Chapter 7
The Batch Processing Program: BATCH

The ability of the Batch Processor to execute a stream of commands allows you to submit jobs to be run without terminal dialogue. Because Batch runs from a pseudo keyboard, your terminal is freed for other use during Batch processing. Batch processing is particularly useful in data processing operations that do not require interaction.

Batch input can be submitted from control files created with PIP or an editor on a random access device. For purposes of this description, input is dealt with as though it were on cards, where each card represents one record. Such input consists of elements of the batch control language and is collectively referred to as a batch stream. It is possible to execute multiple streams simultaneously by running multiple copies of the BATCH program. The capability to run more than a single batch stream is controlled by the system manager.

Sections 7.1 through 7.3 discuss the elements of the batch control language. Operating procedures are described in Section 7.4.

7.1 Control Statements

Batch control language statements consist of a command field, specification field(s) and a comment field, in the following format:

```
$[command-field] [specification-field(s)] [!comment]
```

Fields must be separated by one or more spaces and/or tabs.

A command field must always be present and may contain switches to control or limit the command. When appropriate, the command field is followed by one or more specification fields. The ! character is a comment prefix signifying that the information between the ! character and the line terminating character is a comment. The system takes no action on comment information. Note that the comment character (!) only applies to Batch command lines. If a comment line is included in data statements, BATCH treats it as data.

The hyphen (–) character can indicate a continuation of a command. If a hyphen is the last character in a command, the next line is treated as a continuation of the previous line and must begin with a $ followed by a blank. The hyphen must appear before any comment.

A physical command line can have a maximum of 120 characters.

Quotation marks may be used in control statements to reproduce some text identically and override any special interpretation of characters by BATCH. For example, the exclamation point (!) is, in command environments other than DCL, the designator for the auxiliary library account [1,3] or for an installation defined account. In BATCH, the exclamation point signifies a comment. To prevent BATCH from misinterpreting the ! character given as an account designator, include quotation marks as shown in the following sample control statement:

```
$RUN "!UPDAT"
```

As a result, BATCH executes the program UPDAT from the auxiliary library account. Without the quotation marks in the preceding example, the characters following the ! character are treated as comment characters. In the BASIC–PLUS command environment, RUN without a program name causes the program currently in memory to be executed.

7.1.1 Command Field

The command field consists of the following elements:

1. A $ (dollar sign) character is in the first character position. The $ character is the control statement recognition character.
2. The command name begins in the second character position and immediately follows the $ character. For example, $JOB. No blanks are allowed in the command field.
3. Valid switches follow the command name. No blank can appear between the command name and a switch. Switches are denoted by a slash (/). For example, /NAME=COMPL.
4. A blank (or horizontal tab) delimits the command field.

NOTE

> Command names and switch names can be shortened to their first three characters. For example, BATCH interprets the command $BAS as well as $BASIC.
>
> Multiple, adjacent blank characters are equivalent to one blank character. A horizontal tab is equivalent to one blank character. A blank character delimits a field; otherwise the blank character is ignored.

7.1.2 Specification Fields

A list of specification fields immediately follows the command field delimiter. The following rules apply:

1. Specification fields are separated by blanks.
2. Specification fields are terminated by a ! character if followed by a comment, or are otherwise terminated by a line terminating character.
3. Depending on the command, a specification field consists of a device specification, a file specification, or an arbitrary ASCII string, any of which can be followed by appropriate switches. The / character signals the start of a switch. For example, XYZ.BAS/SOURCE. The switch indicates that the file is a source file.

7.1.3 Comments

The following rules govern comment fields:

1. The start of a comment is defined by an ! character in the control statement.
2. Any character following an ! character and preceding the end-of-line terminator is treated as a comment and is otherwise ignored by the Batch processor. Comment lines with no text may force line spacing on the job log and thereby make the log more readable. To force line spacing, include lines consisting solely of $! followed immediately by a carriage return/line feed.

7.1.4 Syntactical Rules

The following are syntax rules for control language statements:

1. A control statement must have a command name (except in the case of the comment line "$!"). If the command name is omitted, the command is ignored. An unrecognizable command name is illegal, and causes the Batch processor to display an error message.

   Switches in the command field apply to the entire command. If a switch in the specification field contradicts a command field switch, an error results.

2. An asterisk is allowed in the file name or the file type field of a file specification, subject to restrictions on individual commands. See Section 7.2 for the description of file specifications. An asterisk can refer only to files already created. An asterisk appearing in a specification of a file not yet created constitutes an invalid file specification.

   The Batch processor uses the leftmost 6 characters from file name fields longer than 6 characters and uses the leftmost 3 characters from the file type fields longer than 3 characters.

3. Switches can be used in the command field and specification fields of a control statement. Switches appearing in the command field are command qualifiers, and their function applies to the entire command. Switches appearing in specification fields apply only to the field in which they appear.

Unrecognizable switches invalidate the control statements in which they appear.

7.1.5 Syntax Example

The following are sample control statements which illustrate the syntax of Batch statements:

```
$JOB/NAME=SMYTHE !FIRST JOB
$!
$!COMPILATION OF NEW SOURCE FILES
$!
$MESSAGE STARTING COMPILATIONS
$BASIC XYZ/SOURCE XYZ.LIS/LIST XYZ/EXECUTE
$BASIC ABC/SOURCE ABC.LIS/LIST ABC/EXECUTE
$!
$MESSAGE STARTING LISTING OUTPUT
$!
$PRINT *.LIS! ALL LIST FILES
$!
$EOJ
```

Table 7–1: BATCH Special Characters

Character	Meaning
space	Separates fields in a control statement. Otherwise ignored unless embedded in a string delimited by quotation marks.
horizontal tab	Separates fields in a control statement. (Equivalent to one space (blank) character; otherwise ignored.)
hyphen(–)	As last nonblank character in a control statement, indicates that a continuation line follows. If the statement contains a comment, the hyphen must be last nonblank character before the exclamation point.
exclamation point (!)	Indicates a comment unless embedded in a string delimited by quotation marks.
dollar sign ($)	Used as first character in first position of a control statement; causes control statement recognition.
slash (/)	Denotes a switch (separates specification field from switch name).
asterisk (*)	Indicates wildcard in file name or file type.

(continued on next page)

Table 7–1: BATCH Special Characters (Cont.)

Character	Meaning
colon (:) equals (=)	Separate switch name from argument.
quotation marks (") single quotation mark (')	Used to open and close a string to preserve embedded spaces or to pass a special character (such as !) without interpretation by BATCH.
plus (+)	Indicates file concatenation in $COPY command.
comma (,)	Separates file, device, and/or account specifications within a specification field which allows multiple elements.

7.2 File Specifications

A file specification appears in the specification field and is an alphanumeric string containing the following elements:

```
filename.typ
```

The Batch processor assigns default values if part or all of the file specification is optionally omitted.

7.2.1 File Name

A file name is a string of one to six unique alphanumeric characters. An asterisk in place of a file name denotes all files of the specified type in the account designated. If necessary, the Batch processor generates a default file name related to the time of day as described in Section 7.2.3.

7.2.2 File Type

A file type consists of a period immediately followed by a string containing three unique alphanumeric characters.

The file type reflects the nature of the file. For example, a BASIC–PLUS source file has .BAS as its file type. An asterisk in place of a file type denotes all file types, including files with no types specified.

Some standard file types are listed below:

.CTL	Batch control file.
.DAT	Data file.
.DIR	Directory file.
.BAS	BASIC–PLUS source file.
.LIS	List file.
.BAC	BASIC–PLUS compiled output file.
.OBJ	FORTRAN compiled output file.
.SRT	PDP–11 SORT–11 input, output, or listing file.
.MAP	Task Builder map file.
.TMP	Temporary file.
.B2S	BASIC–PLUS–2 source file.
.TSK	Task built executable file (from TKB).
.FOR	FORTRAN source file.
.SAV	RT11 run-time system executable file (from LINK).

These file types are the defaults when no file type is specified. The default chosen is determined by the current operation and by the type of file expected. Table 7–2 summarizes the default file types that apply to particular batch commands.

Table 7–2: BATCH Commands – Related Default File Types

Command / Section	Default File Type	Meaning
$BASIC 7.3.3	.BAS .B2S	Input source file default (BASIC–PLUS) (BASIC–PLUS–2).
	.BAC .TSK	Output executable file default (BASIC–PLUS) (BASIC–PLUS–2).
	.LST	Listing file default.
$CREATE 7.3.4.5	.DAT	The file generated as output by CREATE has a file type of .DAT.
$DIRECTORY 7.3.4.4	.DIR	The file in which the directory is to be recorded has a file type of .DIR.
$FORTRAN 7.3.12	.FOR	Input source file default.
	.OBJ	Output object file default.
	.LST	Listing file default.
	.SAV	Executable linked file.
$JOB 7.3.1	.CTL	Batch control file default; assumed when the Batch job is on a file-structured device.
	.LOG	Batch output log file default.
$PRINT 7.3.4.3	.LIS	Default of file to be printed.
$RUN 7.3.5	--	Check internal default runnable file type list for installed run-time systems. This ordering shows up in SYSTAT.
$SORT 7.3.11	.SRT	Input or output file default.

7.2.3 Defaults

Defaults are assigned to omitted file name and file type elements as shown in Table 7–3.

Table 7–3: File Specification Defaults

Condition	Default	Example
Name specified but no file type	Default assigned as appropriate to the current operation. For example, with the BATCH command BASIC, the default is .BAS.	ABC = ABC.typ

(continued on next page)

Table 7–3: File Specification Defaults (Cont.)

Condition	Default	Example
Name, followed by dot, but no file type	Default file type is null. No file type is assigned.	ABC. = ABC
File type, but no name	Default file name is related to time of day.	.LIS = B2347P.LIS (created at 01:23:47 PM)
No file specification	Default file name (related to time of day) with default type as appropriate to current operation.	Null = B2347P.typ

7.2.4 Switches

Switches consist of a / character followed immediately by a name. If the switch takes an argument, the argument is separated from the switch name by a colon (:) or equal sign (=). If the switch takes an argument and subarguments, each subargument is separated from the argument and from other subarguments by a colon. For example:

```
/NAME=JOB3
/VID="MY TAPE"
```

Switches accept arguments such as decimal constant, alphanumeric string, and date-time.

Switch values can be negated by putting the characters NO between the / character and the switch name. For example:

```
/NOOBJ
```

This switch indicates that no object file is to be produced. Its most frequent use is in conjunction with the $BASIC command.

NOTE

The negation characters NO are not considered part of the switch name. Thus, a negated switch must contain at least five characters. For example:

`/NOOBJ` or `/NOOBJECT`

is valid, but

```
/NOO
```

is invalid.

7.3 Batch Commands

The BATCH command set consists of:

$JOB	Begins a job.
$EOJ	Ends a job.
$BASIC	Executes the BASIC–PLUS interpreter or BASIC–PLUS–2 compiler.
system command or $system command	Executes a system utility function.
$RUN	Executes a program.
$DATA	Begins data images.
$EOD	Ends data images.
$MESSAGE	Logs a message on the operator services console.
$MOUNT	Assigns a device.
$DISMOUNT	Deassigns a device.
$SORT	Executes the PDP–11 Sort program SORT–11.
$FORTRAN	Executes the FORTRAN compiler.
$DELETE	Deletes files.
$COPY	Copies files.
$PRINT	Queues a file for the default line printer.
$DIRECTORY	Lists an account directory.
$CREATE	Creates a file from data in the input stream.

NOTE

The $BASIC, $FORTRAN, and $SORT commands are subsets of BASIC, FORTRAN, and SORT capabilities. Therefore, to use the full capabilities of BASIC, FORTRAN, or SORT, use the BATCH command $RUN.

7.3.1 $JOB

This command marks the beginning of a job. The following command switches are allowed:

/NAME=jobname — Assigns a name to the job. Job names can be up to 6 characters long. This name overrides the control file name as the identifier of the job.

/NONAME — Indicates that no job name is defined. A default job name is assigned. The default job name is the name of the control file. The name appears in all messages to the system operator.

/LIMIT=nnn — Assigns an elapsed time limit to the job. The value of nnn, a decimal number, is interpreted as minutes. Note that the elapsed time taken to execute a job is heavily dependent on overall system loading.

/NOLIMIT — Gives the job an unlimited amount of elapsed time to complete. If neither /LIMIT:nn nor /NOLIMIT appear, the job is given 10 minutes elapsed time to complete execution before BATCH terminates it.

/CPU:nnn — Assigns a CPU time limit to the job. The value of nnn, a decimal number, is interpreted as seconds. If /CPU is specified, and /LIMIT is not specified, no elapsed time limit is enforced, and the only time limit is on CPU time. If both switches are specified, both limits are enforced. If no CPU time limit is specified, the allowable CPU time is infinite.

/NOCPU — Gives the job an unlimited amount of CPU time to complete.

/QUE — Queues the batch log file to LP0 (default) or the device specified at system start up.

/NOQUE — Suppresses printing of the batch log file.

/PRIORITY:n — Sets the RSTS/E job priority to n (or the next lowest multiple of 8) for the BATCH stream. For privileged users, n can be between –120 and +127; for non-privileged users, n is limited to a value between –120 and –8. Unless otherwise altered by the /PRIORITY:n switch, all jobs run at –8 priority.

/DCL — Disables all current BATCH commands that conflict with DCL commands and allows them to be processed by DCL.

/CCL Allows the use of the system's interactive Concise Command Language. When this switch is specified, any of the system commands that do not conflict with existing Batch commands can follow the $ character. The Batch processor ensures that the job is waiting for input for the keyboard monitor (the ^C state reported by SYSTAT) before executing the command.

/ERROR:[operand] Specifies the level of error which the Batch processor tolerates without terminating the job. The level is indicated by [operand], which may be FAT[AL], WAR[NING], or NON[E]. If FATAL, all errors are tolerated until completion. If WARNING, a fatal error terminates the job, but warning errors are tolerated. If NONE, any error terminates the job. If a job is to be terminated because the error level has been exceeded, termination occurs when the job next asks for input. A message is entered in the log file giving the reason for termination. The default error level for the BATCH stream is determined at start-up time.

NOTE

Refer to Table A–1 for the severity standards in error messages. Some programs (for example, the FORTRAN IV compiler, which operates under a run-time system other than BASIC–PLUS) do not use the standard severity characters in error messages. Therefore, compilation errors are not detected by Batch processing. User programs that are coded to run under Batch control must use standard severity characters.

The following specification field may be included:

[n,m] To have the job executed on an account other than that under which it was queued, a specification field may indicate the account number desired. This feature can be used only by a privileged user. Remember from the discussion in Section 7.1.2 that specification fields are separated from other parts of the command by a space.

For example:

```
$JOB/NAME=TEST/NOLIMIT/CCL [1,2]
```

The following error conditions are possible:

Unrecognized switch
Illegal switch value
Multiple conflicting specifications (switches)
Different account specified by nonprivileged user
Higher priority desired by nonprivileged user

7.3.2 $EOJ

This command marks the end of a job. The $EOJ command automatically dismounts all devices mounted by the job. $EOJ prints an appropriate message to the operator that the logical device should be dismounted. A logical deassignment is performed.

NOTE

1. The $EOJ command is implied when BATCH encounters a physical end-of-file condition or another $JOB control statement while processing a control file.
2. No switches are legal in the $EOJ command.

7.3.3 $BASIC

The $BASIC command calls a BASIC compiler, which compiles a source program. The format of the $BASIC command is:

```
$BASIC[switches][specification field [switch]][specification fields]
```

The following switches are valid in the command field:

/BP1

Use the BASIC–PLUS compiler. If neither /BP1 nor /BP2 appears, /BP1 is used. The default file type is .BAS.

/BP2

Use the BASIC–PLUS–2 compiler. If neither /BP2 nor /BP1 appears, /BP1 is used. The default file type is .B2S.

NOTE

> When $BASIC appears in the BATCH command file, the Batch processor assumes that BASIC–PLUS is the default keyboard monitor. If this is not the case and the BASIC–PLUS compiler is desired, use the SWITCH program to switch to the BASIC–PLUS keyboard monitor before the $BASIC command in the BATCH command file.

/RUN

Execute (only) a previously compiled/task built program. If this switch is used, the entire command line must have one and only one file specification, and can have only one other switch: /EXECUTE.

/NORUN

Perform the compile/task build procedure, but do not execute the final file.

/OBJECT
(legal only for
BASIC–PLUS–2
runs; see
/EXECUTE)

Create the object file filename .OBJ, where the file name is that of the source file. This switch implies /BP2, and therefore causes BASIC–PLUS–2 to be run; it also causes a task build operation.

/NOOBJECT

Create the object file filename.TMP, where the file name is that of the source file. Delete this .TMP (temporary) file upon completing the command. This switch implies /BP2, and therefore causes BASIC–PLUS–2 to be run. Thus, like /OBJECT, it is legal only in a BASIC–PLUS–2 run.

/LIST

Produce the listing file filename.LST, where the file name is that of the source file. /NOLIST is the default.

/NOLIST

Do not produce a listing file. /NOLIST is the default.

/MAP

Create the task builder map file filename.MAP, where the file name is that of the source file. This switch implies /BP2, and therefore causes BASIC–PLUS–2 to be run; it also causes a task build operation. Thus, it is legal only in a BASIC–PLUS–2 run. /NOMAP is the default.

/NOMAP

Do not create a map file. This switch implies /BP2, and therefore causes BASIC–PLUS–2 to be run. Thus, it is legal only in a BASIC–PLUS–2 run. /NOMAP is the default.

/EXECUTE
(replaces
/OBJECT for
BASIC–PLUS runs)

Create an executable file, whose file name is that of the source file. Choose its type according to the following rules:

If the language is BASIC–PLUS, give the file a .BAC type (filename.BAC).

If the language is BASIC–PLUS–2, give the file a .TSK type (filename.TSK).

/NOEXECUTE

If an executable file is needed, create a temporary one (.TMP), and delete it upon completion of $BASIC processing.

One of the following switches may appear in the first specification field, described in the format guide at the start of this section:

/SOURCE
/BASIC

Both switches mean that this is the BASIC–PLUS or BASIC–PLUS–2 source file on which to operate.

/EXECUTE

This switch is legal only if /RUN appears in the command field, and means that this is an executable file.

In this field, any specification lacking a switch is assumed to be the input file for the command. Thus, only one file specification may appear without switches. This specification may not contain wildcards (neither asterisks nor question marks). If /NORUN appears in the command field, the lack of a switch here implies /BASIC. If /RUN appears in the command field, the lack of a switch here implies /EXECUTE. And if neither /RUN nor /NORUN appears in the commmand field, the lack of a switch implies /BASIC.

The optional specifications ending the format description (at the beginning of this section) define other files which may be needed in the operation. Any of the following switches may be used in the formats indicated. Each switch, however, may be used only once in the entire $BASIC command line. Moreover, its negation cannot be used anywhere in the command line (/NOLIST and /LIST, for example, cannot appear together in a command line).

file specification/LIST	Define the listing file as specified.
file specification/OBJECT	Define the object file as specified (switch implies /BP2 and causes BASIC–PLUS–2 to be run, with task build; switch is legal only with BASIC–PLUS–2).
file specification/MAP	Define the map file as specified (switch implies /BP2 and causes BASIC–PLUS–2 to be run, with task build; switch is legal only with BASIC–PLUS–2).
file specification/EXECUTE	Define the executable file as specified.

If a source file is not specified, the $BASIC command must be followed by a set of BASIC source statements, terminated by either $EOD (see Section 7.3.7) or some other recognized Batch control statement. For example:

```
$BAS     LSTING/LIS

  BASIC
  Source
  Deck

$EOD
```

If a source file is explicitly specified, any source statements following this command are appended to the source program. Source statements that follow this command and have line numbers equal to those in the source program replace those in the source program. Source input must be provided, either through a file specification, or through source statements, or both.

If no listing file is specified but the /LIST switch is present, the Batch processor creates the default listing file. If a file specification appears with the /LIST switch, the Batch processor uses that specification for the file. To print the file specified as part of the Batch job, supply a $PRINT control statement described in Section 7.3.4.3.

If no executable file is specified with the /EXECUTE switch, a default executable file is created and is deleted after job completion. If an executable file is explicitly specified, it is preserved after job execution. Errors result from conflicting switch specifications such as both /BASIC and /SOURCE on different specification fields.

The default applied when a file is specified without a switch is /SOURCE.

The following error conditions are possible:

Unrecognized switch
Multiple conflicting specifications (switches)
File specification syntax error

7.3.4 Utility BATCH Commands

The following utility functions are provided by the RSTS/E Batch processor:

$DELETE	Deletes files.
$COPY	Copies files.
$PRINT	Prints a file on the system line printer by means of the line printer spooling program SPOOL.
$DIRECTORY	Lists a file directory.
$CREATE	Creates a file from data in the input stream.

7.3.4.1 $DELETE — The $DELETE command deletes specified files. It has the format:

```
$DELETE file1 [file2 ... filen]
```

The file name and file type must be included. An asterisk is not valid in either the file name or the file type field.

No switches are used with $DELETE.

The following error conditions are possible:

No file specification
Syntax error in file specification

7.3.4.2 $COPY — $COPY copies files. Use of the asterisk character in the file specification is invalid. The following are the valid switches:

/OUTPUT	For new files to be created.
/INPUT	For files to be copied.

For example:

```
$COPY TER.LIS/OUTPUT TERRY.LIS/INPUT
```

The $COPY command supports the use of + (plus sign) to indicate file concatenation. When used with $COPY, file concatenation results in the creation of a single file, which consists of files connected together. The + character appears in the file specification field, between the specifications of files to be concatenated. If no switch is specified, /INPUT is assumed.

The following error conditions are possible:

No output specification
No input specification
Multiple conflicting specifications
Syntax error in file description

7.3.4.3 $PRINT — The $PRINT command prints the contents of files on the system line printer by means of the spooling program SPOOL. File specifications accept all switches available in the Q command (see Section 6.13). Asterisks can be used in file specifications. The $PRINT command has the following format:

```
$PRINT file1/switches [file2 ... filen]
```

The specification field contains the file or files to be printed.

The following error conditions are possible:

No file specification
Syntax error in file specification

7.3.4.4 $DIRECTORY — $DIRECTORY produces a directory listing of the file(s) in the specified account and has the format:

```
$DIRECTORY [specification field]
```

The specification field can contain file specifications. If no file specification appears, the $DIR command lists the contents of the current account in the Batch log file. A file specification indicates the directory of a file or set of files and can contain an asterisk in either the file name field or file type field. For example:

```
$DIR *.BAS
```

This command creates a directory listing of all files in the current account with the .BAS file type.

To create a directory in a disk file rather than on the Batch log device, specify a file and the /DIRECTORY switch. For example:

```
$DIR BAJOB.DIR/DIR
```

creates the directory listing in a file BAJOB.DIR on the system disk under the current account.

To create a directory in a disk file and to designate which files are to be listed, specify both the /DIRECTORY and /INPUT switches with the related file specification. For example:

```
$DIR BA.DIR/DIR *.BAC/INPUT
```

The $DIR command in this example creates a directory listing of all compiled BASIC–PLUS files and stores the listing in the file BA.DIR on the system disk under the current account.

The following error conditions are possible:

Syntax error in file specification
Multiple conflicting specifications

7.3.4.5 $CREATE — The $CREATE command creates an ASCII file as indicated in the specification field. The file consists of the data following the $CREATE command in the input stream. Data must follow $CREATE and must be terminated by $EOD, or an error occurs. The data must not be preceded by any other command because the $CREATE function terminates on encountering a $ in the first column of a line.

Any previously existing file of the name specified is deleted at batch execution time, and replaced by the file created by the $CREATE command.

The $CREATE command has the following format:

```
$CREATE file
```

The following error conditions are possible:

Syntax error in file specification
No file name specified
Non-comment characters following file specification

7.3.5 $RUN

The $RUN command causes execution of system programs. For example, to run PIP, type:

```
$RUN $PIP
```

followed by appropriate PIP commands. The PIP program reads the commands as data images in the input stream. Execution of PIP is terminated when the next Batch control statement is read.

No switches can be specified. The general format of $RUN is:

```
$RUN [file]
```

where file specifies the executable program. If file is omitted, the default current program is used.

The following error conditions are possible:

Syntax error in file specification
Non-comment characters following file specification

7.3.6 $DATA

The $DATA command provides a means of entering data to a program that is compiled and run by one of the language commands (e.g., $BASIC, $FORTRAN). $DATA ensures that the program will be run, unless the /NORUN switch was specified. It also ensures that if the program does not use all of its data, the remaining data will be flushed from the stream.

The $DATA command is issued without specification fields or switches, in the following format:

```
$DATA
```

7.3.7 $EOD

The $EOD command marks the end of data records included in the input stream following commands such as $BASIC, $CREATE, $DATA, and $RUN. For example:

```
$DATA
  .
  .
  .
data
  .
  .
  .
$EOD
```

7.3.8 $MESSAGE

The $MESSAGE command logs a message on the operator services console. It provides a way for the job to communicate with the operator. The command has the format:

```
$MESSAGE[/WAIT] message-string
```

The /WAIT switch indicates a pause to wait for operator action. The system pauses until the operator gives the appropriate command. For example, the following command halts the program until the operator takes action:

```
$MESSAGE/WAIT MOUNT SCRATCH TAPE ON DTO:
```

The WAIT condition remains in effect until the operator responds to the message on the operator services console. For information on operator response procedures, refer to the *RSTS/E System Manager's Guide*.

7.3.9 $MOUNT

The $MOUNT command causes a mount message to be printed on the operator service console, and causes a logical to physical device assignment. The physical device refers to a physical device type. The operator responds with the device and unit number in the standard format (e.g., MT1:). An automatic /WAIT occurs. Logical device names of up to six characters are used to specify logical devices.

The $MOUNT command has the format:

```
$MOUNT devn:[/switch] devm:[/switch]
```

Both the logical device and the physical device must be specified. The colon is required as the terminator for each device specification. The following switches can be used for the physical device:

/PHYSICAL — Identifies the device specification to be the physical device (default).

/WRITE — Tells the operator to write-enable the device (or volume).

/NOWRITE — Tells the operator to write-protect the volume.

/VID:[string] — [string] is a visual identification which identifies the volume for the operator.

/DEN[SITY]:nnn — Specifies density for magnetic tape.

/PAR[ITY]:[ODD][EVEN] — Specifies odd or even parity for magnetic tape.

The following switch is used with the logical device:

/LOGICAL — Identifies the device specification to be the logical device name; this specification must correspond to the PACK ID for RSTS/E disks.

The /VID switch on the physical device field specifies the volume identification. The value associated with /VID is the name physically attached to the volume. It is included to help the operator locate the volume. The volume identification cannot contain slashes (/) or commas (,).

If the name specified with /VID must contain blanks, it can be delimited by quotation marks. For example:

```
/VID="FJM JT"
```

Blanks are not allowed in a string without quotation marks. For example:

```
$MOU MT:/PHY/VID="MY TAPE" TAPE:/LOG
```

In this example, logical device name TAPE is assigned to a magnetic tape unit. The operator is told that the reel of tape to be physically mounted is labeled MY TAPE. The operator uses a PLEASE command (see the *RSTS/E System Manager's Guide*) to respond with the device and unit number on which the tape is mounted. Thereafter, in the Batch command file, reference to the device TAPE: accesses the physical device on which the operator mounted the reel MY TAPE. If the physical device is a removable disk pack or cartridge, the logical device name must be the pack identification. The Batch processor logically mounts and unlocks private disks that the operator mounts as a result of $MOUNT.

The valid physical devices that can be requested for mounting are:

CR:	Card Reader
DK:	RK11/RK05 Disk Cartridge
DP:	RP11/RP03/RP02 disk pack
DB:	RH11/RP04/RP05/RP06 disk pack
DM:	RK611/RK06/RK07 disk pack
DR:	RM02/RM03/RM05 disk pack
DF:	RF11 disk
DS:	RH11/RS03/RS04 disk
DL:	RL01/RL02 disk
DX:	RX01/RX02 flexible diskette
DT:	TU56 DECtape
DD:	TU58 DECtape II
LP:	Line printer
MT:	TE10/TU10/TS03 magnetic tape
MM:	TE16/TU16/TU77/TU45 magnetic tape
MS:	TS11 magnetic tape
PP:	Paper tape punch
PR:	Paper tape reader
SY:	System device
KB:	Terminal

The following error conditions are possible:

Syntax error in device specification fields
Invalid device name/unit
Invalid logical device name specifications
Unit number already assigned
Both physical and logical names have not been specified

7.3.10 $DISMOUNT

The $DISMOUNT command causes the logical to physical device assignment effected by the $MOUNT command to be nullified. It also prints an operator message, requesting that the volume be dismounted. If a /WAIT switch is included in the command field, the job will not resume until a response, as with the $MESSAGE command, is received from the operator. For example:

```
$DIS/WAI TAPE:
```

sends a message to the operator to dismount the magnetic tape that was mounted by the example in Section 7.3.9. As a result of the switch /WAI, BATCH pauses until the operator responds.

All devices are automatically dismounted at end-of-job ($EOJ).

The following error conditions are possible:

Syntax error in specification field
Illegal switches
Logical device not assigned

7.3.11 $SORT

The $SORT control statement causes the execution of the SORT–11 program, which is supplied on RSTS/E systems. For additional information on the SORT–11 program, refer to the *PDP–11 SORT Reference Manual.*

The format of the $SORT control statement is:

```
$SORT [job switches] [output [/OUTPUT]] [input [/INPUT]] [spec /SPEC]
```

Job switches define the sort process and can be abbreviated to the first three letters. When more than one switch is specified, they are separated by slashes (/). The following are valid job switches:

/ALL[OCATION]:n
For output, specifies the initial space allocation for the file. The value n is in the range of 0 to 65535 blocks. If no allocation is specified, the default depends on the sort process (see /PROCESS). If /PROCESS is Record or Tag, the default is the input file size. If /PROCESS is Index or Address Routing, the default is the number of records sorted.

/BLO[CKSIZE]:n
For magnetic tape input or output, specifies the tape block size. The default is a 512–byte block.

/BUC[KETSIZE]:n
For output, specifies the file's bucketsize. The default is the bucketsize of the input file.

/CON[TIGUOUS]
For output, specifies that the file will be contiguous. The default is noncontiguous.

/DEV[ICE]:x
For input, specifies the device to be used for scratch files, where x is a 1– to 4–character device name.

/FIL[ES]:n
For input, specifies the number of scratch files, where n is in the range of 3 to 10.

/FOR[MAT]:x:n
For input, specifies the file's record format (x) and maximum record size (n). The value x can be FIXED, STREAM, VARIABLE, or UNKNOWN and can be abbreviated to the first letter. This switch is required. The default format is VARIABLE; the record size must be specified.

/IND[EXEDSEQUENTIAL]:x For input, specifies indexed file organization. The value x specifies the number of keys.

/KEY[S]:abm.n For input, specifies the sorting key field. This switch is required if no specification file appears in the command line. A maximum of 10 keys (separated by colons) can be specified as follows:

```
/KEY:abm.n:abm.n:...abm.n
```

/PRO[CESS]:x For input, specifies the sorting process. The value x can be R (Record sort, the default), T (Tag sort), A (Address Routing), or I (Index sort).

/REL[ATIVE] For output, specifies relative file organization.

/SEQ[UENTIAL] For output, specifies sequential file organization.

/SIZ[E]:n For output, specifies the file's cluster size.

A maximum of three file specifications (separated by spaces) can appear in the $SORT control statement; an input file, an output file, and a specification file. To distinguish these files, the following switches, which can be abbreviated to the first three letters, are used:

/INP[UT] The file to be sorted.

/OUT[PUT] The file to contain the sorted data.

/SPE[CIFICATION] The file which contains the control information for the sorting process.

A file specification without a switch is used as the file to be sorted. If the /SPECIFICATION switch is used, the /KEYS and /PROCESS switches must not appear in the command line. If a specification file is not given in the control statement, the /KEYS switch must be included in the command field to control the sorting process; the /FORMAT switch must always be included in the command field. If a type is omitted from the file specification, BATCH uses .SRT as the type.

7.3.12 $FORTRAN

The $FORTRAN command calls the FORTRAN–IV compiler, which compiles the source program and generates an object program. The format of the command is:

```
$FOR[TRAN][switches][specification field[switch]][spec fields[switch]]
```

The following switches are valid in the command field:

/RUN — Execute the previously compiled file. Only an object file can be specified.

/NORUN — Compile the source program but do not execute the object file.

/OBJECT — Create the compiled file filename.OBJ, where the file name is that of the source file. If neither /OBJECT nor /NOOBJECT appears, /NOOBJECT is used.

/NOOBJECT — Do not create an object file. If neither /NOOBJECT nor /OBJECT appears, /NOOBJECT is used.

/LIST — Produce the listing file file name.LST, where the file name is that of the source file. If neither /LIST nor /NOLIST appears, /NOLIST is used.

/NOLIST — Do not produce a listing file. If neither /NOLIST nor /LIST appears, /NOLIST is used.

/MAP — Create the map file filename.MAP, where the file name is that of the source file. If neither /MAP nor /NOMAP is specified, /NOMAP is used.

/NOMAP — Do not create a map file. If neither /MAP nor /NOMAP is specified, /NOMAP is used.

One of the following switches may appear in the first specification field described in the format guide at the start of this section:

/FORTRAN
/SOURCE — Both switches have the same meaning: i.e., that this is the source file on which to operate. If a file specification lacks a switch, it is assumed to be a source file.

The optional specifications ending the format description define other files which may be needed in the operation. Any of the following switches may be used in the formats indicated. Each switch, however, may be used only once in the entire $FORTRAN command line. Moreover, its negation cannot be used anywhere in the command line (/NOLIST and /LIST, for example, cannot appear together in a command line).

file specification/LIST	Define the listing file as specified.
file specification/OBJECT	Define the object file as specified.
file specification/MAP	Define the map file as specified.

Multiple specifications of the following form may appear anywhere in the command line, delimited by spaces:

```
file specification/LIBRARY
```

Each such specification is a library file that will be linked with the FORTRAN program.

7.4 Batch Operating Procedures

This section describes how you request Batch processing and how the Batch processor generates output.

7.4.1 Requesting a Batch Job Run

To submit a BATCH job, run the library program QUE and specify the Batch control file or files as follows:

```
RUN $QUE
QUE   V7 RSTS V7 Timesharing
#Q BA:BATJOB=FILE1,FILE2,FILE3.DAT
#
```

You normally queue a BATCH job to device BA:. The job and log files in this example will be named BATJOB, and the files FILE1.CTL, FILE2.CTL, and FILE3.DAT will be concatenated to form the Batch control file. The log file BATJOB.LOG will be printed after the job is complete providing the /NOQUE switch was not specified in the $JOB command or during startup of the BATCH processor. Note that QUE schedules jobs on a strict priority basis, thus a high priority compute bound job can slow processing.

The CCL command QUEUE, if available on the system, may also be used to submit a job for Batch processing. Also, the CCL command SUBMIT, if available on the system, can be used. However, unlike QUEUE, the SUBMIT CCL command only accepts file name specifications (separated by commas) and the Q command switches /DE and /MORE.

7.4.2 Batch Processing

As the Batch control file is read, it is checked for command sequence and syntactical validity. If an error is detected, an error message is printed in the log file. The job will not be run, but syntax checking will continue through the remainder of the file(s).

A $MESSAGE/WAIT, a $MOUNT, or a $DISMOUNT/WAIT will cause the job to pause for an operator response. Until the operator takes action, no further commands will be sent to the pseudo keyboard.

If no errors are detected, the job is processed. A log is created, showing the sequence of Batch commands processed during the course of the job. If program output is directed to KB:, this output appears after the command that caused the program to execute. In the example that follows, a BATCH job named JOB1 has been run. The Batch control file contained the following sequence of commands:

```
$JOB/NAME=JOB1/LIMIT=4
$CREATE SUB1.BAS

        source statements

$EOD
$BASIC/BP2 LISTING/LIS MAIN/OBJ

        source statements

$DATA

        data

$PRINT SUB1.BAS
$EOJ
```

These commands have the following effect:

```
$JOB/NAME=JOB1/LIMIT=4
```

A job name of JOB1 is assigned to the job. This name appears on the job log along with the time and date of the job's execution. A time limit of four minutes is set. If the job is not finished in four minutes from its start (actual elapsed time), the job is terminated, and the appropriate error message is printed in the log.

```
$CREATE SUB1.BAS
```

A BASIC source file named SUB1 is created from data records which must follow the $CREATE command.

```
$EOD
```

The $EOD command signals the end of SUB1.BAS.

```
$BASIC/BP2 LISTING/LIS MAIN/OBJ
```

The source statements following this command are compiled by the BASIC–PLUS–2 compiler. A listing of the source statements is created in the file LISTIN.B2S, and the object data is placed in the file MAIN.OBJ. The temporary task built file has the type .EXE. This file is executed.

```
$DATA
```

The data to be read during execution of MAIN.EXE follows this command.

```
$PRINT SUB1.BAS
```

The source file created by $CREATE SUB1.BAS is printed. This command also has the effect of terminating data input to MAIN.EXE.

```
$EOJ
```

This command signals the end of job JOB1.

7.4.3 Error Procedures

When a syntax error is detected in a BATCH command, the job is not executed. Instead, an error log is printed listing all commands and data scanned along with the appropriate error message(s). The Batch log file always indicates all command lines scanned. If an error is found on a command line, the error message follows the command, marked with question marks (????????????). Scanning of the control file continues, but the job will not be executed.

If no syntax errors occur, the time of output of lines is indicated in the left margin of the log. All normal terminal interaction corresponding to the BATCH commands will appear in the log. Table 7–4 lists the BATCH error messages and their meanings.

Table 7–4: Summary of BATCH Error Messages

Message and Meaning
BATCH BEING SHUT DOWN The BATCH processor is going off-line and the job must be terminated.
CANNOT USE THAT ACCOUNT An account specification appeared in the $JOB command but the request did not come from a privileged user.
CANNOT INCREASE PRIORITY A /PRIORITY:n switch appeared in the $JOB command. The user was privileged but specified a value for n greater than 127. Or, the user was non-privileged and specified a value greater than –8.

(continued on next page)

Table 7–4: Summary of BATCH Error Messages (Cont.)

Message and Meaning
CONTINUATION MISSING The hyphen (–) character was the last nonblank character in a control statement to continue the statement on the next line, but the following line did not begin with a dollar sign ($) and a blank.
DEVICE NOT MOUNTED A $DISMOUNT command was present but the device indicated had not been mounted.
DISK MOUNT FAILURE The volume to be mounted was not correct (pack IDs did not match) or the device was in use by another job.
INVALID COMMAND An undefined command name followed the $ character in a statement but the /CCL command switch had not been specified in the $JOB command.
INVALID SPECIFICATION FIELD The specification given in a control statement is in the wrong format.
INVALID SWITCH The switch used in the command field or in the specification field is undefined, in the wrong format, or is privileged.
NO BATCH JOBS POSSIBLE AT THIS TIME The Batch processor requires a pseudo keyboard to execute a job but one is not available. Requeue your request.
NO SUCH ACCOUNT The account specified in the $JOB command or in a specification field could not be found on the device.
SEQUENCE NOT SUPPORTED YET The $SEQUENCE command is not available with this version of BATCH.
TIME LIMIT EXCEEDED Time specified in $JOB command is insufficient to execute the job. Specify a larger limit by using /LIMIT=nnn or /NOLIMIT switch.
TOO MANY MOUNTED DEVICES The job has requested mounting of more devices than the maximum (12) allowed by Batch.

(continued on next page)

Table 7–4: Summary of BATCH Error Messages (Cont.)

Message and Meaning
UNABLE TO LOG IN BATCH JOB To execute a request, the Batch processor logs a job into the system using the account under which the job was queued or the account specified in the $JOB command. For some reason, the login procedure failed. For example, logins had been disabled. The job will be requeued for later execution.
UNMATCHED PARENTHESES An opening left parenthesis appears in a specification field but an accompanying closing parenthesis is not found.
UNMATCHED QUOTATION MARKS Quotation marks (and single quotation marks) must be paired in a control statement.

Appendix A
Error Messages

Messages in RSTS/E are generated for BASIC–PLUS errors* and RSTS/E errors. To avoid confusion, both types of messages are called RSTS/E error messages and are described as one set. The BASIC–PLUS errors cover compiler and run time conditions such as a violation of the syntax rules (?SYNTAX ERROR) and referencing an element of an array beyond the defined limits (?SUBSCRIPT OUT OF RANGE). The RSTS/E errors involve operating system conditions such as failing to locate the file or account specified (?CAN'T FIND FILE OR ACCOUNT) and requesting the hardware to perform a function for which it is not ready (?DEVICE HUNG OR WRITE LOCKED).

In most cases, if no error trapping is being done (that is, an ON ERROR GOTO statement is not in effect), BASIC–PLUS stops running the program. It prints the error message and the line number of the BASIC–PLUS statement that was being executed when the error occurred. The following sample printout shows the procedure:

```
10      OPEN 'Z' FOR INPUT AS FILE 1%
RUNNH
?CAN'T FIND FILE OR ACCOUNT AT LINE 10

READY
```

As the READY message indicates, control returns to the BASIC–PLUS keyboard monitor.

* Different messages are generated while a job is operating under run-time systems other than BASIC–PLUS. For these error messages, consult the appropriate User's Guides.

An exception to this procedure occurs when an INPUT statement is being executed at the job's console terminal and error trapping is not in effect. The system generates the error message and executes the statement again as shown in the sample printout below:

```
10      ON ERROR GOTO 0 \   INPUT 'INTEGER VALUE';A%
RUNNH
INTEGER VALUE? C
%DATA FORMAT ERROR AT LINE 10
INTEGER VALUE?
```

With error trapping disabled at line 10, an invalid response to the INPUT statement causes the system to print the error message, clear the error condition, and execute the statement again.

Associated with each message is an error variable called ERR. Whenever an error occurs with trapping in effect, the system checks the error variable which is a decimal number in the range 0 to 127. In BASIC–PLUS, an error with a number between 1 and 70 causes the system to transfer control to the line number indicated in the ON ERROR GOTO statement. The system does not print the error message. Your program is able to check the ERR variable and perform a recovery procedure. If the error number is between 71 and 127, the system does not transfer control to the recovery routine but prints the message and returns control to the system. (Error number 0 is reserved to identify the system installation name.) Note that the BASIC–PLUS–2 run-time system uses the error numbers 128 through 255 in addition to 0–127; refer to the *RSTS/E BASIC–PLUS–2 User's Guide.*

Because a BASIC–PLUS program can recover from certain errors, this appendix lists errors in two categories – recoverable and non-recoverable. The recoverable error messages are listed in ascending order of their related error numbers. A program can use these error numbers to differentiate errors. Non-recoverable errors are in alphabetical order without error numbers because a program can not use these numbers in an error handling routine.

The first character position of each message indicates the severity of the error. Table A–1 describes this standard.

Table A–1: Severity Standard in Error Messages

Character	Severity	Meaning
%	Warning	Execution of the program can continue but may not generate the expected results.
?	Fatal	Execution cannot continue unless you remove the cause of the error. No space or tab is allowed after the ?.
	Information	A message beginning with neither a question mark nor a percent is for information only.

The severity indication is examined by BATCH to determine the severity of the error.

In the descriptions of error messages, certain abbreviations, as shown in Table A–2, denote special characteristics of the error.

Table A–2: Special Abbreviations for Error Descriptions

Abbreviation	Meaning (BASIC–PLUS Environment)
(C)	Continue. If an ON ERROR GOTO statement is not in effect, execution continues but with the conditions described.
(SPR)	Software Performance Report. This error should occur only under the conditions described. If it occurs under any other conditions, have your system manager submit an SPR to DIGITAL and document the conditions under which the error occurred.

An error whose description is accompanied by the abbreviation (C) indicates an exception to the error trapping procedure. If such an error occurs in a program with no error trapping in effect, BASIC–PLUS prints the error message and line number but continues running the program. The following sample printout shows the procedure:

```
100     ON ERROR GOTO 0 \ A% = 32768.
200     PRINT A%
RUNNH
%INTEGER ERROR AT LINE 100
 0

READY
```

The INTEGER ERROR is generated at line 100 by the attempt to compute a value outside the range for integers. After the error message is printed, processing continues but with the conditions described in the error meaning. 0 is substituted for the erroneously computed value.

The number of RSTS/E error messages is restricted to 256. Because of this restriction, certain error messages have multiple meanings. The specific meaning of an error message depends on the operation being performed when the error condition occurs. For example, if the system attempts a file access and the designated file can not be located, RSTS/E generates the ?CAN'T FIND FILE OR ACCOUNT error (ERR=5). That same error condition, however, applies to other, generically similar access operations. Thus, if a program attempts to send a message to another program and the proper entry is not found in the system table of eligible receivers, RSTS/E returns error number 5. Though the second failure does not involve a file access error, it too is classified as an access failure.

Certain RSTS/E errors, although classified as user recoverable, are not capable of being trapped by a program. Table A–3 lists such errors.

Table A–3: Non-Trappable Errors in Recoverable Class

ERR	Message Printed
34	?RESERVED INSTRUCTION TRAP
36	?SP STACK OVERFLOW
37	?DISK ERROR DURING SWAP
38	?MEMORY PARITY FAILURE

These errors involve special conditions which a program cannot control and which ought not to occur on a normal system. For example, the ?DISK ERROR DURING SWAP error indicates a hardware problem. The system does not return control to the program. The error condition itself, however, can be either transient or recurring. Such errors should be brought to the attention of the system manager for further investigation. The errors are recoverable in the strict sense that the monitor can take corrective action but the BASIC–PLUS run-time system (and others) does not return control to the user program.

A.1 User Recoverable Errors

Message and Meaning **Error Code**

(SYSTEM INSTALLATION NAME) 0

The error code 0 is associated with the system installation name and is used by system programs to print identification lines.

?BAD DIRECTORY FOR DEVICE 1

The directory of the device referenced is in an unreadable format. The magnetic tape label format on tape differs from the system-wide default format, the current job default format, or the format specified in the OPEN statement. Use the ASSIGN command to set the correct format default or change the format specification in the MODE option of the OPEN statement.

?ILLEGAL FILE NAME 2

The filename specified is not acceptable. It contains unacceptable characters or the filename specification format has been violated. The CCL command to be added begins with a number or contains a character other than A through Z, 0 through 9 and commercial at (@).

?ACCOUNT OR DEVICE IN USE 3

Reassigning or dismounting of the device cannot be done because the device is open or has one or more open files. The account to be deleted has one or more files and must be zeroed before being deleted. The run-time system to be deleted is currently loaded in memory and in use. Output to a pseudo keyboard cannot be done unless the device is in KB wait state. An echo control field cannot be declared while another field is currently active. The CCL command to be added already exists.

?NO ROOM FOR USER ON DEVICE 4

Storage space allowed for the current user on the device specified has been used, or the device as a whole is too full to accept further data, or a contiguous file was specified and there is insufficient contiguous space.

?CAN'T FIND FILE OR ACCOUNT 5

The file or account number specified was not found on the device specified. The CCL command to be deleted does not exist.

?NOT A VALID DEVICE 6

The device specification supplied is not valid for one of the following reasons. The unit number or its type is not configured on the system. The specification is logical and untranslatable because a physical device is not associated with it.

?I/O CHANNEL ALREADY OPEN 7

An attempt was made to open one of the twelve I/O channels which had already been opened by the program. (SPR)

?DEVICE NOT AVAILABLE 8

The specified device exists on the system but an attempt to ASSIGN or OPEN it is prohibited for one of the following reasons. The device is currently reserved by another job. The device requires privileges for ownership and you do not have privilege. The device or its controller has been disabled by the system manager. The device is a keyboard line for pseudo keyboard use only.

?I/O CHANNEL NOT OPEN 9

An attempt was made to perform I/O on one of the twelve channels which has not been previously opened in the program.

?PROTECTION VIOLATION 10

You were prohibited from performing the requested operation because the kind of operation was illegal (such as input from a line printer) or because you did not have the privileges necessary (such as deleting a protected file).

?END OF FILE ON DEVICE 11

Attempt to perform input beyond the end of a data file; or a BASIC source file is called into memory and is found to contain no END statement.

?FATAL SYSTEM I/O FAILURE 12

An I/O error has occurred on the system level. You have no guarantee that the last operation has been performed. This error is caused by a hardware condition. Report such occurrences to the system manager. (See the discussion at the beginning of this appendix.)

?DATA ERROR ON DEVICE 13

One or more characters may have been transmitted incorrectly due to a parity error, bad punch combination on a card, or similar error.

?DEVICE HUNG OR WRITE LOCKED 14

Check hardware condition of device requested. Possible causes of this error include a line printer out of paper or high-speed reader being off-line.

?KEYBOARD WAIT EXHAUSTED 15

Time requested by WAIT statement has been exhausted with no input received from the specified keyboard.

?NAME OR ACCOUNT NOW EXISTS 16

An attempt was made to rename a file with the name of a file which already exists, or an attempt was made by the system manager to insert an account number which is already within the system.

?TOO MANY OPEN FILES ON UNIT 17

Only one open DECtape output file is permitted per DECtape drive. Only one open file per magnetic tape drive is permitted.

?ILLEGAL SYS( ) USAGE 18

Illegal use of the SYS system function.

?DISK BLOCK IS INTERLOCKED 19

The requested disk block segment is already in use (locked) by some other user.

?PACK IDS DON'T MATCH 20

The identification code for the specified disk pack does not match the identification code already on the pack.

?DISK PACK IS NOT MOUNTED 21

No disk pack is mounted on the specified disk drive.

?DISK PACK IS LOCKED OUT 22

The disk pack specified is mounted but temporarily disabled.

?ILLEGAL CLUSTER SIZE 23

The specified cluster size is unacceptable. The cluster size must be a power of 2. For a file cluster, the size must be equal to or greater than the pack cluster size and must not be greater than 256. For a pack cluster, the size must be equal to or greater than the device cluster size and must not be greater than 16. The device cluster size is fixed by type.

?DISK PACK IS PRIVATE 24
You do not have access to the specified private disk pack.

?DISK PACK NEEDS 'CLEANING' 25
Non-fatal disk mounting error; use the CLEAN operation in UTILTY or the ONLCLN (On Line Clean) program. (Both programs may be run only by privileged users.)

?FATAL DISK PACK MOUNT ERROR 26
Fatal disk mounting error. Disk cannot be successfully mounted.

?I/O TO DETACHED KEYBOARD 27
I/O was attempted to a hung up dataset or to the previous, but now detached, console keyboard for the job.

?PROGRAMMABLE ^C TRAP 28
A CTRL/C combination was typed while an ON ERROR GOTO statement was in effect and programmable CTRL/C trapping was enabled.

?CORRUPTED FILE STRUCTURE 29
Fatal error in CLEAN operation.

?DEVICE NOT FILE STRUCTURED 30
An attempt is made to access a device, other than a disk, DECtape, or magnetic tape device, as a file structured device. This error occurs, for example, when you attempt to gain a directory listing of a non-directory device.

?ILLEGAL BYTE COUNT FOR I/O 31
The buffer size specified in the RECORDSIZE option of the OPEN statement or in the COUNT option of the PUT statement is not a multiple of the block size of the device being used for I/O, or is illegal for the device. An attempt is made to run a compiled file which has improper size due to incorrect transfer procedure.

?NO BUFFER SPACE AVAILABLE 32
You access a file and the monitor requires one small buffer to complete the request but one is not currently available. If the program is sending messages, two conditions are possible. The first occurs when a program sends a message and the receiving program has exceeded the pending message limit. The second occurs when a sending program attempts to send a message and a small buffer is not available for the operation.

?ODD ADDRESS TRAP 33

This error occurs when an attempt is made to reference a nonexistent address, reference an odd address using a word instruction, or perform a PEEK function with an odd or nonexistent parameter.

?RESERVED INSTRUCTION TRAP 34

An attempt is made to execute an illegal or reserved instruction or an FPP instruction when floating-point hardware is not available. (See discussion at beginning of this appendix.)

?MEMORY MANAGEMENT VIOLATION 35

This hardware error occurs when an illegal Monitor address is specified using the PEEK function. Generation of the error message in situations other than using PEEK is cause for an SPR.

?SP STACK OVERFLOW 36

An attempt to extend the hardware stack beyond its legal size is encountered. (See discussion at beginning of this appendix.)

?DISK ERROR DURING SWAP 37

A hardware error occurs when your job is swapped into or out of memory. The contents of the job area are lost but the job remains logged into the system and is reinitialized to run the NONAME program. Report such occurrences to the system manager. (See discussion at beginning of this appendix.)

?MEMORY PARITY FAILURE 38

A parity error was detected in the memory occupied by this job. (See discussion at beginning of this appendix.)

?MAGTAPE SELECT ERROR 39

When access to a magnetic tape drive was attempted, the selected unit was found to be off line.

?MAGTAPE RECORD LENGTH ERROR 40

When performing input from magnetic tape, the record on tape was found to be longer than the buffer designated to handle the record.

?NON–RES RUN–TIME SYSTEM 41

The run-time system referenced has not been loaded into memory and is therefore nonresident.

?VIRTUAL BUFFER TOO LARGE 42

Virtual array buffers must be 512 bytes long.

?VIRTUAL ARRAY NOT ON DISK 43

A nondisk device is open on the channel upon which the virtual array is referenced.

?MATRIX OR ARRAY TOO BIG 44

In-memory array size is too large.

?VIRTUAL ARRAY NOT YET OPEN 45

An attempt was made to use a virtual array before opening the corresponding disk file.

?ILLEGAL I/O CHANNEL 46

An attempt was made to open a file on an I/O channel outside the range of the integer numbers 1 to 12.

?LINE TOO LONG 47

An attempt was made to input a line longer than 255 characters (which includes a line terminator). Buffer overflows.

%FLOATING POINT ERROR 48

An attempt was made to use a computed floating-point number outside the range 1E–38 <n <1E38 excluding zero. If no transfer to an error handling routine is made, zero is returned as the floating-point value. (C)

%ARGUMENT TOO LARGE IN EXP 49

Acceptable arguments are within the approximate range –89<arg< +88. The value returned is zero. (C)

%DATA FORMAT ERROR 50

A READ or INPUT statement detected data in an illegal format. For example, 1..2 is an improperly formed number, and 1.3 is an improperly formed integer, and X" is an illegal string. (C)

%INTEGER ERROR 51

An attempt was made to use a computed integer outside the range –32768<n<32767. For example, an attempt is made to assign to an integer variable a floating-point number outside the integer range. If no transfer to an error handling routine is made, zero is returned as the integer value. (C)

?ILLEGAL NUMBER 52

Integer overflow or underflow or floating-point overflow. The range for integers is –32768 to +32767; for floating-point numbers, the upper limit is 1E38. (For floating-point underflow, the ?FLOATING POINT ERROR (ERR=48) is generated.)

%ILLEGAL ARGUMENT IN LOG 53

Negative or zero argument to LOG function. Value returned is the argument as passed to the function. (C)

%IMAGINARY SQUARE ROOTS 54

An attempt was made to take square root of a number less than zero. The value returned is the square root of the absolute value of the argument. (C)

?SUBSCRIPT OUT OF RANGE 55

An attempt was made to reference an array element beyond the number of elements created for the array when it was dimensioned.

?CAN'T INVERT MATRIX 56

An attempt was made to invert a singular or nearly singular matrix.

?OUT OF DATA 57

The DATA list was exhausted and a READ requested additional data.

?ON STATEMENT OUT OF RANGE 58

The index value in an ON GOTO or ON GOSUB statement is less than one or greater than the number of line numbers in the list.

?NOT ENOUGH DATA IN RECORD 59

An INPUT statement did not find enough data in one line to satisfy all the specified variables.

?INTEGER OVERFLOW, FOR LOOP 60

The integer index in a FOR loop attempted to go beyond 32766 or below –32767.

%DIVISION BY 0 61

Your program attempted to divide some quantity by zero. If no transfer is made to an error handler routine, a 0 is returned as the result. (C)

?NO RUN–TIME SYSTEM 62

The run-time system referenced has not been added to the system list of run-time systems.

?FIELD OVERFLOWS BUFFER 63

An attempt was made to use FIELD to allocate more space than exists in the specified buffer.

?NOT A RANDOM ACCESS DEVICE 64

An attempt was made to perform random access I/O to a non-random access device.

?ILLEGAL MAGTAPE ( ) USAGE 65

Improper use of the MAGTAPE function.

?MISSING SPECIAL FEATURE 66

Your program employs a BASIC–PLUS feature not present on the given installation.

?ILLEGAL SWITCH USAGE 67

A CCL command contains an error in an otherwise valid CCL switch. (For example, the /SI:n switch was used without a value for n or a colon; or more than one of the same type of CCL switch was specified.) A file specification switch is not the last element in a file specification or is missing a colon or an argument.

A.2 Non-Recoverable Errors

Message and Meaning

?ARGUMENTS DON'T MATCH

Arguments in a function call do not match, in number or in type, the arguments defined for the function.

?BAD LINE NUMBER PAIR

Line numbers specified in a LIST or DELETE command were formatted incorrectly.

?BAD NUMBER IN PRINT–USING

Format specified in the PRINT–USING string cannot be used to print one or more values.

?CAN'T CONTINUE

Program was stopped or ended at a spot from which execution cannot be resumed.

?DATA TYPE ERROR

Incorrect usage of floating-point, integer, or character string format variable or constant where some other data type was necessary.

?DEF WITHOUT FNEND

A second DEF statement was encountered in the processing of a user function without an FNEND statement terminating the first user function definition.

?END OF STATEMENT NOT SEEN

Statement contains too many elements to be processed correctly.

?ERROR TEXT LOOKUP FAILURE

An I/O error occurred while the system was attempting to retrieve an error message. Possible cause could be the device containing the system error file (ERR.SYS) is off-line, or the system error file contains a bad block.

?EXECUTE ONLY FILE

Attempt was made to add, delete or list a statement in a compiled (.BAC) format file.

?EXPRESSION TOO COMPLICATED

This error usually occurs when parentheses have been nested too deeply. The depth allowable depends on the individual expression.

?FILE EXISTS–RENAME/REPLACE

A file of the name specified in a SAVE command already exists. To save the current program under the name specified, use REPLACE, or use RENAME followed by SAVE.

?FNEND WITHOUT DEF

An FNEND statement was encountered in your program without a previous function call having been executed.

?FNEND WITHOUT FUNCTION CALL

A FNEND statement was encountered in your program without a previous DEF statement being seen.

?FOR WITHOUT NEXT

A FOR statement was encountered in your program without a corresponding NEXT statement to terminate the loop.

?ILLEGAL CONDITIONAL CLAUSE

Incorrectly formatted conditional expression.

?ILLEGAL DEF NESTING

The range of one function definition crosses the range of another function definition.

?ILLEGAL DUMMY VARIABLE

One of the variables in the dummy variable list of a user-defined function is not a legal variable name.

?ILLEGAL EXPRESSION

Double operators, missing operators, mismatched parentheses, or some similar error has been found in an expression.

?ILLEGAL FIELD VARIABLE

The FIELD variable specified is unacceptable.

?ILLEGAL FN REDEFINITION

An attempt was made to redefine a user function.

?ILLEGAL FUNCTION NAME

An attempt was made to define a function with a function name not subscribing to the established format.

?ILLEGAL IF STATEMENT

Incorrectly formatted IF statement.

?ILLEGAL IN IMMEDIATE MODE

You issued a statement in immediate mode that can only be performed as part of a program.

?ILLEGAL LINE NUMBER(S)

Line number reference outside the range 1<n<32767.

?ILLEGAL MODE MIXING

String and numeric operations cannot be mixed.

?ILLEGAL STATEMENT

An attempt was made to execute a statement that did not compile without errors.

?ILLEGAL SYMBOL

An unrecognizable character was encountered. For example, a line consisting of a # character can cause this error.

?ILLEGAL VERB

The BASIC verb portion of the statement cannot be recognized.

%INCONSISTENT FUNCTION USAGE

A function is defined with a certain number of arguments but is elsewhere referenced with a different number of arguments. Fix the reference to match the definition and reload the program to reset the function definition.

%INCONSISTENT SUBSCRIPT USE

A subscripted variable is being used with a different number of dimensions from the number with which it was originally defined.

?LITERAL STRING NEEDED

A variable name was used where a numeric or character string was necessary.

?MATRIX DIMENSION ERROR

An attempt was made to dimension a matrix to more than two dimensions, or an error was made in the syntax of a DIM statement.

?MATRIX OR ARRAY WITHOUT DIM

A matrix or array element was referenced beyond the range of an implicitly dimensioned matrix.

?MAXIMUM MEMORY EXCEEDED

During an OLD operation, the job's private memory size maximum was reached. While running a program, the system required more memory for string or I/O buffer space and the job's private memory size maximum or the system maximum (16K words for BASIC–PLUS) was reached.

?MODIFIER ERROR

An attempt was made to use one of the statement modifiers (FOR, WHILE, UNTIL, IF, or UNLESS) incorrectly. An OPEN statement modifier, such as a RECORDSIZE, CLUSTERSIZE, FILESIZE, or MODE option, is not in the correct order.

?NEXT WITHOUT FOR

A NEXT statement was encountered in your program without a previous FOR statement having been seen.

?NO LOGINS

Message printed if the system is full and cannot accept additional users or if further logins are disabled by the system manager.

?NOT ENOUGH AVAILABLE MEMORY

An attempt was made to load a nonprivileged compiled program that is too large to run, given the job's private memory size maximum. The program must be made privileged to allow it to expand above a private memory size maximum; or the system manager must increase the job's private memory size maximum to accommodate the program.

?NUMBER IS NEEDED

A character string or variable name was used where a number was necessary.

?1 OR 2 DIMENSIONS ONLY

An attempt was made to dimension a matrix to more than two dimensions.

?ON STATEMENT NEEDS GOTO

A statement beginning with ON does not contain a GOTO or GOSUB clause.

?PLEASE SAY HELLO

Message printed by the LOGIN system program.

?PLEASE USE THE RUN COMMAND

A transfer of control (as in a GOTO, GOSUB or IF GOTO statement) cannot be performed from immediate mode.

?PRINT–USING BUFFER OVERFLOW

Format specified contains a field too large to be manipulated by the PRINT–USING statement.

?PRINT–USING FORMAT ERROR

An error was made in the construction of the string used to supply the output format in a PRINT–USING statement.

?PROGRAM LOST–SORRY

A fatal system error has occurred which caused your program to be lost. This error can indicate hardware problems or use of an improperly compiled program. Consult the system manager or the discussion of such errors in the *RSTS/E BASIC–PLUS Language Manual.*

?REDIMENSIONED ARRAY

Usage of an array or matrix within your program has caused BASIC–PLUS to redimension the array implicitly.

?RESUME AND NO ERROR

A RESUME statement was encountered where no error had occurred to cause a transfer into an error handling routine via the ON ERROR GOTO statement.

?RETURN WITHOUT GOSUB

RETURN statement encountered in your program without a previous GOSUB statement having been executed.

%SCALE FACTOR INTERLOCK

An attempt was made to execute a program or source statement with the current scale factor. The program runs but the system uses the scale factor of the program in memory rather than the current scale factor. Use REPLACE and OLD or recompile the program to run with the current scale factor. (C)

?STATEMENT NOT FOUND

Reference is made within the program to a line number that is not within the program.

STOP

STOP statement was executed. You can usually continue program execution by typing CONT and the RETURN key.

?STRING IS NEEDED

A number or variable name was used where a character string was necessary.

?SYNTAX ERROR

BASIC–PLUS statement was incorrectly formatted.

?TOO FEW ARGUMENTS

The function has been called with a number of arguments not equal to the number defined for the function.

?TOO MANY ARGUMENTS

A user-defined function may have up to five arguments.

?UNDEFINED FUNCTION CALLED

BASIC–PLUS interpreted some statement component as a function call for which there is no defined function (system or user).

?WHAT?

A command or immediate mode statement entered to BASIC–PLUS could not be processed. Illegal verb or improper format usually causes this error.

?WRONG MATH PACKAGE

Program was compiled on a system with either the 2–word or 4–word math package and an attempt is made to run the program on a system with the opposite math package. Recompile the program using the math package of the system on which it will be run.

Index

HOW TO ORDER ADDITIONAL DOCUMENTATION

DIRECT TELEPHONE ORDERS

In Continental USA
and Puerto Rico
call **800–258–1710**

In Canada
call **800–267–6146**

In New Hampshire,
Alaska or Hawaii
call **603–884–6660**

DIRECT MAIL ORDERS (U.S. and Puerto Rico*)

DIGITAL EQUIPMENT CORPORATION
P.O. Box CS2008
Nashua, New Hampshire 03061

DIRECT MAIL ORDERS (Canada)

DIGITAL EQUIPMENT OF CANADA LTD.
940 Belfast Road
Ottawa, Ontario, Canada K1G 4C2
Attn: A&SG Business Manager

INTERNATIONAL

DIGITAL EQUIPMENT CORPORATION
A&SG Business Manager
c/o Digital's local subsidary
or approved distributor

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Northboro, Massachusetts 01532

*Any prepaid order from Puerto Rico must be placed
with the Local Digital Subsidiary:
809–754–7575

RSTS/E
System User's Guide
AA-5133C-TC

Reader's Comments

Note: This form is for document comments only. Digital will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement. ____________________

Did you find errors in this manual? If so, specify the error and the page number. ____________________

Please indicate the type of user/reader that you most nearly represent.

- □ Assembly language programmer
- □ Higher-level language programmer
- □ Occasional programmer (experienced)
- □ User with little programming experience
- □ Student programmer
- □ Other (please specify) ____________________

Name ____________________ Date ____________________

Organization ____________________

Street ____________________

City ____________________ State ____________ Zip Code or Country ____________

--- Do Not Tear - Fold Here and Tape ---

digital

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN: Commercial Engineering Publications MK1-2/ H3
DIGITAL EQUIPMENT CORPORATION
CONTINENTAL BOULEVARD
MERRIMACK N.H. 03054

--- Do Not Tear - Fold Here and Tape ---

Cut Along Dotted Line