core in system at the time of the KJOB command to make an output queue entry. (QUEUE).

?INVALID ARGUMENT

The argument specified on a BACKSPACE or PARITY command is unknown. (BACKUP, RESTORE).

?INVALID ENTRY - TRY AGAIN
#

An illegal project-programmer number or password was entered and did not match identification in system. The user is to retype his project-programmer number and password. (LOGIN).

?I/O TO UNASSIGNED CHANNEL AT USER adr

An attempt was made to do an OUTPUT, INPUT, OUT, or IN to a device that the user's program has not initialized.

?x IS AN ILLEGAL $\left\{\begin{array}{l}\text{CHARACTER}\\ \text{SWITCH}\end{array}\right\}$

An illegal character or switch was encountered in the command string. (FUDGE2).

?symbol IS A MULTIPLY DEFINED LOCAL

The named symbol is in more than one symbol table with different values. (DUMP).

?symbol IS AN UNDEFINED SYMBOL

The named symbol is not in DUMP's symbol table. (DUMP).

?symbol IS AN UNDEFINED SYMBOL TABLE NAME

The named symbol table has not been loaded with an XTRACT command. (DUMP).

?JOB CAPACITY EXCEEDED

This message is received by a user who attempts to login after the maximum number of jobs that the system has been set to handle has been initiated. The user should login in at a later time. (LOGIN).

?JOB NOT WAITING

The job specified is not waiting to be continued. (JCONT).

JOB SAVED

The output is completed.

JOBn USER [p,p] LOGGED OFF TTY n AT hhmm dd-mm-yy
DELETED <ALL> n FILES
SAVED <ALL> n FILES m TOTAL BLOCKS USED
ANOTHER JOB STILL LOGGED IN UNDER [p,p]
RUNTIME n MIN m SEC

This information is typed as user logs off successfully. Note that m is total blocks allocated as opposed to blocks written. Therefore, it is always greater than or equal to the number of blocks written. Files are allocated in units of blocks called clusters. The system administrator selects the cluster size for each file structure, usually one block per cluster for FH file structures, and 5 or 10 blocks per cluster for DP file structures. (KJOB).

?LANGUAGE PROCESSOR CONFLICT

The use of the + construction has resulted in a mixture of source languages. (COMPIL).

?LEVEL D ONLY

The command issued is available only in 5-series monitors.

?LINKAGE ERROR - RUN UUO

An I/O error occurred while reading a program from the device SYS:. (COMPIL).

%LISTING DEVICE OUTPUT ERROR, STATUS =

The device specified for the output has an error. A new OUT command selecting a new file can be given or an OUT and APPEND command sequence to try again. (DUMP).

?LISTING ENTER FAILURE n

The ENTER to write the output file failed; n is the disk error code. (QUEUE).

?LISTING OPEN FAILURE ON DEVICE dev

The OPEN failed on device dev. (QUEUE).

?LOCKED-OUT BY OPERATOR

The operator is preventing any new accesses to the file structure in order that it may be removed. (MOUNT).

file structure name LOGGED OUT QUOTA n EXCEEDED BY m BLOCKS

The user's allocation on the file structure named is greater than his logged out quota. The user must go through the CONFIRM dialogue and delete files until he is under the quota allowed to log off. (KJOB, LOGOUT).

%LOGICAL NAME WAS IN USE, DEVICE dev ASSIGNED

The user previously assigned this logical name to another device. The logical name is cleared from the first device and assigned to the second.

?LOGIN PLEASE

A command that requires the user to be logged in has been typed to the monitor; it cannot be accepted until the user performs a LOGIN.

?LOGIN PLEASE TO USE SWITCH CREATE

The user must be logged in to make a new entry into a system queue. (QUEUE).

?LOOKUP ERROR n
?file.ext FILE NOT FOUND

The named file cannot be found in the directory on the specified device. (GLOB)

%LOOKUP ERROR DSKn file [proj,prog]

The BACKUP or RESTORE program cannot access the indicated file and continues by skipping to the next file. (BACKUP, RESTORE).

?LOOKUP FAILED, "BSLDR.REL"

While processing the /L switch, COPY could not find the bootstrap loader named BSLDR.REL. (COPY program).

?LOOKUP FAILURE

The LOOKUP to read the disk file failed. This message is followed by a line explaining the reason for failure. (FUDGE2).

?file structure name LOOKUP FAILURE n

The LOOKUP to read the file failed; n is the disk error code.

?LOOKUP FAILURE FOR INPUT FILE n file

DUMP cannot read the input file. (DUMP).

?LOOKUP FAILURE n ON DAEMON FILE

The LOOKUP to read the DAEMON file failed; n is the disk error code. (DAEMON).

?MAX = n

A value was specified for an argument that is greater than the maximum value (n) allowed. (DUMP).

?MAY NOT LOGIN AS MFD PPN

No one can login as [1,1] because this number is the project-programmer number of the MFD. (LOGIN).

?MAY NOT LOGIN $\left\{\begin{array}{l}\text{LOCAL}\\ \text{REMOTE}\\ \text{DATA SET}\\ \text{BATCH JOB SUBJOB}\\ \text{REMOTE CTY OR OPR}\end{array}\right\}$

ACCT.SYS entry does not permit the project-programmer number to login at the terminal that is being used. (LOGIN).

?MAY NOT LOGOUT WITH FILE STRUCTURES FOR LOGICAL NAMES

A file structure in the job's search list is assigned a logical name, and only physical device names are recognized. The user should deassign the logical names. (KJOB, LOGOUT).

?MEM PAR ERROR AT USER PC adr

The processor detected a memory parity error in the low or high segment while the job was executing. The adr is the address of the PC stored by the hardware rather than the user address of the parity error. The operator also receives an error message giving the range of absolute addresses in case memory reconfiguration is necessary. DAEMON is awakened in order to record the pertinent information about the error for field service personnel.

The user must start a new copy of his program by typing the appropriate monitor command R, RUN, or GET. He should not start the program over by typing START, since the error is likely to reoccur or the program operate with incorrect data.

?MFD $\left\{\begin{array}{l}\text{LOOKUP}\\ \text{READ}\end{array}\right\}$ FAILURE

The MFD cannot be accessed. (BACKUP).

?MORE THAN ONE $\left\{\begin{array}{l}\text{OUTPUT}\\ \text{INPUT}\end{array}\right\}$ DEVICE ILLEGAL

Files for the BACKUP operations can be taken from or written to only one device at a time. (BACKUP, RESTORE).

?MORE THAN ONE OUTPUT FILE ILLEGAL

Only one output queue-name may be specified in the QUEUE command string. (QUEUE).

device MOUNTED

The device is mounted and ready for use. The MOUNT command has completed. If a file structure was mounted, a list of the unit ID's and the drives on which they are mounted is output. (MOUNT).

?device MOUNT INCOMPLETE

The MOUNT command has not completed successfully. In most cases, the reasons for failure have already been listed by nonerror messages. In a Batch job, MOUNT INCOMPLETE not preceded by a message may indicate that the user is attempting to mount a spooled device without executing a SET SPOOL command to unspool the device. The user must have unspool privileges in his accounting file entry in order to unspool and mount spooled devices.

?MUST BE IN OWNER'S PROJECT FOR SINGLE ACCESS

The user may not request single-access (/SINGLE switch) unless he has the same project number as the owner of the file structure. This requirement is enforced since a user with single access may execute super-USETI/USETO UUOs. (MOUNT).

name MUST NOT BE A LOGICAL NAME

The structure named contains the operator request queue (3,3.UFD) and must not be the logical name for some other structure. (MOUNT).

?file structure name MUST NOT BE WRITE-PROTECTED

The named structure is being used to queue requests to the operator and therefore may not be write-protected, SETSRC may be used to change the protection. (MOUNT, DISMOUNT, FILE).

NAME:

The ACCT.SYS entry for this project-programmer number requires the user to type a name which matches the one in ACCT.SYS in order to login. (LOGIN).

?NEED 5.03 OR LATER FOR REATTACH COMMAND

The REATTA program depends on UUOs available in the 5.03 release of the monitor. The user attempted to run the program using an older monitor. (REATTA).

?NESTING TOO DEEP

The @ construction exceeds a depth of nine and may be due to a loop of @ command files. (COMPIL).

?NO CORE ASSIGNED

No core was allocated when the GET command was given and no core argument was specified in the GET.

NO DIFFERENCES ENCOUNTERED

No differences were found between the two input files. (FILCOM).

?(1) NO DIRECTORY FOR PROJECT-PROGRAMMER NUMBER - file.ext

A UFD does not exist for the requested project-programmer number. (COMPIL).

?NO END BLOCK ENCOUNTERED

The last block of the bootstrap loader program must be an end block (refer to the MACRO manual). (COPY program).

?NO ENTRY IN AUXACC.SYS
NO SEARCH LIST OR UFDS CREATED

If the user has no entry in AUXACC.SYS, LOGIN does not create UFDS or a search list. User is logged-in and has UFDs if they existed previously. He may write only on file structures that have UFDs or read all file structures. He may also create a file structure search list with SETSRC. The user can create UFDs on those file structures for which he has an entry in QUOTA.SYS by using the MOUNT command. (LOGIN).

NO ENTRY IN QUOTA.SYS

The user may utilize the file structure, but no UFD is created if he does already have one. (MOUNT).

%NO INFO ON "name"

The user specified a feature that has no available documentation. (HELP).

?NO INPUT DEVICE SPECIFIED
SPECIFY INPUT DEVICE NOW:

An input device name was not specified prior to the START command. (RESTORE).

?NO MODIFIER ALLOWED IN SWITCH switch

The switch specified cannot have an argument. (QUEUE).

NONE PENDING

None of the user's requests to the operator are pending.

?NON-EXISTENT DRIVE DPAn

The user has specified a drive that does not exist in the system. (MOUNT).

%NON-EXISTENT FILE input specification

The file specified for input could not be found. This message is not output if the /NEW switch is specified for the file. (QUEUE).

?NON-EX MEM AT USER adr

Usually due to an error in the monitor.

?NO OPR.JOB FOR THIS REQUEST

An operator request has been issued, but there is no OMOUNT running and enabled to service the request. The request is still queued unless the /PAUSE switch was given.

?NO OUTPUT DEVICE SPECIFIED
SPECIFY OUTPUT DEVICE NOW:

An output device name was not specified prior to the START command. (BACKUP).

?NO PRIVILEGES TO SET CPU

The user does not have the privilege bits set by LOGIN from ACCT.SYS to change the CPU specification. The user should request that these privilege bits be set by the system manager.

?NO PRIVS TO UNSPOOL

The user does not have privileges to unspool devices, and the operator has not set bit 28 in the STATES word.

?NO REMOTE USERS. TRY AGAIN LATER

The operator has used the SET SCHEDULE command to prevent LOGINs from remote terminals. The message of the day is still typed. (LOGIN).

NO ROOM IN QUEUE, TRY AGAIN LATER

There is no room in the queue for the user's request to be sent to the operator. (MOUNT).

?(14) NO ROOM OR QUOTA EXCEEDED - file.ext

There is no room on the file structure or the user's quota on the file structure has been exceeded.

%NO RUNNING CPUS IN SPECIFICATION

If none of the CPUs in the job's CPU specification are running, the user receives this message every minute until the CPU is started or he types a new SET CPU command.

?NO START ADR

Starting address or reenter address is zero, because the user failed to specify the starting address in the END statement of the source program or in the START command. However, an implicit starting address of 0 may be specified.

?NO SUCH DEVICE

The device name does not exist or was not assigned to this job.

?NO SUCH JOB

An attempt was made to attach to a job that has not been initialized.

?NO SUCH STR

A nonexistent file structure was specified. (KJOB).

?NO SUCH TTY

The terminal number is not part of the system configuration.

?NO SUCH UNIT

The unit does not exist or all units of this type are in use.

?NOT A JOB

The job number is not assigned to any currently running job. (ATTACH, DSK, JCONT). There is no job logged in at this terminal. (CONTINUE).

?NOT A SAVE FILE

The file is not a core image file.

?NOT A SPOOLING DEVICE

The device specified is not one of the spooling devices (LPT, CDP, CDR, PTP, PLT).

?NOT A STR - TRY AGAIN

The file structure specified is not recognized by the monitor.

?NOT A TTY

The device name given is not a terminal. (REATTA).

?drive NOT AVAILABLE

The drive indicated by the user is not currently available. (MOUNT).

?command NOT CODED

A command that is not in this version of DUMP was specified in the command string. (DUMP).

?NOT ENOUGH ARGUMENTS

An insufficient number of files of one type has been specified. (FUDGE2).

?NOT ENOUGH CORE

The system cannot supply enough core to use as buffers or to read in a system program. (COMPIL).

NOT ENOUGH DRIVES AVAILABLE

There are currently not enough drives of the right type to mount the file structure. (MOUNT).

NOT ENOUGH TABLE SPACE FOR SWAPPING UNITS

There are more swapping units than DAEMON allowed for. DAEMON should be reassembled. (DAEMON).

?dev file.ext program NOT FOUND

The file or the program was not found on the device or in the file specified. If a program name is printed, this message may indicate that the program names in the command string appear in a sequence different from their sequence within the file. Therefore, the program may actually exist but was missed because of the incorrect sequence in the command string. (FUDGE2).

?file.SAV NOT FOUND

The program file requested cannot be found on the system device or the specified device.

drive NOT READY

The indicated drive is either off-line or physically write-locked when write-enabled was requested. The operator will be notified. (MOUNT).

?NOT YET SUPPORTED COMMAND CODE switch

A switch has been specified that is not implemented. (QUEUE).

NO UFD CREATED

The user may access the file structure, but he cannot write in his disk area since he has no UFD. (MOUNT).

?NULL DEVICE ILLEGAL

A colon has been found without a preceding device name. (QUEUE, BACKUP, RESTORE).

?NXM adr

While computing the value of an expression, a non-existent location was specified when referencing the input file. (DUMP).

?nk OF CORE NEEDED or ?nP OF CORE NEEDED

There is insufficient free core to load the files; n is the size being requested for the segment that failed (either high or low segment, not the sum of the high and low segments). This message occurs when the virtual core for the system has been exceeded or the core for this job has been exceeded. The user should type CORE↩ to determine what core has been exceeded, and whether the high or low segment was too big. K denotes 1024 words which is the unit of core allocation on a KA10-based system, and P denotes 512 words (one page) which is the unit of allocation on a KI10-based system.

?OFFSET = 1000 TO 777600 (OCTAL)

The offset specified by the user is not within the acceptable range. (COPY program).

?ONLY BATCH USERS MAY LOGIN. TRY AGAIN LATER

The operator has used the SET SCHEDULE command to prevent LOGINs, except for BATCH jobs. The message of the day is still typed. (LOGIN).

?OPEN FAILED FOR DEVICE dev

The OPEN for the named device failed. (BACKUP, RESTORE).

?OPEN FAILURE ON DATA DEVICE dev

The OPEN on the specified device failed. (DUMP).

OPERATOR BUSY, HANG ON PLEASE.

The user must wait for the operator to become available.

OPERATOR NOTIFIED

(1) The operator is available and the user may continue typing his message. (PLEASE).
(2) A request is queued to the operator to perform a specified action. (MOUNT, DISMOUNT).

OPERATOR REQUESTED TO MOUNT UNITS

A request is queued to the operator to mount and ready the packs on the proper drives. (MOUNT).

OPERATOR REQUESTED TO READY DRIVES

One or more drives (as specified by previous messages) are not ready. A request is queued to the operator. (MOUNT).

OPERATOR REQUESTED TO REMOVE PACKS

A request to physically remove the packs has been queued to the operator. (DISMOUNT).

OTHER USERS - CANNOT SINGLE ACCESS

Other users are currently using the file structure that has been specified with the single-access switch (/SINGLE). The switch is ignored. (MOUNT).

OTHER USERS - CANT REMOVE

A DISMOUNT command requesting physical removal (/REMOV switch) of a pack has been issued and there are other users of the pack. The switch is ignored. (DISMOUNT).

OTHER USERS SAME PPN

A program has determined that other jobs are currently logged-in under the same project-programmer number. (LOGIN, KJOB).

?OUT OF BOUNDS

The specified adr is not in the user's core area, or the high segment is write-protected and the user does not have privileges to the file that initialized the high segment. (D, E).

?OUTPUT DEVICE dev CANNOT DO INPUT AT USER adr

An attempt was made to input from an output device (e.g., the line printer).

?OUTPUT DEVICE ERROR

An error has occurred on the output device. (FILCOM).

?OUTPUT ERROR

An I/O error occurred while writing a temporary command file on disk. (COMPIL).

?OUTPUT ERROR, STATUS = n

An I/O error occurred while writing the file on disk; n is the disk error code. (DAEMON).

?OUTPUT INITIALIZATION ERROR

The output device cannot be initialized for one of the following reasons:

1. The device doe not exist or is assigned to another job.
2. The device is not an output device.
3. The file cannot be placed on the output device. (FILCOM).

PASSWORD:

The user must type a PASSWORD which matches that in the ACCT.SYS entry for this project-programmer number. Echoing is suppressed to preserve PASSWORD security. If the user is at a half-duplex (local copy) terminal, this message is replaced by a sequence of random over-typed characters, over which the user types his PASSWORD. (LOGIN).

PAUSE...(↑C TO QUIT, CR TO CONT)

The /PAUSE switch has been specified, and an operator action is about to be requested. ↑C aborts the command before the request is queued to the operator. Carriage return-line feed allows the command to continue, and the request is queued to the operator. (DISMOUNT).

?PC OUT OF BOUNDS AT USER adr

An illegal transfer has been made by the user program to user location adr.

?PLEASE KJOB OR DETACH

Attempt was made to LOGIN a job when the user already has a job initialized at that terminal. (LOGIN).

?PLEASE LOGIN AS [OPR]

The operator is the only person that can initialize DAEMON by typing R DAEMON.

?PLEASE TYPE ↑C FIRST

A command which would start a job has been issued after a CSTART or CCONT.

?PPN HAS EXPIRED

The current date is greater than the expiration date of the project-programmer number. The user may not login until expiration date is changed by the system manager. (LOGIN).

?PROGRAM ERROR WHILE RESETTING MASTER DEVICE

FUDGE2 cannot find the master device or cannot find the program on the master device. (FUDGE2).

?PROJECT 1 MAY NOT BE PTY

Project 1 is never allowed to login over a pseudo-TTY. (LOGIN).

?PROTECTION FAILURE DSK file.ext [proj,prog]

The user does not have access to the specified disk areas for either a read or a write. (BACKUP, RESTORE).

?(2) PROTECTION FAILURE - file.ext

There was a protection failure or the directory on DECtape had no room for the file. (COMPIL).

?PTR INIT FAILURE

The logical device PTR is not available or could not otherwise be initialized. (COPY program).

QUOTA.SYS LOOKUP FAILURE

The LOOKUP to read QUOTA.SYS failed. (MOUNT).

QUOTA.SYS NOT ON STRUCTURE

QUOTA.SYS is not part of this structure. The user may still use the file structure, but no UFD will be created. (MOUNT).

QUOTA.SYS READ ERROR

An I/O error occurred while reading QUOTA.SYS. (MOUNT).

QUOTA.SYS WRONG FORMAT VERSION

Wrong version of QUOTA.SYS is on the file structure being mounted. Consult the operator. (MOUNT).

%READ ERROR DSKn file [proj,prog]

The BACKUP or RESTORE program cannot input the designated file. (BACKUP, RESTORE).

?file structure name RENAME FAILURE n

The RENAME to change the protection of the file failed; n is the disk error code. (KJOB, LOGOUT).

?(4) RENAME FILENAME ALREADY EXISTS - file.ext

The new filename on a RENAME command already exists. (COMPIL).

REQUEST STORED
n COMMANDS IN QUEUE

The request typed by the user has been placed in a queue to be performed when possible. n is the number of requests in the queue for all users. (FILE, MOUNT, DISMOUNT).

?REQUIRES DEVICE NAME

The device name or file structure name is required with the MOUNT and DISMOUNT commands.

$RESTOR COMPLETED AT time

The RESTORE program has successfully completed. (RESTORE).

?RIGHT BRACKET REQUIRED IN DIRECTORY

The project-programmer number must be enclosed in square brackets. (QUEUE).

%SEARCH LIST DOES NOT ALLOW CREATES

There are no file structures available to the user on which he can write. Run MOUNT or SETSRC to modify the search list as necessary. (LOGIN).

%SEARCH LIST ERROR [proj,prog]

The BACKUP or RESTORE program is unable to obtain the search list for the named project-programmer number. The program advances to the next project-programmer number. (BACKUP, RESTORE).

%SEARCH LIST IS EMPTY

There are no file structures in the DSK: search list that are available to the user. He can run the SETSRC program to modify his search list. (LOGIN).

?SINGLE-ACCESS BY JOB n

The file structure is already single access by the indicated user. (MOUNT).

file.ext SOFTWARE CHECKSUM OR REDUNDANCY ERROR

The file has no error as flagged in the UFD of the file structure. (LOGIN).

?SOME OTHER TIME

The user is not scheduled to LOGIN at this time. He should try again when he is allowed to login. (LOGIN).

?SORRY, CANT OPEN DSK, PLEASE CALL THE OPERATOR

This message is returned from the GRIPE program.

?SORRY, CANT WRITE IN COMPLAINT AREA, PLEASE CALL THE OPERATOR

This message is returned from the GRIPE program.

?SORRY, COMPLAINT BASKET IS FULL, PLEASE CALL THE OPERATOR

This message is returned from the GRIPE program.

?SORRY, NO UFD FOR COMPLAINT BASKET, PLEASE CALL THE OPERATOR.

This message is returned from the GRIPE program.

START OF $\left\{\begin{matrix}\text{BACKUP}\\ \text{RESTORE}\end{matrix}\right\}$ VERSION n AT time YEAR nn DAY dd

The BACKUP or RESTORE program is beginning its processing. (BACKUP, RESTORE).

?STATION NOT IN CONTACT

The requested station is not in contact with the central station. (LOCATE).

?STATION NUMBER INVALID

The requested station number is not recognized by the system. (LOCATE).

STRUCTURE ALREADY MOUNTED

The requested file structure already exists and does not need to be physically mounted. (MOUNT).

?STRUCTURE NOT IN STRLST.SYS

The file structure name does not exist in the system administrator's file SYS:STRLST.SYS and, therefore, is not defined for the system. The operator or administrator may be requested to define the file structure by adding it to STRLST.SYS with the REACT program. (MOUNT).

?STRUUO FAILURE

The STRUUO UUO gave an error return. Notify the operator. (KJOB, LOGOUT).

%SUPERSEDING EXISTING FILE

A warning message indicating that a file already exists with the specified name. This file is being superseded. (TECO).

%SWAP READ ERROR UNIT abc STATUS = n

An I/O error occurred while reading the swapping space. The data is written into the DAEMON file as read. (DCORE).

?SWITCH ERROR

An illegal switch specification was given. (COPY program).

?switch SWITCH ILLEGAL

The switch specified cannot be used with the given queue name. (QUEUE, BACKUP, RESTORE).

?SWITCH VALUE TOO LARGE x

The value given to the switch exceeds the maximum value. (QUEUE).

?SYNTAX ERROR

There is a syntax error in the command string. Check for incorrect parentheses or two operators in a row.

?SYSSTR FAILURE

The SYSSTR UUO gave an error return. Notify the operator. (KJOB, LOGOUT).

?SYSTEM ERROR - xxxxxx

System errors designate operator or system errors and are not a direct fault of the user. They are typed for possible diagnostic used.

?SYSTEM NOT AVAILABLE

The operator has used the SET SCHED command to prevent LOGINs from timesharing terminals. The message of the day is still typed. (LOGIN).

?TABLE OVERFLOW - CORE UUO FAILED TRYING TO EXPAND TO xxx

The GLOB program requested additional core from the monitor, but none was available. (GLOB).

?THIS MONITOR WAS BUILT FOR A xxx AND WILL NOT RUN PROPERLY ON A yyy

The monitor is not running on the machine for which it was built. xxx and yyy are PDP-6, KA10, or KI10.

?TIME LIMIT EXCEEDED

The time limit allocated for the job has been reached. The job is stopped and the terminal is returned to monitor mode.

TIMESHARING WILL CEASE IN m HOURS n MINUTES

The KSYS command (OPSER) or SET KSYS UUO has been issued in order to stop timesharing on the system at the indicated time.

?TOO FEW ARGUMENTS

A command has been typed, but necessary arguments are missing.

?TOO MANY FILENAMES OR PROGRAM NAMES

More than 40 program names or filenames were specified in the command string. The user should separate the job into several segments. (FUDGE2).

?TOO MANY FILE STRUCTURES

The number of file structures exceeds the capacity of the monitor data base. The current limit is 14_{10}. (ONCE ONLY).

?TOO MANY NAMES or ?TOO MANY SWITCHES

Command string complexity exceeds table space in the COMPIL program. (COMPIL).

?TRANSMISSION ERROR

During a SAVE, GET, or RUN command, the system received parity errors from the device, or was unable to read the user's file in some other way. This can be as simple as trying to write on a write-locked tape.

?TRANSMISSION ERROR ON INPUT DEVICE dev

A transmission error has occurred while reading data from the specified device. (FUDGE2).

?TRIED TO OVERWRITE DATA WORD

After writing the core image file, DAEMON backs up to overwrite a word not known previously (e.g., the length of the category). In overwriting the word, DAEMON encountered a deviation from the standard pattern used in originally writing the word. (DAEMON).

?TRY LARGER ARG

The specified argument is too small for the program. This message is followed by the standard output. (CORE).

?TTYn ALREADY ATTACHED

Job number is erroneous and is attached to another console, or another user is attached to the job.

?TTY IN USE

The terminal requested is already controlling a job or is otherwise in use. (REATTA).

TYPE CORE BANK OR OFFSET FOR DTBOOT

On a /T switch, COPY asks for a core bank or offset for the bootstrap loader. The core bank is 16K to 256K and the offset is 1000 to 777600 octal. (COPY program).

TYPE H FOR HELP

An unintelligible response or command has been typed. Either the filename or the CONFIRM: message is repeated, depending upon what was typed. (KJOB).

?UFD ENTER FAILURE n

Failure in trying to create UFD; n is the disk error code. Notify the operator. (LOGIN).

%UFD ERROR DSKn [proj,prog]

The BACKUP or RESTORE program cannot access the UFD (LOOKUP failure). It advances to the next UFD. (BACKUP, RESTORE).

?file structure name UFD INTERLOCK BUSY

Could not get UFD interlock when trying to set up a UFD. The UFD is not currently set up. Notify the operator. (LOGIN).

?UFD LOOKUP FAILURE n

A failure occurred in setting up a UFD; n is the disk error code. Notify the operator. (LOGIN).

?UFD OUTPUT FAILURE n

The output failed when trying to create the UFD (4-series); n is the software channel status. (LOGIN).

?file structure name UFD READ ERROR, STATUS = n

A read error occurred while reading the user's UFD on the file structure. Status n tells which error occurred. Notify the operator. (KJOB, LOGOUT).

?UFD RENAME FAILURE n

A failure occurred in setting up a UFD; n is the disk error code. Notify the operator. (LOGIN).

?UNDEFINED SWITCH switch

The specified switch is either undefined or not unique. (MOUNT, DISMOUNT).

?UNEQUAL NUMBER OF MASTER AND TRANSACTION PROGRAMS

On a replace request, the number of master programs (or files) does not equal the number of transaction programs (or files). (FUDGE2).

UNIT abc ALREADY MOUNTED ON DRIVE DPAn

The file structure is already mounted but is on different drives than the user specified. (MOUNT).

?UNKNOWN COMMAND

The monitor passed a command to COMPIL which COMPIL does not recognize. (COMPIL).

?UNKNOWN DEFAULT FOR SWITCH switch

The default condition is not known for the specified switch. (DUMP, QUEUE).

?UNKNOWN OR INVALID COMMAND - TYPE GO TO CONTINUE

This message is typed by the BACKUP and RESTORE programs.

?UNKNOWN SWITCH switch

The switch named has been mistyped. (DUMP, QUEUE).

?UNKNOWN SWITCH VALUE n

The argument specified with the switch has been mistyped. (DUMP, QUEUE).

?UNRECOGNIZABLE SWITCH

An ambiguous or undefined word followed a slash. (COMPIL).

?UUO AT USER adr

This message accompanies many error messages and indicates the location of the UUO that was the last instruction the user program executed before the error occurred.

n VERIFICATION ERRORS

On a word by word comparison requested via the /V switch, n discrepancies have been detected between the input DECtape and output DECtape. (COPY program).

WAITING...

A request has been queued to the operator and the command is waiting for the operator to complete the request. If the user does not want to wait for completion of the operator's action, he can type control-C without aborting the command. The operator action will still be completed. Later a DISMOUNT/CHECK or MOUNT/CHECK can be given to check for completion. (MOUNT, DISMOUNT).

WAIT PLS

The system's primary accounting file FACT.SYS was busy. It is retried for ten seconds before FACT.X01 is tried. This message can appear if many users are logging in simultaneously. (LOGIN, KJOB, LOGOUT).

%WARNING - INPUT REQUEST USES ONLY TWO ENTRIES

Only two files can be specified in the input queue request, the control file and the log file. (QUEUE).

!WARNING NO INDEX ON OUTPUT FILE-CONTINUING

The user has changed the structure of the index library file when deleting, appending, or inserting, thereby invalidating the index. The index has been removed from the new file. Re-indexing is required. (FUDGE2).

?dev WASNT ASSIGNED

The device is not currently assigned to the user's job and cannot be deassigned or reassigned by the job.

?WASNT DET

The specified device is not detached.

?WILDCARD ILLEGAL IN INPUT QUEUE FILE $\left\{\begin{matrix}\text{NAME}\\\text{DIRECTORY}\\\text{EXTENSION}\end{matrix}\right\}$

The wildcard construction cannot be used when specifying the Batch input queue. (QUEUE).

?WILDCARD ILLEGAL IN OUTPUT $\left\{\begin{matrix}\text{NAME}\\\text{DIRECTORY}\\\text{EXTENSION}\end{matrix}\right\}$

The wildcard construction cannot be used in the output queue specification. (QUEUE).

?WRITE LOCK ERROR

An attempt was made to write on a write-locked DECtape. (COPY program).

?WRONG FORMAT FOR SYMBOL

A symbol was given in the format program :symbol and a symbol name did not follow the colon; in other words, the colon must be followed by a symbol. (DUMP).

?WRONG FORMAT VERSION NUMBER IN SYSTEM FILES

Wrong version of ACCT.SYS or AUXACC.SYS is on the system. Consult the operator so that he can run REACT to change the accounting files. (LOGIN).

YOU ARE LOGGED IN AS n,m

When a user logs in with a unique programmer number (project, #), this message informs him of the project-programmer number that LOGIN assigned. (LOGIN).

?YOU DONT HAVE PRIVILEGES TO WRITE $\left\{\begin{matrix}\text{DAEMON}\\\text{CCL}\end{matrix}\right\}$ FILE

The user attempted to write in a file to which he did not have access. (DAEMON).

?1+1nK CORE
VIR. CORE LEFT = 0

The swapping space or the core allocated to timesharing is all in use (i.e., there is no available virtual core). The user should wait a few minutes, and then attempt to login again. If this message still appears, it should be reported to the operator.

m+n/p CORE
VIR. CORE LEFT = v

Key: m = number of blocks in low segment.
n = number of blocks in high segment.
p = maximum core per job. (Maximum physical user core unless limited by operator, or there are jobs locked in core (refer to DECsystem-10 Monitor Calls)).
v = number of K blocks unassigned in core and on the swapping device.

Note that nK represents 1024-word blocks which is the unit of core allocation on a KA10-based system, and nP respresents 512-word blocks which is the unit of allocation on a KI10-based system.

4.2 ERROR CODES

The following error codes are returned in AC on RUN and GETSEG UUOs, in location E + 1 on 4-word argument blocks of LOOKUP, ENTER, and RENAME UUOs, and in the right half of location E + 3 on extended LOOKUP, ENTER, and RENAME UUOs. The codes are defined in the S.MAC monitor file.

Table 4-1
Error Codes

Symbol	Code	Explanation
ERFNF%	0	File not found, illegal filename (0,*), or filenames do not match (UPDATE).
ERIPP%	1	UFD does not exist on specified file structures. (Incorrect project-programmer number.)
ERPRT%	2	Protection failure or directory full on DTA.
ERFBM%	3	File being modified (ENTER).
ERAEF%	4	Already existing filename (RENAME) or different filename (ENTER after LOOKUP).
ERISU%	5	Illegal sequence of UUOs (RENAME with neither LOOKUP nor ENTER, LOOKUP after ENTER).
ERTRN%	6	a. Transmission, device, or data error (RUN, GETSEG only). b. Hardware-detected device or data error detected while reading the UFD RIB or UFD data block. c. Software-detected data inconsistency error detected while reading the UFD or file RIB.
ERNSF%	7	Not a saved file (RUN, GETSEG only).
ERNEC%	10	Not enough core (RUN, GETSEG only).
ERDNA%	11	Device not available (RUN, GETSEG only).
ERNSD%	12	No such device (RUN, GETSEG only).
ERILU%	13	Illegal UUO (GETSEG only). No two-register relocation capability.
ERNRM%	14	No room on this file structure or quota exceeded (overdrawn quota not considered).
ERWLK%	15	Write-lock error. Cannot write on file structure.
ERNET%	16	Not enough table space in free core of monitor.
ERPOA%	17	Partial allocation only.
ERBNF%	20	Block not free on allocated position.

(continued on next page)

Table 4-1 (Cont)
Error Codes

Symbol	Code	Explanation
ERNSD%	21	Cannot supersede an existing directory (ENTER).
ERDNE%	22	Cannot delete a non-empty directory (RENAME).
ERSNF%	23	Sub-directory not found (some SFD in the specified path was not found).
ERSLE%	24	Search list empty (LOOKUP or ENTER was performed on generic device DSK and the search list is empty).
ERLVL%	25	Cannot create a SFD nested deeper than the maximum allowed level of nesting.
ERNCE%	26	No file structure in the job's search list has both the no-create bit and the write-lock bit equal to zero and has the UFD or SFD specified by the default or explicit path (ENTER on generic device DSK only).
ERSNS%	27	A GETSEG from a locked low segment is not for a high segment that is a dormant, active, or idle segment.

APPENDIX A
STANDARD FILENAME EXTENSIONS

Table A-1
Filename Extensions

Filename Extension	Meaning	Type of File
AID	Source file in AID language	Source
ALG	Source file in ALGOL language	Source
ALP	Printer forms alignment	ASCII
ATO	OPSER automatic command file	ASCII
B10	Source file in BLISS-10	Source
B11	Source file in BLISS-11	Source
BAC	.Output from the BASIC Compiler	Object
BAK	Backup file from TECO or LINED	Source
BAS	Source file in BASIC language	Source
BCM	Listing file created by FILCOM (binary compare)	ASCII
BIN	Binary file for PDP-8 (DC68A)	Object
BKP	Index file created by the BACKUP program	ASCII
BLB	Blurb file	ASCII
BLI	Source file in BLISS language	Source
BUG	Saved to show a program error	Object
CAL	CAL data and program files	Object
CBL	Source file in COBOL language	Source
CCL	Alternate convention for command file (@ command file construction for programs other than COMPIL)	ASCII
CCO	Listing of modifications to non resident software	ASCII
CDP	Spooled output for card punch	ASCII, Binary
CKP	Checkpoint core image file created by COBOL operating system	Binary

(continued on next page)

Table A-1 (Cont)
Filename Extensions

Filename Extension	Meaning	Type of File
CHN	CHAIN file	Object
CMD	Command file for indirect commands (@ construction for COMPIL)	ASCII
CMP	Complaint file by GRIPE	ASCII
COR	Correction file for SOUP	ASCII
CRF	CREF (cross-reference) input file	ASCII
CTL	MP batch control file	ASCII
DAE	Default output for DAEMON-taken core dumps	Binary
DAT	Data (FORTRAN) file	ASCII, Binary
DDT	Input file to FILDDT	ASCII
DIR	Directory from FILE command or DIRECT program	ASCII
DMP	COBOL compiler dump file	ASCII
DOC	Listing of modifications to the most recent version of the software	ASCII
DSE	Directory sorted by extension	ASCII
DSF	Directory sorted by filename	ASCII
ERR	Error message file	ASCII
F4	Source file in FORTRAN language	Source
FAL	Source file in FAIL language	Source
FLO	English language flowchart	ASCII
FRM	Blank form for handwritten records	ASCII
FUD	FUDGE2 listing output	ASCII
HGH	Nonsharable high segment of a two-segment program (created by SAVE command)	Object
HLP	Help files containing switch explanations, etc.	ASCII
IDA	COBOL ISAM data file	ASCII, Binary
IDX	Index file of a COBOL ISAM file	ASCII, SIXBIT
INI	Initialization file	ASCII, Binary
LAP	Output from the LISP compiler	ASCII
LIB	COBOL source library	ASCII
LOG	MP batch log file	ASCII

(continued on next page)

Table A-1 (Cont)
Filename Extensions

Filename Extension	Meaning	Type of File
LOW	Low segment of a two-segment program (created by SAVE command)	Object
LPT	Spooled output for line printer	ASCII
LSD	Default output for DUMP program	ASCII
LSP	Source file in LISP language	Source
LSQ	Queue listing created by QUEUE program	ASCII
LST	Listing data created by assemblers and compilers	ASCII
MAC	Source file in MACRO language	Source
MAN	Manual (documentation) file	ASCII
MAP	Loader map file	ASCII
MEM	Memorandum file	ASCII
MIM	Snapshot of MIMIC simulator	Binary
MSB	Music compiler binary output	Object
MUS	Music compiler input	Source
OLD	Backup source program	Source
OPR	Installation and assembly instructions	ASCII
OVR	COBOL overlay file	Object
PAL	Source file in PAL 10 (PDP-8 assembler)	Source
P11	Source program in MACX11 language	Source
PL1	Source file in PL1 language	Source
PLT	Spooled output for plotter	ASCII
PTP	Spooled output for paper-tape punch	ASCII, Binary
Qxx	BAK files (all xx)	ASCII
QUD	Queued data file	ASCII, Binary
QUE	Queue request file	Binary
QUF	Master queue and request file	Binary
REL	Relocatable binary file	Object
RIM	RIM loader file	Object
RMT	Read-in mode (RIM) format file (PIP)	Object
RNC	RUNOFF input for producing a .CCO file	ASCII
RND	RUNOFF input for producing a .DOC file	ASCII

Table A-1 (Cont)
Filename Extensions

Filename Extension	Meaning	Type of File
RNO	Programming specifications in RUNOFF input	ASCII
RNP	RUNOFF input for producing a .OPR file	ASCII
RSP	Script response time log file	ASCII
RST	Index file created by the RESTORE program	ASCII
RTB	Read-in mode (RIM10B) format file (PIP)	Object
SAV	Low segment from a one-segment program (created by SAVE command)	Object
SCM	Listing file created by FILCOM (source compare)	ASCII
SCP	SCRIPT control file	ASCII
SEQ	Sequential COBOL data file, input to ISAM program	ASCII, SIXBIT
SFD	Subfile directory (restricted usage)	Binary
SHR	Sharable high segment file of a two-segment program (created by SAVE command)	Object
SNO	Source file in SNOBOL language	Source
SNP	Snapshot of disk by DSKLST	ASCII
SRC	Source files	ASCII
SVE	.SAVed file from a single user monitor	Object
SYS	Special system files	Binary
TEC	TECO macro	ASCII
TEM	Temporary files	ASCII, Binary
TMP	Temporary files	ASCII, Binary
TXT	Text file	ASCII
UFD	User file directory (restricted usage)	Binary
UPD	Updates flagged in margin (FILCOM)	ASCII
WCH	SCRIPT monitor (WATCH) file	ASCII
XPN	Expanded save file (FILEX)	Object

APPENDIX B
CARD CODES

Table B-1
ASCII Card Codes

ASCII Character	Octal Code	Card Punches	ASCII Character	Octal Code	Card Punches
NULL	00	12-0-9-8-1	@	100	8-4
CTRL-A	01	12-9-1	A	101	12-1
CTRL-B	02	12-9-2	B	102	12-2
CTRL-C	03	12-9-3	C	103	12-3
CTRL-D	04	9-7	D	104	12-4
CTRL-E	05	0-9-8-5	E	105	12-5
CTRL-F	06	0-9-8-6	F	106	12-6
CTRL-G	07	0-9-8-7	G	107	12-7
CTRL-H	10	11-9-6	H	110	12-8
TAB	11	12-9-5	I	111	12-9
LF	12	0-9-5	J	112	11-1
VT	13	12-9-8-3	K	113	11-2
FF	14	12-9-8-4	L	114	11-3
CR	15	12-9-8-5	M	115	11-4
CTRL-N	16	12-9-8-6	N	116	11-5
CTRL-O	17	12-9-8-7	O	117	11-6
CTRL-P	20	12-11-9-8-1	P	120	11-7
CTRL-Q	21	11-9-1	Q	121	11-8
CTRL-R	22	11-9-2	R	122	11-9
CTRL-S	23	11-9-3	S	123	0-2
CTRL-T	24	9-8-4	T	124	0-3
CTRL-U	25	9-8-5	U	125	0-4
CTRL-V	26	9-2	V	126	0-5
CTRL-W	27	0-9-6	W	127	0-6
CTRL-X	30	11-9-8	X	130	0-7
CTRL-Y	31	11-9-8-1	Y	131	0-8
CTRL-Z	32	9-8-7	Z	132	0-9
ESCAPE	33	0-9-7	[	133	12-8-2
CTRL-\	34	11-9-8-4	\	134	0-8-2
CTRL-]	35	11-9-8-5	]	135	11-8-2
CTRL-↑	36	11-9-8-6	↑ Λ	136	11-8-7
CTRL-←	37	11-9-8-7	← _	137	0-8-5
SPACE	40		`	140	8-1

NOTE: The ASCII character ESCAPE (octal 33) is also CTRL-[on a terminal.

Table B-1 (Cont)
ASCII Card Codes

ASCII Character	Octal Code	Card Punches	ASCII Character	Octal Code	Card Punches
!	41	12-8-7	a	141	12-0-1
"	42	8-7	b	142	12-0-2
#	43	8-3	c	143	12-0-3
$	44	11-8-3	d	144	12-0-4
%	45	0-8-4	e	145	12-0-5
&	46	12	f	146	12-0-6
'	47	8-5	g	147	12-0-7
(	50	12-8-5	h	150	12-0-8
)	51	11-8-5	i	151	12-0-9
*	52	11-8-4	j	152	12-11-1
+	53	12-8-6	k	153	12-11-2
,	54	0-8-3	l	154	12-11-3
-	55	11	m	155	12-11-4
.	56	12-8-3	n	156	12-11-5
/	57	0-1	o	157	12-11-6
0	60	0	p	160	12-11-7
1	61	1	q	161	12-11-8
2	62	2	r	162	12-11-9
3	63	3	s	163	11-0-2
4	64	4	t	164	11-0-3
5	65	5	u	165	11-0-4
6	66	6	v	166	11-0-5
7	67	7	w	167	11-0-6
8	70	8	x	170	11-0-7
9	71	9	y	171	11-0-8
:	72	8-2	z	172	11-0-9
;	73	11-8-6	{	173	12-0
<	74	12-8-4	|	174	12-11
=	75	8-6	}	175	11-0
>	76	0-8-6	~	176	11-0-1
?	77	0-8-7	DEL	177	12-9-7

NOTE: The ASCII characters } and ~ (octal 175 and 176) are treated by the monitor as ALTmode which is often considered to be the same as ESCAPE.

Table B-2
DEC-029 Card Codes

Character	Octal Code	Card Punches	Character	Octal Code	Card Punches
SPACE	40		@	100	8-4
!	41	11-8-2	A	101	12-1
"	42	8-7	B	102	12-2
#	43	8-3	C	103	12-3
$	44	11-8-3	D	104	12-4
%	45	0-8-4	E	105	12-5
&	46	12	F	106	12-6
'	47	8-5	G	107	12-7
(	50	12-8-5	H	110	12-8
)	51	11-8-5	I	111	12-9
*	52	11-8-4	J	112	11-1
+	53	12-8-6	K	113	11-2
,	54	0-8-3	L	114	11-3
-	55	11	M	115	11-4
.	56	12-8-3	N	116	11-5
/	57	0-1	O	117	11-6
0	60	0	P	120	11-7
1	61	1	Q	121	11-8
2	62	2	R	122	11-9
3	63	3	S	123	0-2
4	64	4	T	124	0-3
5	65	5	U	125	0-4
6	66	6	V	126	0-5
7	67	7	W	127	0-6
8	70	8	X	130	0-7
9	71	9	Y	131	0-8
:	72	8-2	Z	132	0-9
;	73	11-8-6	[	133	12-8-2
<	74	12-8-4	\	134	11-8-7
=	75	8-6	]	135	0-8-2
>	76	0-8-6	↑ ∧	136	12-8-7
?	77	0-8-7	← _	137	0-8-5

NOTE: Octal codes 0-37 and 140-177 are the same as in ASCII.

Table B-3
DEC-026 Card Codes

Character	Octal Code	Card Punches	Character	Octal Code	Card Punches
SPACE	40		@	100	8-4
!	41	12-8-7	A	101	12-1
"	42	0-8-5	B	102	12-2
#	43	0-8-6	C	103	12-3
$	44	11-8-3	D	104	12-4
%	45	0-8-7	E	105	12-5
&	46	11-8-7	F	106	12-6
'	47	8-6	G	107	12-7
(	50	0-8-4	H	110	12-8
)	51	12-8-4	I	111	12-9
*	52	11-8-4	J	112	11-1
+	53	12	K	113	11-2
,	54	0-8-3	L	114	11-3
-	55	11	M	115	11-4
.	56	12-8-3	N	116	11-5
/	57	0-1	O	117	11-6
0	60	0	P	120	11-7
1	61	1	Q	121	11-8
2	62	2	R	122	11-9
3	63	3	S	123	0-2
4	64	4	T	124	0-3
5	65	5	U	125	0-4
6	66	6	V	126	0-5
7	67	7	W	127	0-6
8	70	8	X	130	0-7
9	71	9	Y	131	0-8
:	72	11-8-2/11-0	Z	132	0-9
;	73	0-8-2	[	133	11-8-5
<	74	12-8-6	\	134	8-7
=	75	8-3	]∧	135	12-8-5
>	76	11-8-6	↑	136	8-5
?	77	12-8-2/12-0	←	137	8-2

NOTE: Octal codes 0-37 and 140-177 are the same as in ASCII.

APPENDIX C
TEMPORARY FILES

The temporary files in Table C-1 are used by various programs in the DECsystem-10 computing system. These files are in the following form:

nnn xxx.TMP

where nnn is the user's job number in decimal, with leading zeroes to make three digits, and xxx specifies the use of the file.

Table C-1
Temporary Files

Name	Meaning
nnn ALG.TMP	Read by ALGOL and contains one line for each program to be compiled. It may also contain the command NAME! which causes ALGOL to transfer control to the named program.
nnn AS1.TMP nnn AS2.TMP nnn AS3.TMP	Written, read, and deleted by COBOL and contains input to the COBOL assembler.
nnn BLI.TMP	Read by BLISS and contains one line for each program to be compiled.
nnn COB.TMP	Read by COBOL and contains one line for each program to be compiled. It may also contain the command NAME! which causes COBOL to transfer control to the named program.
nnn CPY.TMP	Written, read, and deleted by COBOL and contains copies of source files with library routines inserted.
nnn CRE.TMP	Read by CREF and contains commands for each file which has produced a CREF listing on the disk. COMPIL also reads this file each time a new CREF listing is generated to prevent multiple requests for the same file and to prevent discarding other requests that may not yet have been listed.
nnn DAE.TMP	Written by DAEMON to be read by DUMP.
nnn DMP.TMP	Read by DUMP as an input command file.

Table C-1 (Cont)
Temporary Files

Name	Meaning
nnn EDS.TMP	Used by COMPIL to store the arguments of the most recent EDIT, CREATE, TECO, or MAKE command.
nnn EDT.TMP	Written by COMPIL and read by LINED or TECO. It contains a command for each EDIT, CREATE, TECO, or MAKE command. For the MAKE or CREATE commands, it contains the command S file.ext [p,p] ($) For TECO or EDIT commands, it contains the command S file.ext [p,p] ⤸
nnn ERA.TMP	Written, read, and deleted by COBOL and is the error file.
nnn FOR.TMP	Read by FORTRAN and contains one line for each program to be compiled. It may also contain the command NAME! which causes FORTRAN to transfer control to the named program.
nnn GEN.TMP	Written, read, and deleted by COBOL and contains the output of syntax processing.
nnn KJO.TMP	Read by KJOB as an input command file.
nnn LGO.TMP	Read by LOGOUT as an input command file.
nnn LIN.TMP	Created by LINED and contains output file until the rename process.
nnn LIT.TMP	Written, read, and deleted by COBOL and contains copy of the literal pool.
nnn LOA.TMP	Read by LOADER and contains commands necessary for loading.
nnn MAC.TMP	Read by MACRO and contains one line for each program to be assembled. It may also contain the command NAME! which causes MACRO to transfer control to the named program.
nnn P11.TMP	Read by MACX11 (the PDP-11 assembler for the PDP-10) and contains one line for each program to be assembled.
nnn PLS.TMP	Read by PLEASE as an input command file.
nnn PIP.TMP	Read by PIP and contains commands to implement the COMPIL-class commands that run PIP.
nnn QUE.TMP	Read by QUEUE as an input command file.
nnn RNO.TMP	Read by RUNOFF and contains commands for each file which has produced a RUNOFF listing on the disk.
nnn S01.TMP	Written, read, and deleted by COBOL and contains the intermediate sorted results of the data.
nnn SVC.TMP	Used by COMPIL to store the arguments of the most recent COMPILE, LOAD, EXECUTE, or DEBUG command.
nnn SNO.TMP	Read by SNOBOL and contains one line for each program to be compiled.

(continued on next page)

Table C-1 (Cont)
Temporary Files

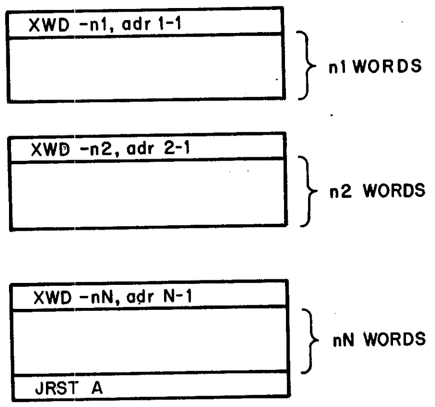
Name	Meaning
nnn TEC.TMP	Created by TECO and contains output file until the rename process.
nnn TMP.TMP	Created by LINED during the rename process.
nnn XFO.TMP	Created by FILEX as a result of the Q switch on the output side.
nnn XFR.TMP	Created by FILEX as a result of the Q switch on the input side.

APPENDIX D
SAVE AND SSAVE COMMANDS

Before writing SAVed or LOW files in response to SAVE and SSAVE commands (refer to the individual command descriptions in Chapter 2), the monitor compresses the user's core image by eliminating consecutive blocks of zeroes. This technique is known as zero-compression and is used to save space on file media. Low segment files are zero-compressed on devices DTA, MTA, and DSK, but high segment files are not because the high segment can be shared at the time of the command.

SAVed files are ordinary binary files and can be copied using the /B switch in PIP. Files with the LOW or SAV extension may be read in dump mode, but must be reexpanded before being run. The monitor expands the file after input on a RUN, R, or GET command. The FILEX program may be used to expand the file for other purposes.

The data format of a zero-compressed SAVed file consists of a series of IOWDs and data block pairs and is terminated by a JRST A where A is the program starting address as specified by the contents of .JBSA. The format is as follows:



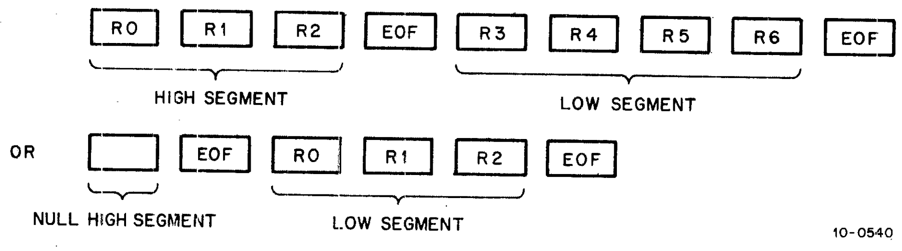
10-0544

Each IOWD describes the length of the following data block and the original location of the data in core. The LH of the IOWD can be positive in which case the number of words is taken as the number of words greater than 128K.

SAVed files are read into the user's core area starting at location .JBSAV and then are expanded to occupy the original relative locations. If the first word read is not an IOWD and is positive, an old-format, noncompressed saved file is assumed and no expansion is performed.

A SAVE command issued to a magnetic tape writes

a. a high segment (possibly null)
b. an EOF
c. a low segment (possibly null)
d. an EOF.



The monitor does not determine the file size of a low segment on a GET from magnetic tape; therefore, a user must always specify a core argument or have enough core assigned to his job for the file.

To save file space, only the high segment up through the highest nonzero location (relative to high segment origin) loaded, as specified in the LH of .JBHRL, will be written by the SAVE command. If LH is zero (high segment created by CORE or REMAP UUO) or DDT is present, the entire high segment will be written.

The LOADER indicates to the SAVE command how much data was loaded above the job data area in the low segment by setting the LH of .JBCOR to the highest location in the low segment that was not explicitly loaded with data (either zero or nonzero). Most programs are written so that only the high segment contains nonzero data. In this case, SAVE and SSAVE write only the high segments. This also saves file space and I/O time with the GET command.

A number of locations in the job data area need to be initialized on a GET, although there is no other data in the low segment. The SAVE command copies these locations into the first 10_8 locations of the high segment, provided it is not sharable. The locations are referred to as the vestigial job data area (refer to DECsystem-10 Monitor Calls, Chapter 1). Therefore, the LOADER will load high segment programs starting at location 400010.

To prevent user confusion, SAVE and SSAVE delete a previous file with the extension .SHR or .HGH; therefore, SAVE deletes a file with the extension .SHR and SSAVE deletes a file with the extension .HGH. SAVE and SSAVE commands also delete files with the extension .LOW, if the high segment was the only segment written.

The regular access rights of the saved file indicate whether a user can perform a GET, R, or RUN command. These commands assume that the user wants to execute (but not modify) the high segment, independent of the access rights of the file used to initialize the segment. The monitor always enables the hardware user-mode write protect to prevent the user program from storing into the segment inadvertently.

To debug a reentrant system program, the user should make a private, nonsharable copy, rather than modify the shared version and possibly cause harm to other users. To make a private, nonsharable copy, the following commands are used:

GET prog	
SAVE	Writes a file in the user directory as nonsharable. The high segment in the user's addressing space remains sharable.
GET	Overlays the sharable program with the nonsharable one from the user's directory. Now the user can make patches while other users share the version in the library.

If the user is debugging a sharable program in his UFD with the D command or the DDT program, it is recommended that the program be nonsharable instead of sharable. The reason for this is that the user may wish to modify the high segment during the debugging phase and later reinitialize the original unmodified high segment from the file with a GET command. However, since the high segment is sharable, the monitor will not do I/O into it, will not reinitialize it from the disk file, and the user will receive the modified high segment instead.

NOTE

DDT modifies the high segment when it inserts breakpoints.

The following examples are the incorrect and correct methods of debugging a sharable program. After the debugging phase is completed, the SSAVE command should be used to save the program.

Example 1: Incorrect Method

```
.DEBUG prog )
EXECUTION
↑C
.SSAVE )                                ;SAVE should be used in debugging
.GET )
JOB SETUP )
.E 400010
400010/777777 777777 .D Ø Ø )
.E )
400010/ Ø Ø
.GET )
JOB SETUP
.E 400010 )
400010/ Ø Ø                             ;not the original 777777 777777
```

Example 2: Correct Method

```
.DEBUG prog )
EXECUTION
↑C
.SAVE )
.GET )
JOB SETUP
.E 400010 )
400010/777777 777777 .D Ø Ø )
.E )
400010/ Ø Ø
.GET )
JOB SETUP
.E 400010 )
400010/777777 777777                    ;the original file
```

Note that there are applications for a sharable data segment when the modified version of the sharable segment is wanted rather than the original segment as initialized from the file. The SSAVE command is then used.

A SAVE of a one-segment program and a SSAVE of a two-segment program of the same name can coexist in the same directory, and the monitor keeps the two versions separate. This allows for a common library, of reentrant and non-reentrant versions of the same system programs to service both the PDP-6 and the DECsystem-10. A sharable program may be superseded into the directory by the SSAVE command. The monitor clears the high segment in its table of sharable segments in use but does not remove the segment from the addressing space of users currently using it. Only the users doing a GET, R, or RUN command or a RUN or GETSEG UUO have the new sharable version.

When the SAVE or SSAVE command is used to save a sharable program with only a high file, the monitor does not modify the vestigial job data area. This prohibits unauthorized users from modifying the first 10 locations of a shared segment by executing a SAVE or SSAVE command. This restriction does not exist if a low file is also written, because the GET command reads the low file after the high file, so that the real job data area locations are set from the low file. To change the version number of a sharable two-segment program with only a high file, the following commands are used.

```
GET prog
SAVE
GET
D nnn mmm 137
SSAVE
```

The SAVE command makes the program non-sharable so that the vestigial job data area can be modified by the SSAVE.