GRAPH/NET Computer Integrated Design System

Utility Program Guide

Utility Program Guide

Command Structure

The commands and utilities fit into functional groups as follows:

×INITIALIZATION COMMANDS

Bye
Dismount
Login
Path
SetSearch

×INFORMATIONAL COMMANDS

?
Details
DirTree
Help
Perq.Files
Statistics

×DEVICE MANAGEMENT COMMANDS

Partition
Scavenger

×FILE MANAGEMENT COMMANDS

Append   MakeDir
Copy   Print
Delete   Rename
Directory   TypeFile
Edit
FindString
Floppy

×PROGRAM DEVELOPMENT COMMANDS

Compile
Consolidate
Mount
ODTPRQ

×PROCESS CONTROL COMMANDS

Pause
ReRun
Run

Scrounge

×COMMUNICATIVE COMMANDS

Chatter
FTP

×SYSTEM MAINTENANCE COMMANDS

ExpandTabs
MakeBoot
Patch
SetBaud
SetTime
UserControl

Conventions (File System)

There are four methods to specify a filename. First, you can specify a full path name. For example:

    device:partition>[directory>]filename

Second, you can use the default boot device. For example:

    :partition>[directory>]filename

Using this syntax enables you to look for a file in a different partition.

Third, you can use the default device and partition that were set at boot time. For example:

    >[directory>]filename

The search then starts at the first directory you specify.

Fourth, you can just type:

    filename

and the search begins at the current directory (which you can set using the PATH command). This can involve any device, any partition, and any directory in that partition.

Setting the default path doesn't affect the default device and partition used in the first three methods.

In parallel with the concept of a current directory, the file system provides a search list. This provides you the ability to specify a set of directories to search, in a specific order, whenever you simply specify a filename. Refer to the SetSearch command description.

Conventions (File System)

Default File Extensions

An extension is a conventional sequence of characters that appears at the end of a filename. The operating system treats extensions simply as part of filenames. However, certain utilities interpret extensions in special ways. An extension is found by finding the last period (.) in the filename and taking the following symbols. Backup files conventionally have a "$" as the last character of their last extension; they take the form

    Name.Ext$

The following is a list of the standard extensions and how they are used.

.PAS - Pascal source files.

.FOR - FORTRAN source files.

.DAT - FORTRAN data files.

.SEG - The Pascal and FORTRAN compilers produce .SEG files when they compile a .PAS or .FOR file. The .SEG files are used as input to the Linker and contain the code that will be executed when the program is run.

.PSG - When the FORTRAN compiler cannot resolve calls to external routines at compile time, it produces pre-seg (.PSG) files. The .PSG files are used as input to the Consolidator, which produces .SEG files.

.RUN - Executable file produced by the Linker. Refers to the executable .SEG files that will actually be executed.

.MICRO - PERQ microcode source files have the extension .MICRO. They can be used as input to the micro assembler, PRQMIC.

.BIN - The runnable version of PERQ microcode is contained in .BIN files. These files are produced by the micro placer PRQPlace.

Conventions (File System)

.REL, .RSYM, .INT — These three file types are temporary files that are produced by the micro assembler and read by the placer. They can be deleted after a .BIN file has been created.

.DFS — These files are used to communicate definitions between programs that may be in different languages, for example, between Pascal and microcode.

.CMD — A number of programs on the PERQ accept commands from a file as well as from the keyboard. Files that contain the commands usually have this extension.

.DR — In the PERQ file system, directories are files. These files appear in a directory listing with the extension .DR.

.KST — Character set definitions are kept in files that have the extension .KST.

.MBOOT, .BOOT — When the PERQ is booted it reads the microcode instructions and Pascal operating system from files that have the extension .MBOOT and .BOOT respectively.

.ANIMATE, .CURSOR — Pictures that are put into the cursor that follow the puck. Various utilities use the pictures to depict progress or action.

.INDEX — An index to .HELP files.

.DOC — Formatted documentation.

.HELP — Files containing help about a subsystem.

.PARAS — An input file to HelpGen (HelpGen creates .INDEX and .HELP files).

Conventions (File System)

Wildcards

A number of utilities use a wildcard convention when looking up files.
You can use any number of wildcards in file specifications to
utilities that handle this convention. The following lists the
wildcard characters and describes their associated meanings:

        * matches 0 or more characters.
        & matches 1 or more characters.
        # matches exactly 1 character.
        '0 matches any digit.
        'A or 'a matches any alphabetic.
        '@ matches any non-alphanumeric.
        '* matches *. Other wild cards can be quoted also.

Examples of wildcard usage are:

        Dir *.Txt

        Dir Boot*

The first command provides a directory of all files with a .Txt
extension. The second command provides a directory of all files that
have one or more characters followed by the characters "boot",
followed by zero or more characters.

Introduction

The Shell commands are specified in the #ShellCommands section of the profile. Initially, you can use a copy of the file DEFAULT.PROFILE. Later, you may wish to define commands of your own. The format for each line in this section is:

⟨cmdline⟩ ⟨useDefault⟩ ⟨setDefault⟩ ⟨screensize⟩ ⟨cmdname⟩ ⟨comment⟩

Where ⟨cmdline⟩ is the command line Shell issues to execute the command. ⟨useDefault⟩ tells whether to use the default file name if no file name argument is specified for the command. ⟨setDefault⟩ specifies whether to set the default file name if an argument is provided. ⟨screensize⟩ can limit the screen to less than full size. ⟨cmdname⟩ is the name that will be used to invoke the command. This name and the rest of the line is printed when a "?" is typed so the additional comments are provided to explain the command.

The Shell and some other utility programs allow you to enter commands and arguments by means of PopUp Menus. (This only works, however, if you have booted from the harddisk.) When you press a puck button, a menu displays all the legal commands or arguments. Point to the desired command or argument and press a puck button to specify the selection. A menu is sometimes more convenient than typing out the name of the desired command or argument. Section 1.4 provides complete details on the use of PopUp menus.

COMMAND LINES

A command line consists of a command name, arguments, optional switches, and a line terminator.

Command Names

The command name describes the action the system performs or the name of the utility that performs the action. When entering a command that is in your profile, you need not type the entire command name; you can abbreviate the command name to the number of letters unique to other command names. If you do not have a user name and profile file, the Shell recognizes commands that are included in the file Default.Profile supplied with the system. These are the command names that you can abbreviate.

Utility Program Guide

Introduction

Input and Output Arguments

Input and output arguments further define the command action. The input and output arguments are usually file specifiers, as described in Chapter 2. Some commands use the default file as the argument if you do not supply one. Other commands require arguments as part of the command line.

If you neglect to supply an argument to a utility that uses the default file (for example, the Editor), the default file name is appended to your command. For example, if the default file is sys:user)myfile, typing the command:

    EDIT

is the same as typing:

    Edit sys:user)myfile

If a command requires an input and an output argument, you can specify the arguments as either

    INPUT~OUTPUT

        or

    INPUT OUTPUT

In the second form, there is at least one space, but multiple spaces are.

However, if a command accepts multiple input or output arguments, you must separate like arguments with commas (,) and distinguish input from output with the tilda character (~).
For example:

    input1,input2,...inputn ~ output1,output2,...outputn

If a command accepts multiple input arguments and no output arguments, you must separate the arguments with a comma (,).

If you neglect to supply a required argument, the utility prompts with a few words indicating the general nature of the missing argument.

Introduction

For example, the Rename command performs as follows:

    RENAME
    File to rename: OLDFILE
    Rename OLDFILE to: NEWFILE

The single line format for the above command is:

    RENAME OLDFILE ~ NEWFILE

            or

    RENAME OLDFILE NEWFILE

You can mix formats; the utility prompts for whatever you omit. For example:

    RENAME OLDFILE
    Rename OLDFILE to: NEWFILE

When a missing argument has a default, the program prints the default answer in square brackets ("[]"). You can choose this answer by pressing a return (thus providing an empty argument). To supply a different value, type the value followed by a carriage return. For example, if:

    Delete MyFile [No]:

is the prompt for an argument, a return means no. Of course, you could type "yes" or "no" as the argument. When there is no default for a prompt (as in the Rename example above), you must supply a response. However, your response can request help (type /HELP or press the HELP key). If you request HELP, the utility displays specific information and exits to the Shell.

Switches

Switches modify the action of the command and therefore must follow the command specification. An exception is the Help switch (either type /HELP or press the HELP key); when specified before the command, the Help switch supplies general information and when specified after a command, the Help switch provides specific information on the command. Switches always start with a slash (/) and are generally

optional. If a switch accepts a parameter, specify the parameter after the switch, but preceded by an equal sign (=). For example:

    /switch=parameter

The effect of a switch is global; regardless of where the switch appears on the command line, it has the same effect. A switch applies to every argument. If a command accepts multiple input or output arguments, no switch applies to one and not another argument. Some switches are mutually exclusive (for example, /ASK and /NOASK). If you specify a switch that conflicts with a previously specified switch, the last occurence has precedence. Likewise, if you change parameters by specifying a switch multiple times, only the last occurence has an effect.

All of the commands and utilities described in this manual accept the /HELP switch (either type /HELP or press the HELP key) to provide general information. Note that /HELP overrides all other switches; the utility displays specific help and then exits.

Line Terminator

Carriage return is the line terminator for all commands. You can also press the HELP key to terminate a command line. However, note that pressing the HELP key simply displays explanatory text without executing the command.

USER COMMAND FILES

The Shell and some utilities permit the use of command files. Rather than typing a command line to initiate and direct a utility, you can use a command file. A command file is a sequential file containing a list of utility specific commands. Instead of typing commonly used command sequences, you can type the sequence once and store it in a file. Specify the user command file in place of the command line(s) normally submitted to the utility.

To inititate user command files, replace the command line for a utility with a file specifier (as described in Chapter 2), preceded by an at sign (@). The utility requesting input then accesses the specified file and starts to read and respond to the commands

Introduction

contained within it. For example, to initiate a file, named MyFile.Cmd, of FLOPPY commands, type the following in response to the FLOPPY prompt:

    FLOPPY>@MyFile.Cmd

The FLOPPY utility accesses the file and then executes the floppy commands contained within the file MyFile.Cmd. The default file extension for user command files is .CMD. Thus the above command line could also be typed as follows:

    FLOPPY>@MyFile

You can nest user command files by simply specifying @file as a line within the user command file.

To include comments in a user command file, start the comment line with an exclamation mark (!). The exclamation mark can appear anywhere on a line in a user command file, but remember that the system ignores all characters after the exclamation mark.

POPUP MENUS

You can invoke a PopUp menu whenever the prompt (">") is displayed and no characters have been typed. (The prompt is actually a dark grey, right pointing triangle.) Press on the puck to invoke a PopUp menu. The menu appears at the current cursor position. Move the puck to guide the cursor up or down inside the menu and press any button to select a command. You can also press the HELP key for help on PopUp menus.

Reverse video highlights the selected command. A press over the selected command invokes the command just as if you typed the command on the keyboard.

If the menu is not large enough to hold all the commands, a scrolling mechanism is provided. When scrolling is necessary, a "gauge" is displayed in a black border at the bottom left of the menu. When you move the cursor over this area, the cursor changes to a scroll. A rsor to the right while
pressing causes the menu text to scroll up and moving the cursor to the left while pressing causes the text to scroll down. The further the cursor is moved from the original press position, the faster the text scrolls. A line in the gauge shows how fast the menu is scrolling. Of course, you cannot scroll past either end of the menu. (Each end is signified by a row of "~"). Releasing the button causes

Introduction

scrolling to stop.

If you press in an illegal part of the menu, or if you try to invoke a menu and have typed some text, the PERQ beeps. If a menu is displayed and you press outside of the menu, the menu disappears, causing no side effects. In addition, if you type CTRL/C or CTRL/SHIFT/C while a menu is displayed, the menu disappears.

In some applications, for example Delete, you can make multiple selections. The letter x in the lower right corner of the menu signals the end of a selection. Point at the x and press a puck button.

Notation Conventions

The command descriptions in this chapter are in alphabetical order regardless of command function. The command descriptions observe the following notational conventions:

o   Lowercase text indicates a variable whose actual value is determined when the command is entered

o   Square brackets ([]) indicate optional entries in a command line. Note that when an option is used, the brackets are not part of the syntax.

o   CTRL indicates the control key

o   CR indicates carriage return

o   SHIFT indicates the shift key

All numeric values required in a command are decimal values.

Each command description begins on a new page.

?

The question mark command lists all the commands implemnted in the
Shell. Additionally, the command provides some information on what
each command does and how it is called.

APPEND

Append copies one or more files to the end of an existing file. The command accepts a list of input files, separated by commas, and appends each file to the end of the first.

Format:

    APPEND [file][,file2][,...filen][/Help]

Append prompts for any missing arguments. The /Help switch displays a description of the Append command.

The Append command alters only the first file you specify. For example, the command:

    APPEND file1,file2

appends file2 to the end of file1, but file2 remains unchanged. The append operation is successive. For example, the command line:

    APPEND file1,file2,file3

first appends file2 to the end of file1 and then appends file3 to the end of file1.

BYE

The Bye command logs you off the PERQ system.

Format:

    BYE [/switch]

Valid switches are:

    Switch   Description
    ------   -----------

    /OFF   This switch logs you off the system and turns off the
                PERQ. Note that /OFF requires some special microcode,
                which should be available on your machine. If the
                microcode is not available, Bye /OFF simply logs you
                off and requests that you power down the machine by
                hand.   In this case, pressing CTRL/C causes Login to
                run.   If the /OFF switch is disabled on your PERQ, call
                Three Rivers Computer Corporation for details   on
                reenabling it.

    /WAIT   This switch logs you off the system and sends the hard
                disk heads to the center of the disk (the highest disk
                address).   Bye   then   prints   a   message and waits for
                manual power down.   (If you do not wish to power down,
                press CTRL/C twice or press CTRL/SHIFT/C once to cause
                Login to run). When shipping or moving a PERQ, always
                type BYE /WAIT to position the disk heads at the center
                of the disk.   This minimizes the risk of media damage.

    /HELP   This switch displays a description of the Bye command
                and the associated switches.   Note that the /HELP
                switch does not log you off.

If you do not specify a switch, Bye displays your name, the date, and
time of logoff.   Login then loads to permit you or some other
authorized user to log back on.

You must issue the Bye command before powering down the machine; Bye
deletes   certain   temporary disk files before logging off. Otherwise,
these files waste disk space. If you inadvertently power down the
machine without logging off, use the Scavenger program (see the
Scavenger command description) to reclaim the space.

CHATTER

The Chatter command allows a PERQ to act as a terminal, using an RS232 line for communication. The command supports either RS232 port (RSA or RSB) on a PERQ2.

Format:

    CHATTER

To issue a Chatter command, type CTRL/R. Chatter then displays a menu at the top of its window. Enter the command, in uppercase, followed by a <CR>. You must type CTRL/R before every Chatter command. The following describes the valid Chatter commands:

Command	Description
D	(Device) This command selects the RS232 port (RSA or RSB) to use with a PERQ2.
S	(Save) This command writes everything that comes from the remote computer in a file. The command prompts you for a file name. To access this file, issue a Chatter Close command before exiting the utility.
T	(Transmit) This command transmits a file across the RS232 line, as if it were being typed at the keyboard.
C	(Close) This command closes the file you opened with the Save command.
B	(Baud) This command changes the RS232 baud rate. Valid baud rates are: 110; 150; 300; 600; 1200; 2400; 4800; 9600; and 19200. The default is 4800 baud.
Q	(Quit) This command exits Chatter and returns to the Shell. The Quit command doesn't log off or disconnect the remote host.

While in Chatter, if the characters you type are not echoed properly, either the baud rate is incorrect or the RS232 cable is not properly connected.

Chatter cannot be used as a half duplex terminal since it does not echo characters locally.

PERQ Pascal Compiler

The PERQ Pascal compiler translates Pascal source code into a .Seg file that you can link and run. There are several ways to invoke the compiler and several options that you can use with it.

The system-supplied Default.Profile includes parameters so that when you specify the Compile command, the Shell automatically shrinks the screen and thus provides more memory for the compiler. The more memory available, the faster the compiler executes. You can also initiate the compiler with the Pascal.Run file, but only the Compile command is included in the system-supplied Default.Profile. Thus, if you initiate the compiler with the Pascal.Run file, you should first issue the ScreenSize command to provide maximum memory. (Of course, you could edit your profile to include Pascal and direct the Shell to shrink the screen for you.)

Format:

        COMPILE [inputfile] [~] [outputfile] [/switch(es)]

The inputfile is the name of the source file to compile. If the compiler does not find inputfile, it appends the extension .Pas and searches again. If inputfile is still not found, the compiler prompts for an entire command line. If you do not specify inputfile, the compiler uses the default file name remembered by the Shell.

The outputfile is the name of the file to contain the output of the compiler. The compiler appends the extension .Seg if it is not already present. Note that if you omit outputfile, the compiler uses the file name from inputfile. Then, if the .Pas extension is present, it is replaced with the .Seg extension. If the .Pas extension is not present, the compiler appends the .Seg extension. The compiler rewrites outputfile if it already exists.

You can specify any number of switches. Note that if you specify a switch multiple times, the last occurrence is used. If you specify the /HELP switch, the compiler ignores other information on the command line.

PERQ Pascal Compiler

Valid switches are:

Switch	Description

/AUTO

> The compiler automatically generates a RESET(INPUT) and REWRITE(OUTPUT). This switch enables automatic initialization. The default is /AUTO.

/NOAUTO

> This switch disables automatic initialization.

/COMMENT=<string>

> The /COMMENT switch permits the inclusion of arbitrary text in the first block of the .Seg file. This string has a maximum length of 80 characters. The switch is particularly useful for including copyright notices in .Seg files.

/ERRORFILE[=filename]

> When the compiler detects an error in a program, it displays error information (file, error number, and the last two lines where the error occurred) on the screen and then requests whether or not to continue. The /ERRORFILE switch overrides this action. When you specify the switch and the compiler detects an error, the error information is written to a file and there is no query of the user. However, the compiler does display the total number of errors encountered on the screen. The compiler appends the extension .Err if it is not already present. If you do not specify a filename, the compiler uses the source file name. If the .Pas extension is present, it is replaced with the .Err extension. If the .Pas extension is not present, the compiler appends the .Err extension.

PERQ Pascal Compiler

The error file exists after a compilation if and only if you specify the /ERRORFILE switch and an error is encountered. If the file filename.Err already exists from a previous compilation, it is rewritten, or deleted in the case of no compilation errors. This switch allows compilations to be left unattended.

/HELP

The /HELP switch provides general information and overrides all other switches.

/LIST[=filename]

The /LIST switch controls whether or not the compiler generates a program listing of the source text. With each source line, the compiler prints the line number, segment number, and procedure number. The default is to not generate a list file. The compiler appends the extension .Lst to filename if it is not already present. If you omit filename, the compiler uses the source file name. If the .Pas extension is present, it is replaced with the .Lst extension. If the .Pas extension is not present, the .Lst extension is appended.

/NAMES

The compiler generates a table of the procedure and function names at the end of the .Seg file. This table may be useful for debugging programs. This switch generates the table. The default is /NAMES.

/NONAMES

This switch disables generation of the table of procedure and function names.

/QUIET

This switch disables the Compiler from displaying the name of each procedure and function as it is compiled.

PERQ Pascal Compiler

/VERBOSE

This switch enables the Compiler to display the name of
each procedure and function as it is compiled. The
default is /VERBOSE.

/RANGE

This switch enables the generation of additional code
to perform checking on array subscripts and assignments
to subrange types. The default is /RANGE.

/NORANGE

This switch disables the generation of additional code
to perform checking on array subscripts and assignments
to subrange types. Note that programs compiled with
range checking disabled run slightly faster, but
invalid indices go undetected. Therefore, until a
program is fully debugged, it is advisable to keep
range checking enabled.

/SYMBOL=number

This switch sets the number of symbol table swap blocks
(number) used by the Compiler. As the number of symbol
table swap blocks increases, compiler execution time
becomes shorter; however physical memory requirements
increase (and the Compiler may abort due to
insufficient memory).
The default number of symbol table blocks and the
maximum number of symbol table blocks are both
dependent on the size of memory. For systems with 256k
bytes of main memory, the default number of symbol
table blocks is 24 and the maximum number of symbol
table blocks is 32. Note that you can specify more
than 32 symbol table blocks with a 256k byte system,
but performance usually degrades considerably. For
systems with 512k or 1024k bytes of main memory, the
default number of symbol table blocks is 200 and the
maximum number of symbol table blocks is also 200.

PERQ Pascal Compiler

/VERSION=string

The /VERSION switch permits the inclusion of a version
string in the first block of the .Seg file. This
string has a maximum length of 80 characters.
Currently this string is not used by any other PERQ
software, however, it may be accessed by user programs
to identify .Seg files.

The version string is terminated by either the end of
the command line or the occurrence of a '/' character
(hence a '/' may not appear in the version string).

You can include certain switches in the source program text.

Examples of legitimate compiler calls include:

COMPILE Program.pas

COM ProgramX
     (Note that the .pas extension is implicit;
      if ProgramX does not exist, the compiler
      looks for ProgramX.PAS.)

COMP Program2~Program1
     (creates the output file Program1.SEG)

COM
     (compiles the default file)

COM /symbol=32
     (compiles the default file with 32 symbol
      table blocks)

If you want to compile a program immediately after editing it, you
need not specify its name since the Shell remembers the last file
edited, compiled, linked, or run.

COPY

The Copy command creates a new file identical to the specified source.

Format:

    COPY SourceFile[~]DestinationFile[/switch]

Copy prompts for missing arguments and accepts wildcards in file
specifications. If the names of SourceFile and DestinationFile are
identical, a new file is not created; if you want two files, use
nonidentical names.

You can copy across devices and partitions and you can specify the
non-file-structured devices CONSOLE: and RS:. If you copy from the
console, CTRL/Z writes to the output file and CTRL/C aborts the copy
operation. Note that you cannot include control characters in the
file when you copy from the console. If you copy a file to the RS232
interface, the interface is driven at 9600 baud by default. Use the
SetBaud command (see the SetBaud command description) to change the
baud rate.

Valid switches are:

    Switch   Description
    ------   -----------

    /ASK

                    When you specify a wildcard, this switch requests
                    verification for each file copied. /ASK is the default
                    when you use a wildcard.

    /NOASK

                    This switch overrides verfication requests for
                    individual files. /NOASK is the default when you do
                    not specify wildcards.

    /CONFIRM

                    This switch requests verification before overwriting an
                    existing file. /CONFIRM is the default.

in directory to be copied from

1) Copy file full name ~ destination file full name

COPY

/NOCONFIRM

This switch overrides requests for confirmation before overwriting an existing file. /NOCONFIRM also sets /NOASK.

/HELP

This switch displays a description of the Copy command and the associated switches, but does not copy files.

If the source contains wild cards, the destination must contain the same wild cards in the same order. When you specify a wild card, Copy matches all files in the directory with the source pattern. For each match, the part of the source file name that matches each wild card replaces the corresponding wild card in the destination. For example,

    COPY foo*.abc   anotherdir>*baz.rmn z

copies the input file "FOOZAP.ABCD" into a new file named "anotherdir>ZAPbaz.rmnDz".

Copy asks for verification of each file copied when you specify a wildcard. The /NOASK switch disables the verification request. Copy also requests confirmation before overwriting an existing file. The /NOCONFIRM switch overrides the confirmation request. (Note that the /NOCONFIRM switch implies the /NOASK switch.)

Wild cards are not valid in the directory part of the source file. However, Copy uses the search list to try to find the source. Note that this is different from the Rename and Delete commands, both of which look in only one directory.

When the source file name contains no wild cards, the destination file name can contain, at most, one occurance of the asterisk wild card (*). In this case, the non-directory part of the source replaces the asterisk in the destination. For example,

        COPY sys:Boot>newOS>myprog.Pas dir3>new.*

copies the file "sys:Boot>newOS>myprog.Pas" into a new file named

COPY

"dir3>new.myProg.Pas".   This   is   most useful when you want to copy a
file from one directory to another with the same name.   For example,

                    COPY dir1>prog.Pas *

copies prog.pas from the directory "dir1" into the current directory.

If there are no wild cards in the source, an attempt to   include   wild
card   characters   other   than a single asterisk (*) in the destination
name leads to problems later; the system treats these extra wild cards
as   simple literal characters.   Because a file name with wild cards in
it is hard to specify, Copy   requires   confirmation   before   creating
files with wild cards in the name.

If   an error occurs and wild cards were used, Copy asks whether or not
to continue processing the rest of the files   that   match   the   input,
regardless of switches you specify.

The Preliminary Debugger

The current operating system includes a simple debugger. When the
user types CTRL/SHIFT-D or an uncaught exception is discovered, a dump
of the user stack is shown. This has the form:

```
Control-shift-D dump
Debug at        108 in routine 7 in IO_PRIVA.
Called from   214 in routine RANDOMWI (8) in WIPEWIN.
Called from   395 in routine WIPEWIN (0) in WIPEWIN.
Called from   149 in routine 0 in LOADER.
Called from   222 in routine 1 in SYSTEM.
Called from   520 in routine 0 in SYSTEM.
```

First, the reason for taking the dump is shown. Next is a trace of
all the procedures on the stack. Each line shows the location in the
code, the routine that location is in, and the module which contains
that routine. The location is the QCode offset from the beginning of
the procedure. You can use QDis to try to associate this with the
corresponding place in the source. When the debugger can find the
procedure name, it is printed followed by the procedure number. At
other times, only the routine number will be printed. You can count
the procedure (and exception and function) headers in the module
source file to determine which procedure it is.

When counting procedures, start with one for programs and zero for
modules. The main body of a program is its procedure zero. Exported
and forward procedures are counted only once, where the name first
appears. Internal (nested) procedures are counted exactly like other
procedures.

Procedure names are always truncated to eight characters and converted
to all uppercase. To get the names, the debugger examines the .Seg
file for the module. If the .Seg file found is not the one that was
loaded, the procedure names will be wrong. However, the procedure
numbers will always be correct. For the procedure names of system
modules, the system run file is checked to find the .Seg file name.
This .Seg file is used to get the procedure names. If the system run
file or the system .Seg files are not accessible, the debugger will
not be able to print the procedure names for system routines.

After the dump is printed, you are asked if you want to debug. (If a
dump is printed due to a CTRL/SHIFT/C, debugging is not allowed). If
you answer No, the program continues as if called from CTRL/SHIFT/D.
Otherwise the program aborts and control returns to the Shell.

The Preliminary Debugger

If you decide to debug, the debugger prints something like:

Scrounge, V0.10

   Now at routine KEYINTR (7) in IO_PRIVA
   There are 5 local words, 0 argument words, and 0 result words

Debug>

Now you can use the debugger commands to investigate your program. Notice that the debugger goes to more effort to find the procedure names than the original dump. Thus, if a procedure name was not printed at first, entering the debugger may display the name.

The debugger does not know the types or sizes of variables, but it does know the number of words allocated for locals, arguments and results. Note that the compiler may generate temporary variables which are included in the local count.

When you invoke a debugger command that takes an offset, you can type a number which is the offset of the word to print. Zero is the first word. If you type -1, the debugger prints all the words in the current context. For example, to an "a" command, the debugger prints all the arguments for -1. If you type -2, the debugger requests the first and the last offsets to print. In this manner, you can print a range of values. No checking is done to make sure that the offsets are in range; if a number is out of range, the debugger prints some random data.

The debugger prints data in the form:

    [7] (5053^) = 6

where 7 is the offset in the current procedure, 5053 is the offset from the bottom of the entire stack and 6 is the value in that location.

When a list of variables are defined in the same statement, they are allocated in reverse order. For example, if your procedure were defined with the following variables:

    var a,b,c: Integer;
        d: Char;

The Preliminary Debugger

the first word is the variable "c". The second word is "b", the third "a" and the fourth is "d". This is true for local and global variables and for records. Note that if you had declared the variables as:

```
var a: Integer;
    b: Integer;
    c: Integer;
    d: Char;
```

then "a" would be the first word, "b" is the second one, and so on. However, this does not hold for procedure parameters where the variables are always stored in exactly the order declared.

The debugger's commands are:

Command	Description

x   Set the radix.

        All integers are normally printed in signed decimal.
        With this command, you can specify any radix from 2 to
        36. If the radix is negative then all output will be
        unsigned. If it is positive then output will be
        signed. Note that this radix is only for output; all
        input is still in signed decimal.

>   Up level.

        Move up one level in the stack towards the top of the
        stack. To investigate the variables of a procedure,
        move up or down the stack until that procedure is
        reached and then it can be investigated.

<   Down level.

        Move down one level towards the bottom of the stack.
        When entering the debugger, the current procedure is
        set at the top so this command has to be used first.

.   Dereference.

        Dereference any pointer in memory. This takes a

The Preliminary Debugger

           segment number and the first and last offsets. It prints the memory locations from the first to last offsets (inclusive).

t   Top of Stack.

        Move to the top of the stack.

b   Bottom of Stack.

        Move to the bottom of the stack.

c   Current.

        Shows number of words for arguments, returns and locals for the current procedure.

d   Display Stack.

        This command reprints the original dump. Some additional procedure names may be printed. In addition, the procedure where you are debugging is marked with "<**>".

l   Local.

        Examine the local data. The debugger reprints the number of local words. You can type the offset of the word you want to see.

a   Argument.

        Examine the arguments to a procedure or function.

e   Exception.

        Examine the arguments (parameters) to an exception.

r   Returns.

        Examine the return values from a procedure.

The Preliminary Debugger

g   Globals.

Examine the globals for the module or program that the
procedure is in. When the g command is given, the
module or program name is printed. If it is a program,
the debugger asks if you want to skip input and output.
These are the two file variables that are defined for
every program and take up space at the top.   If you
answer yes to this question, you do not have to worry
about the space for them when counting variables.
Unfortunately, there is no way to examine data in
modules that do not have procedures on the stack.

m   Mode.

The debugger cannot know the type of data, but if you
know, you can tell the debugger. The mode command lets
you specify the output mode for data.   When you type
the Mode command, it prints the current mode and asks
for a new one. If you type "?" at this point, a list
of the options is printed.   They are: i=integer,
s=string, c=char, B=Boolean, b=byte. Notice the case
sensitivity of the arguments. When the mode is string,
the debugger cannot print a range or all data since it
cannot know how much memory was allocated to hold the
first string. In this case, if the -1 argument is
given, the offset is assumed to be zero. When printing
strings, the length is printed first.   When printing
bytes, the radix specified still holds (although they
will always be unsigned). For bytes and characters,
offsets are still in terms of words; the debugger
prints both bytes in the word specified.

s   Stack.

This command permits display of words anywhere on the
program stack.   Detailed knowledge of the compiler's
memory allocation is necessary to utilize this command
so it is generally not useful.

q   Quit.

Exits the debugger and aborts the program that was
running. This returns control back to the Shell.   It
requires confirmation.

The Preliminary Debugger

    p   Proceed.

            Exits the debugger and resumes the program executing.
            Note that this command allows you to resume from
            uncaught exceptions, but this is not recommended. In
            this case, confirmation is required. If the debugger
            was entered through CTRL/SHIFT-D, then no confirmation
            is required.

If an exception is raised inside the debugger, the debugger aborts
immediately and exits to the Shell. Also, in the debugger, CTRL/C and
CTRL/SHIFT-C both cause immediate exit to the Shell. CTRL/SHIFT-D is
disabled while inside the debugger.

Note that the HELP key does not work while in the debugger; use the
debugger question mark (?) command.

DELETE

The Delete command deletes the name of one or more files from the directory and places the blocks the file occupied on the free list, thus making those blocks available for use by other files. When you delete a file, you irrevocably destroy it.

Format:

        DELETE [filespecification][,file2][...,filen][/switch]

You can specify wildcards in the file specification, but not in a directory specification (refer to Chapter 2 for details on wildcard conventions).

Delete also allows you to select the files you want to delete by using a PopUp menu (see Chapter 1). To use a PopUp menu, type Delete followed by a carriage return (no arguments). Delete then prompts with

        File to delete or press for Menu:

You can simply press the pen or puck for a menu. You can also type a file name followed by switches. For example, you might type

        :boot>myDir>*.TMP/confirm

and then press the pen or puck. Do not type a carriage return before pressing.

When you press, Delete displays a menu of the files that match the file pattern. If you did not specify a file pattern, Delete lists all the files in the current directory. Simply press on the file name to select files in the menu that should be deleted. Note that unlike the menu for the Shell, the Delete menu permits you to select multiple files. Delete marks selected files with reverse-video. These are the files that will be deleted. You can de-select a selected file by simply pressing on it again. Since the number of files that can be displayed is limited, scrolling is provided when the the number of files matching the file specification is large. Use of the scrolling feature is described in Chapter 1.

Once you have selected all the files you wish to delete, move the cursor to the lower right corner of the menu to the spot with the "x" in it. The cursor changes to a large exclamation point. When you press here, all the selected files are deleted. If, however, you press outside of the menu before pressing here, no deletes will take place and Delete aborts.

DELETE

The valid switches are:

Switch	Description

/CONFIRM

This switch requests verification before deleting a file. /CONFIRM is the default when you use a wildcard unless you use a PopUp menu. When you use a PopUp menu, the default is /NOCONFIRM; all selected files are deleted when you press the exclamation point. For added safety, you can specify the /CONFIRM switch to request confirmation prior to deleting the selected files.

/NOCONFIRM

This switch overrides requests for confirmation. /NOCONFIRM is the default when you specify only a filename without a wildcard.

/HELP

This switch displays a description of the Delete command and the associated switches. Note that /HELP does not delete any files.

COMMAND DESCRIPTIONS

DETAILS

The Details command provides information about the state of a PERQ.

Format:

        DETAILS [command][,command][/switch][~outputfile]

You can include multiple commands on a single line, but you must separate each command with a comma (,). Also, note that unless you specify a command, the tilde character (~) is required with an output file.

The Details command provides information based on the commands and/or switches you specify. Each switch provides specific information while each command combines information from multiple switches. Valid commands are:

        Command   Description
        -------   -----------

        ALL   Displays all information.

        USER   Displays username and status.

        FILE   Displays file system status.

        SYSTEM   Displays operating system environment.

        DISK   Displays information on the disks.

        IO   Displays IO system information.

        SWITCHES   Displays information on the available Details switches.

Valid switches are:

        Switch   Description
        ------   -----------

        /USERNAME

                Displays the current user name.

COMMAND DESCRIPTIONS

DETAILS

/MEMORYSIZE

Displays the size of memory.

/PROFILENAME

Displays the name of the current profile file.

/PARTITION

Displays the names of all known devices and partitions.
The display includes the number of free blocks in each
partition.

/SEARCH

Displays the current search list.

/SHELLNAME

Displays the name of the current Shell runfile.

/SYSTEMVERSION

Displays the current operating system version and
hardware version.

/CONFIRM

If you direct the output of the command to a file, this
switch requests confirmation before overwriting an
existing file. /CONFIRM is the default.

/NOCONFIRM

This switch overrides confirmation requests; Details
output to a file overwrites an existing file of the
same name.

/TIME

Displays the current date and time.

COMMAND DESCRIPTIONS

DETAILS

/PATH

Displays the current path, default partition name and default device name.

/LASTFILE

Displays the default file for Edit and Compile.

/BOOTCHAR

Displays the boot character for the current system.

/BOOTS

Displays all valid boot characters.

/SWAP

Indicates whether or not swapping is enabled and, if so, to which partition.

/IOERRORS

Displays a count of how many times each of the IO errors occurred since the last boot.

/KEYBOARD

Displays keyboard and monitor information.

/ETHERNET

Displays Ethernet address and node name.

/RS232STATUS

Displays available RS232 hardware.

/POINTALLOWED

Indicates whether or not pointing is allowed.

COMMAND DESCRIPTIONS

DETAILS

/ALL

Displays all of the information.

/HELP

Displays an explanation of Details and its switches.

You can abbreviate a command or switch to as few characters as are unique.

If you do not specify a command or a switch, the defaults are /USERNAME, /PATH, /SEARCH, /CONFIRM, /SWAP, /BOOTCHAR, /BOOTS, /PARTITION, and /IOERRORS.

COMMAND DESCRIPTIONS

DIRECTORY

The Directory command provides a list of files in a directory. By default, the command provides an alphabetical list of the file names, but you can specify other sort algorithms. You can display the directory listing at your terminal or you can direct the listing to a file.

Format:

        DIRECTORY [dirSpec>][fileSpec][/switch]["][outputfile]

If invoked without a switch, the command lists, in alphabetical order, all files in the current directory.

To write the output of a Directory listing to a file, specify an output filename.

If you specify a wild card, Directory matches the dirSpec part of the command line against all directories and the fileSpec part against all files in the directories that matched dirSpec. Wild cards are described in Chapter 2. Note that wildcards are not valid in partition or device names.

The valid switches are:

        Switch   Description
        ------   -----------

        /ALL

                        This switch provides the following information for each file:

                                Number of Blocks
                                Number of Bits
                                Kind of file
                                Creation date
                                Last Update date
                                Last Access date

COMMAND DESCRIPTIONS

DIRECTORY

/DELIMITER

This switch lists the file names as

name : name

The switch is most useful when used in conjunction with an output file specification (for example, when you create a command file from the Directory listing).

/FAST

This switch lists only the file names, a short directory. /FAST is the default.

/HELP

This switch displays a description of the Directory command and the associated switches. Note that /HELP does not provide a directory listing.

/LISTDIRECTORIES

When performing a multi-directory operation, Directory lists only the directories that have valid matches for the fileSpec. This switch instructs Directory to list all directories that match the dirSpec, regardless of whether or not they contain matches for fileSpec.

/MULTICOLUMN

This switch instructs Directory to list files in four columns. /MULTICOLUMN is the default when Directory displays a short (/FAST) listing on the screen. Note that /ALL and /SIZE override this switch.

/ONECOLUMN

This switch lists all files in a single column. /ONECOLUMN is the default when you specify an output file.

COMMAND DESCRIPTIONS

DIRECTORY

/PARTITIONS

This switch provides partition information (for each partition) after the directory listing.

/SIZE

This switch displays the number of blocks and bits for each file in the directory listing.

/SORT=option

This switch permits you to specify the algorithm Directory uses to sort and list the file names. The options are:

NOSORT - lists the files in essentially random order.

NAME - sorts by the name of the file and produces an alphabetical listing. NAME is the default.

SIZE - sorts by file size. This option lists files in decreasing order, with the largest file first.

CREATEDATE - sorts by creation date. This option lists the most recent file first.

ACCESSDATE - sorts by last access. This option defines access as the last read operation performed on the file.

UPDATEDATE - sorts by last update. This option defines update as the last write operation performed on the file.

Examples:

DIR
lists every file in the current directory

DIR *>*
lists all files in all directories starting with the current directory and including all subdirectories.

COMMAND DESCRIPTIONS

DIRECTORY

DIRECTORY :BOOT>x*>*.run~run.list
looks in the Boot partition for all the run files in
directories whose names start with "x" and writes all
of these names into the file "run.list".

DIRECTORY Program*
tells you what files beginning with
"Program" are in the current directory, for example,
Program.pas, Program.seg, Program.run.

DIRECT *zing*
lists all files with "zing" in their names.

DIR Program*/SIZE
lists files beginning with "Program" and
tells how much disk space each occupies.

COMMAND DESCRIPTIONS

DIRTREE

The DirTree command provides a graphic representation of the file
system tree structure. The command erases the screen and displays all
the directories starting from the root directory on the left.

Format:

DIRTREE [rootdirectory][/switch]

You can specify any directory as the root of the tree. The default is
the default device; DirTree displays all the partitions, all the
directories in each partition, all the subdirectories, and so on.
Lines connect each directory to its parent. If you specify a
directory, DirTree simply starts the search from that directory.

If the specified, or default, root directory contains the current
directory, DirTree highlights the current directory with reverse
video.

You can change the path (equivalent to issuing a Path command) by
moving the cursor to the desired directory and pressing.

If the directory structure is too deep to fit on the screen, DirTree
puts an asterisk (*) on the right of the parent. To see more,
reinvoke DirTree with this directory as the root.

The valid DirTree switches are:

Switch	Description
/BLOCKS	

          Counts and displays the number of blocks in use in each
directory. Note that the count does not include blocks
used for file headers (at least one such block exists
for each file) nor does the count include blocks used
to hold the directory itself. For each directory that
has subdirectories, DirTree displays the total number
of blocks in the parent directory followed by a tilde
(~), followed by the total of all blocks in the
parentdirectory and in all its subdirectories. For
example, imagine a directory, User, with two

COMMAND DESCRIPTIONS

DIRTREE

subdirectories, Oldsource and Newsource. In the User
directory, 300 blocks are used for files, in Oldsource
150 blocks are used, and in Newsource 45 are used. The
count for User would be:

```
User>
300~495
```

The /BLOCKS switch increases DirTree execution time
about fourfold.

/WAIT

Enables pressing to select a new path. (Default)

/NOWAIT

Disables pressing to select a new path (DirTree simply
displays the tree structure).

/HELP

Displays a description of the DirTree command and the
associated switches. Note that /HELP does not display
the tree structure.

To exit DirTree, type any character or press in an area that does not
contain a directory. If you type a character, DirTree exits and
submits the character to the Shell. Thus, you can type the next
command to exit DirTree.

COMMAND DESCRIPTIONS

Dismount

The Dismount command detaches devices from the file system. When you dismount a device, the device cannot be accessed by the file system (refer to the Mount command description for more details).

It is imperative that you dismount file system floppies before removing the floppy disk from the drive.

Format:

DISMOUNT device

where device is either H (to specify the hard disk) or F (to specify the floppy disk).

COMMAND DESCRIPTIONS

EDITOR

The Editor creates or alters text files on the PERQ.

This command description discusses three basic uses of the Editor: to create a new file; to update an existing file; and to read a file at leisure. Refer to the GRAPH/NET Editor User's Guide for complete details on Editor use and operation.

Format:

    EDITOR FileSpecification

        or

    EDITOR/replay

If you omit the switch, Edit assumes that you want to edit the default file name remembered by the Shell. The /REPLAY switch is useful when disaster occurs during an edit session; you specify the switch, redo the edit session, and stop just before the disaster. Refer to the GRAPH/NET Editor User's Guide for details.

The Editor performs extension completion on the file name you specify; if the file to edit is FOO.PAS, you only need type FOO. The extensions that the Editor knows about, in order tried, are: Pas, For, Micro, Cmd, and Dfs.

To create a new file, invoke the Editor with the name of the new file. The Editor clears the screen to give you a blank page. Type I to insert the text that you want to type in. When you're finished, press the INS key (upper lefthand corner of the keyboard). This is important; it's the only way to save what you typed. Next, type Q. The Editor again clears the screen and prompts with a list of alternatives. See the GRAPH/NET Editor Guide for details on these.

To make changes to an existing file, invoke the Editor with the name of an existing file. Read the PERQ Editor User's Guide to familiarize yourself with the available editing functions. If you find that you've made changes to a file that isn't yours or that you've done irreparable damage to one that is, don't panic - if, after typing Q you type E, the Editor ignores all of the changes you've made. If you'd like to save your changes but don't want to alter the source file, type W after Q to make a new file.

COMMAND DESCRIPTIONS

EDITOR

To read a file at your leisure, you can EDIT it, reading and scrolling at your own pace. To safeguard against your having made any accidental changes to the file, type E after Q.

COMMAND DESCRIPTIONS

EXPANDTABS

ExpandTabs simulates tabs in every 8th column by replacing tabs in the
input file with the correct number of spaces. ExpandTabs is used when
the input file was written for another system and put onto a PERQ,
which does not support tabs. Its command line takes the form:

EXPANDTABS SourceFile DestinationFile

Note that the ExpandTabs command does not accept the Help switch.

COMMAND DESCRIPTIONS

FINDSTRING

The FindString command searches through a number of files for a
particular string. The command operates in two modes: context; and
nocontext. In context mode, FindString prints leading and trailing
characters for each occurrence of the specified string. In nocontext
mode, FindString prints only the first occurrence of the specified
string and does not print leading or trailing characters. The default
mode prints leading and trailing characters for each occurrence. You
specify which mode with the /CONTEXT or /NOCONTEXT switches.

Format:

        FINDSTRING string,filelist[/switch][~outputfile]

The first argument to FindString is the string to search for. To
include a space, comma (,), or slash (/) in the search string, precede
the the character with a single quote ('). The next argument is the
file pattern to match files against. The remaining arguments are
optional. You can direct FindString to write the occurrence(s) to a
file by specifying an output file.

        Switch   Description
        ------   -----------

        /CASESENSITIVE

                This switch specifies that case is significant (for
                example, if you specify the switch and the string to
                search for is XYZ, FindString does not view XyZ as a
                match.

        /NOCASESENSITIVE

                This switch specifies that case is not significant;
                FindString ignores upper and lower case.
                /NOCASESENSITIVE is the default.

        /CONTEXT

                This switch directs FindString to list leading and
                trailing characters for each occurrence of the string.
                /CONTEXT is the default.

COMMAND DESCRIPTIONS

FINDSTRING

/NOCONTEXT

This switch directs FindString to list only the first occurrence of the string.

/HELP

This switch displays a description of the FindString command and the associated switches. Note that /HELP does not search for string occurences.

Example:

    FindString screen, :boot>os>*.pas~screen.users/nocontext

This command directs FindString to search all files with a .PAS extension in the OS directory of the BOOT partition for an occurrence of the string screen. FindString writes the output to the file "screen.users". By default, case is not significant; in the example above, Screen matches screen. You can force FindString to match case exactly by specifying the /CASESENSITIVE switch.

COMMAND DESCRIPTIONS

FLOPPY

The FLOPPY utility formats, tests, reads, and writes RT-11 format floppy disks. You can use FLOPPY to transfer files between the hard disk and the floppy disk. Note that the RT-11 file format restricts filenames to six characters with a three character extension. Thus, if you wish to copy a file from the hard disk to the floppy disk, Floppy limits the file name on the floppy to six characters and the extension to three characters (you can enter the full file name, but only the first six characters will name the file on the floppy). Valid characters are the upper and lower case alphabetics, the digits 0 through 9, the dollar sign ($), and the period (.); Floppy uses the RAD50 character set.

Format:

    FLOPPY [command][/switch(es)]

To execute a single FLOPPY command, enter a command on the command line.

To execute multiple FLOPPY functions, type FLOPPY and press return. In this case, the utility prompts with FLOPPY>. You can then enter commands or use a PopUp menu. All of the Floppy commands prompt for missing arguments.

You can also direct FLOPPY to access a command file containing a list of FLOPPY commands. Simply enter an at sign (@) followed by the file name that contains the FLOPPY commands (@file).

Some FLOPPY commands require confirmation. This confirmation comes from the keyboard even if a user command file is in use. If you use the FAST command (see below), no confirmation is required. The Zero and Format commands, however, require an explicit /switch to override the confirmation request.

While FLOPPY is processing a command, a "hand" cursor moves down the right margin of the screen. When it has reached the bottom, your operation is complete. (With small files, the cursor does not reach the bottom of the screen.)

FLOPPY only accepts wildcards for the DELETE and DIRECTORY commands (see the respective command descriptions). In addition, the only wild card available is the asterisk (*) and it can only be used by itself in either the filename or extension part of a file specification (or both).

COMMAND DESCRIPTIONS

FLOPPY

Valid FLOPPY commands and their switches are:

Command   Description
-------   -----------

COMPARE

This command takes a disk file and a floppy file (in that order) and ensures that all bytes are identical.

Format:

COMPARE [diskfile][~][floppyfile]

You can specify multiple disk files and multiple floppy files for the comparison. If you specify multiple arguments, you must separate like arguments by a comma (,) and delimit input from output arguments with the tilde character (~).

The COMPARE command prints a message for every block that contains a difference.

COMPRESS

This command moves files so that all the unused blocks are at the end of the floppy; the command coalesces free space on the floppy.

Format:

COMPRESS [/switch]

The command does not accept input or output arguments, but has a switch which turns verification on or off. The default is verify; if this switch is on, COMPRESS checks every transfer to assure there are no errors. To use COMPRESS in verify mode, you need not specify the switch, since /VERIFY is the default. The /NOVERIFY switch overrides the default.

Do not interrupt this command once it starts; interrupting the Compress command renders the floppy data unreadable..

COMMAND DESCRIPTIONS

FLOPPY

DELETE

This command deletes a file or multiple files from the floppy.

Format:

DELETE [file][...,filen][/switch]

To delete multiple files, you must separate the filenames with a comma (,). By default, the DELETE command requests confirmation before deleting a file. You can override this by specifying the /NOCONFIRM switch.

You can specify an asterisk (*) only in place of the file name and/or in place of the file extension. For example, *.txt, *.*, or myprogram.* are valid uses of the wildcard (you cannot specify my*.*).

DENSITY

This command displays whether the floppy is single or double density.

Format:

DENSITY

The command does not accept arguments nor switches.

DIRECTORY

This command lists the files contained on a floppy and optionally writes the directory listing to a file.

Format:

DIRECTORY [file][~][outputfile][/switch]

If you specify the DIRECTORY command with no arguments, it lists all the files contained on the floppy. If you specify an input argument, DIRECTORY lists all files that match the specified filename.

COMMAND DESCRIPTIONS

FLOPPY

By default, the DIRECTORY command prints a full listing. To override this and print only the file names, specify the /SHORT switch.

If you specify an output argument, DIRECTORY only writes the file names (/SHORT is the default when you specify an output file) to a disk file with that name.

Note that when you direct the output to a file, Directory does not display the filenames on the screen.

DUPLICATE

This command copies the contents of one floppy onto another formatted floppy.

Format:

DUPLICATE [/switch]

The command does not accept arguments.

Duplicate creates a set of scratch files on the hard disk, copies the scratch files back to the new floppy, and then, if there were no errors, deletes the scratch files from the hard disk. To retain the scratch files on the hard disk, specify the /NODELETE switch. By default, the Duplicate command creates a double-sided floppy. To override this and create a single-sided floppy, you must specify the /SINGLESIDED switch.

Remember that the blank floppy to be duplicated must have been formatted prior to invoking Duplicate.

FAST

This command turns off requests for confirmation for all subsequent commands except Format and Zero; the command sets the /NOASK and /NOCONFIRM switches for all subsequent commands except Format and Zero.

COMMAND DESCRIPTIONS

FLOPPY

Format:

FAST

The command does not accept arguments nor switches.

Use the Fast command only in conjunction with command files.

FLOPPYGET

This command copies the contents of a floppy disk to the hard disk. The command creates a set of scratch files on the hard disk, which can be copied to another floppy with the FloppyPut command. The scratch files are in a special format for use by Floppy.

Format:

FLOPPYGET [/SINGLESIDED]

The Floppyget command assumes you wish to copy a double-sided floppy to disk. Specify the /SINGLESIDED switch for a single-sided floppy.

FLOPPYPUT

This command copies the disk files created by Floppyget onto a floppy disk.

Format:

FLOPPYPUT [/switch]

You can specify the /DELETE switch to delete the disk files after copying them to the floppy (/NODELETE is the default). Note that the disk files are not deleted if an error occurs. For single-sided floppies, you must use the /SINGLESIDED switch (/DOUBLESIDED is the default).

COMMAND DESCRIPTIONS

FLOPPY

FORMAT

This command formats a floppy disk and destroys its current contents. Format command switches permit you to specify the floppy density, number of sides, and whether or not to test after formatting. The /Interleave switch permits you to specify the number of floppy sectors between sequentially numbered sectors. For example, if the /Interleave value is one, the sectors are numbered sequentially. If the /Interleave value is two, every other sector is numbered sequentially.

Format:

        FORMAT [/switch]

The valid switches are:

        /DblDensity
        /SingleDensity (default)
        /Noconfirm
        /Singlesided
        /DblSided (default)
        /Test
        /NoTest (default)
        /Interleave=value (default is 2)

GET

This command copies one or more floppy files to the hard disk.

Format:

        GET [floppyfile][~][hardfile][/switch]

In its simplest form, you specify only an input filename to copy from the floppy to the hard disk. GET then copies the floppy file to the hard disk using the same filename. If the filename already exists on the hard disk, GET requests confirmation before overwriting it. If you omit the input file name, GET prompts for

floppy format /dbldensity /Hov
floppy zero /nov

COMMAND DESCRIPTIONS

FLOPPY

the file to copy. Other forms of the command permit you to name the hard disk file. For example

        GET floppyfile~harddiskfile

To specify multiple files, use the following construct

        GET f1,f2,...fn~h1,h2,...hn

Like the COMPRESS command, GET takes the /VERIFY and /NOVERIFY switches. The default is /VERIFY.

The GET command requires confirmation before overwriting a file on the hard disk. You can specify the /NOCONFIRM switch to override this action.

HELP

When issued as a command, HELP provides general information for the FLOPPY utility. You can get specific help by using the /HELP switch. Note that HELP and /HELP override any other commands or switches; FLOPPY displays the requested help and then reprompts.

PATH

This command changes the current hard disk directory.

Format:

        Path [pathname]

Specify the directory for the new path. If you omit the directory, Path prompts for it.

PAUSE

This command suspends FLOPPY execution.

Format:
        PAUSE
Press carriage return to continue.

COMMAND DESCRIPTIONS

FLOPPY

PUT

This command parallels the GET command, but transfers a file or files from the hard disk to the floppy disk.

Format:

PUT [hardfile][~][floppyfile][/switch]

PUT also requires confirmation before overwriting an existing file on the floppy (override this with /NOCONFIRM) and, like GET, prompts for a missing file name and accepts the /VERIFY and /NOVERIFY switches.

QUIT

This command exits the FLOPPY utility.

Format:

QUIT

RENAME

This command changes the name of a floppy file.

Format:

RENAME [oldname][~][newname][/NOCONFIRM]

If the new filename already exists, RENAME asks for confirmation before overwriting it. You can override this with the /NOCONFIRM switch.

TYPE

This command displays a floppy file on the screen. The "hand" cursor tells how much of file has been displayed.

Format:

TYPE [filename][/NOWAIT]

COMMAND DESCRIPTIONS

FLOPPY

When the Type command finds a formfeed character (^L) in the file, it waits after displaying a screenful of text. This enables the user to read the page before the screen is erased and the next page displayed. To continue reading, type CTRL/Q. You can disable this wait feature by specifying the /NOWAIT switch. The Type command displays a solid, left pointing triangle when it reaches the end of the file.

ZERO

This command creates a new directory on the floppy.

Format:

ZERO [/switch]

By default, the directory matches the number of sides on the floppy. You can override this by specifying the /SINGLESIDED switch. The Zero command always requests confirmation before creating the directory. You can override this only by specifying the /NOCONFIRM switch. In the process of creating the new directory, the ZERO command destroys the contents of the current floppy. Use the ZERO command after formatting a floppy (see the FLOPPY FORMAT command description) or whenever you wish to destroy the current content of a floppy.

Note that when you issue the ZERO command, you irrevocably destroy the current contents.

COMMAND DESCRIPTIONS

FTP

The FTP utility copies files across either an RS232 link or an ETHRENET connection to another PERQ or to any other computer that supports the FTP protocol.

To use the RS232 line, the two machines must be connected by an RS232 cable, which plugs into their RS232 ports. On a PERQ, the cable plugs into port A. On a PERQ2, the cable can plug in either RS232 port (RSA or RSB).

To use an ETHERNET connection, first assign an Ethernet node name for each machine on the network in the file ETHERNET.NAMES. Each machine on the network must include the file ETHERNET.NAMES in the boot partition (for example, SYS:BOOT)). The file simply contains the machine name for the specific node. You can list up to ten names (aliases) for an individual node. If you wish to use an alias, list each name on a separate line in the ETHERNET.NAMES file. If two nodes use the same name, the first to respond makes the connection. Once the node names are established, you can issue the FTP Connect command and specify a remote node name. If the specified name exists in ETHERNET.NAMES for a machine on the network, that machine returns its ETHERNET address and the connection completes.

Once the connection or addresses are established, each machine must run FTP.

Format:

    FTP [commnad][/switch(es)]

FTP prompts with FTP>. You can enter commands directly to FTP or use the PopUp menu.

The following lists the FTP commands and their format. Valid commands are:

    Command   Description
    -------   -----------

    BAUD baudrate

            This command sets the baud rate for the RS232 line.
            Initially, the default is 9600 and then the default

COMMAND DESCRIPTIONS

FTP

QUIT   This command exits FTP.

REMOTEPREFIX string

> This command allows you to specify a character string
> to precede all remote file pathnames. The string you
> specify can be a simple character string or the string
> can be a path specification (see the LocalPrefix
> command).

TEXT   This command specifies that only seven bits of each
              byte are significant in a file transfer. BINARY
              transfers are the default. When transfering textual
              files between a PERQ and some other machine, use the
              Text method.

You can specify the following commands as switches:

    ASK
    NOASK
    BAUD=baudrate
    BINARY
    CONFIRM
    NOCONFIRM
    CONNECT
    DEVICE=devicetype
    LOCALPREFIX=string
    MODE=mode
    REMOTEPREFIX=string

Simply precede the command with a slash (/name). You can include
switches on the command line or you can include the FTP switches in a
FTP section of your profile.

When issued as a switch to another command, switches apply only to the
current command line. This permits you to issue one line commands.
For example,

    FTP GET file/Mode=PDP-11/BAUD=1200/DEVICE=RS232

Control is returned to the command interpreter Shell after that one
command is executed.

COMMAND DESCRIPTIONS

FTP

matches the baud rate you specify. Valid RS232 baud rates are: 300; 1200; 2400; 4800; 9600; and 19200.

BINARY   This command specifies that all eight bits of each byte are significant in a file transfer. BINARY transfers are the default. When transfering between two PERQs, always use the Binary method.

CONFIRM   This command requests confirmation before overwriting a file on the local node. Note that the confirmation request does not apply to the remote node; when you transfer a file to another node, the file overwrites an existing file with the same name. Confirm is the default.

NOCONFIRM This command overrides confirmation requests for files on the local node.

CONNECT nodename

This command establishes the Ethernet connection between the local node and the specified remote node (nodename).

DEVICE type

This command specifies whether the nodes are connected by Ethernet or the RS232 line. The valid entries for type are: Ethernet; ByteStreamEthernet; and RS232. Use Ethernet (the default type) for an Ethernet transfer between two PERQs, ByteStreamEthernet for an Ethernet transfer between a PERQ and some other machine, and RS232 for a transfer between machines connected by an RS232 link.

GET [SourceFile][~][DestinationFile]

This command specifies the name of a file to copy from a remote node to the local node. You can optionally include a file name for the file on the local node. By default, FTP requests verification before overwriting an existing local file (override this with the NoConfirm command). If you do not specify the name of the file to transfer (SourceFile), the command prompts for it.

COMMAND DESCRIPTIONS

FTP

HELP   This command displays information on FTP and its
            associated commands.

LOCALPREFIX string

            This command allows you to specify a character string
            to precede all local file pathnames. The string you
            specify can be a simple character string or the string
            can be a path specification. For example, if you
            specify Foo as the LocalPrefix string, all files
            transferred to your machine will begin with the prefix
            Foo. However, if you specify Foo> as the LocalPrefix
            string, the string specifies a directory; all files
            transferred to your machine are placed in directory
            Foo>. When specifying a directory, you must explicitly
            state the complete path unless the directory is a
            subdirectory of the current path.

MODE mode

            This command specifies machine type. Valid entries
            are: PERQ (default); PDP-11; and VAX.

POLL   This command permits a node to send or receive
            transfers at the request of a remote node. The other
            machine issues a Connect command specifying the polling
            machine as the node and then issues a Get or Put
            command to transfer files.

PUT [SourceFile][~][DestinationFile]

            This command specifies the name of a file to copy from
            the local node to a remote node. You can optionally
            include a file name for the file on the remote node.
            The file you copy from the local node overwrites an
            existing file on the remote node. If you do not
            specify the name of the file to transfer (SourceFile),
            the command prompts for it. Note that you can copy a
            directory file (file.DR) to a remote node; the Put
            command creates a file on the remote node that lists
            the names of all files in the directory from the local
            node.

COMMAND DESCRIPTIONS

FTP

When issued without a command, switches apply to all succeeding command lines (until you reset or override the switch value). Note that FTP reads the profile file and thus permits you to tailor switch settings to your situation.

You can transfer files with either machine taking the active role; the following describes the alternate scenarios:

1. PERQ #1 takes the active role and PERQ #2 is passive. They'll follow this script:

        PERQ #2: Runs FTP and starts polling
        PERQ #1: Runs FTP, connects to the polling node
                 for an Ethernet transfer, and then
                 issues the next command
        PERQ #1: PUT SourceFile[~][DestinationFile]
        Both PERQs: #!#!#!#!#!...#!#!
                 (this appears on the screen as
                 the file is being transferred
                 and a "hand cursor" moves from top
                 to bottom to show percentage
                 complete. When the hand cursor reaches
                 the bottom, the transfer
                 is complete.)

2. PERQ #2 has the active role:

        PERQ #1: Runs FTP and starts polling
        PERQ #2: Runs FTP, connects to the polling node
                 for an Ethernet transfer, and then
                 issues the next command
        PERQ #2: GET DestinationFile[~][SourceFile]
        Both PERQs: #!#!#!#!...#!
                 (same as above)

The passive PERQ polls while the active machine processes the transfer. Note that for an Ethernet transfer, the active machine must issue a Connect command.

If an FTP transfer fails, the current transfer aborts and FTP waits for another command (or exits if invoked with a command line). However, if FTP was invoked from a command file and the transfer

COMMAND DESCRIPTIONS

FTP

fails, it restarts the transfer and repeatedly tries again until it succeeds. FTP pauses command file execution if it detects a syntax error in the command file and asks whether or not you wish to continue.

COMMAND DESCRIPTIONS

HELP

The Help command displays information about other commands and utilities and the PERQ operating system.

Format:

HELP [command]

If you do not specify an argument, Help displays a message describing the PERQ and the use of PopUp menus. If you specify an argument, Help displays information describing the specified command or utility.

If the Help utility is available, pressing the HELP key or typing the Help command without an argument gets you to the Help utility. The Help utility erases the screen and then prints a list of all commands for which help exists. Type one of these names (or use the PopUp menu to specify one). Help then prints specific information for that command.

COMMAND DESCRIPTIONS

LINKER

The Linker produces a runfile, with a .Run extension from compiler-generated .Seg files. The runfile is created by linking together all of the separately compiled modules that make up a program.

Format:

    LINK Source[,Source1][~runfile][/switch(es)]

Examples of valid command lines include

        LINK Program
        where Program.Seg is the output of a compilation

        LINK Program1, Program2
        where Program1.seg and Program2.seg are compiler
        output files. The output will be Program1.Run.

        LINK Program1, Program2~Program0/VERBOSE
        where Program1.Seg and Program2.Seg are com-
        piled .Seg files and Program0.run is to be the
        runfile. The VERBOSE switch specifies that the
        name of each import module is to be listed.

Any number of source files can be listed separated by commas. The following describes the valid Linker switches:

    Switch   Description
    ------   -----------

    /MAP=name

                This switch creates a map file. If you don't specify a
                name after the equal sign, the MAP file will have the
                same name as the runfile with a .Map extension. A map
                file contains a listing showing placement and size of
                the code and data segments of the program as well as
                other linkage details. Thus, map files are useful in
                debugging user programs.

COMMAND DESCRIPTIONS

LINKER

/STACKSIZE=blocks

This switch specifies the initial size of the memory stack. The size is in blocks and the default is 16 blocks. Normally, you need not specify the memory stack size, but if your program makes numerous procedure calls, you can increase the memory stack size with this switch.

/STACKINCR=blocks

This switch specifies the number of blocks to add to the memory stack when it fills. The increment is in blocks and the default is 4 blocks.

/HEAPSIZE=blocks

This switch specifies the size of the free memory pool. The size is in blocks and the default is 4 blocks.

/HEAPINCR=blocks

This switch specifies the number of blocks to add to the free memory pool when you exhaust space. The increment is in blocks and the default is 4 blocks.

/USER

This switch specifies that the program being linked is a user program. /USER is the default. (The opposite is /SYSTEM.)

/VERBOSE

This switch lists the imports of each module as the module is being processed.

/SYSTEM

This switch specifies that this is a system program.

COMMAND DESCRIPTIONS

LINKER

/VERSION=nn

This switch links the program with the version of the system specified by "nn". If the run file name ends in a number (for example, "LOGIN.42"), then the Linker uses the number as the version number. This can be overridden by using /VERSION=nn switch.

/HELP

This switch displays a description of the Link command and the associated switches. Note that the /HELP switch does not log you onto the system.

Every system has a version number. When you create a new system, you also should use a new system version number. You then have to relink all the runfiles. This insures that the programs execute correctly with the new system. In order to prevent confusion, critical system run files (Link, Shell, Login, and System) include the system version number in their name. For example, the .Run file for Login is Login.6.Run. When you link a file, it looks up the system run file based on the version in use. If no version is specified, it uses the current system version (which is displayed at the top of the screen when in the Shell). If a version is specified, a system run file with that number is used to resolve references to system routines. If the /SYSTEM switch is used, however, no run file is looked for since a new system run file is created. The syntax for creating a system run file is:

LINK SYSTEM~SYSTEM.NewNumber/SYSTEM/VERSION=NewNumber

where the /VERSION switch is optional.

To create runfiles for a new system, you link using the VERSION switch after making the system runfile. It's done like this:

LINK FileSpecification/VERSION=NewNumber

COMMAND DESCRIPTIONS

LOGIN

The Login command initiates a session at a PERQ.

Format:

LOGIN [UserName][/switch(es)]

Login searches the System.Users file for UserName and validates the password. The UserControl program (see the UserControl command description) maintains the System.Users file. If you omit a UserName, Login searches System.Users for the name guest. Note that initially the Guest account does not have password protection. To help prevent unauthorized access, you should assign a password to the Guest account (see the UserControl command description).

If Login encounters an error with the System.Users file (for example, a file read error), Login bypasses account validation and initiates the session.

The boot code as well as the Bye command automatically call Login. You can initiate a session manually by specifying a valid user name and password or you can initiate a session automatically by creating a special file for the main PERQ user. Note that automatic Login requires you to create a file and thus cannot be used the first time you use your PERQ.

At boot time, Login searches the boot partition for a file named Principal.User. This file, which must be in the boot partition, contains as its first line the UserName of the main user of the machine. Login then searches the System.Users file for the UserName and validates the password. If the file Principal.User does not exist in the boot partition or if the UserName specified in the file does not match a UserName in System.Users, Login requests your name, and password.

When initiated following a Bye command, Login simply requests your name and password.

Following account validation, Login checks for a user profile file (the UserControl command allows you to specify a path for your profile). If the user has a profile, Login reads the file to find parameters to tailor the PERQ for the individual user.

COMMAND DESCRIPTIONS

LOGIN

You can include switches on the command line or you can include the Login switches in a Login section of your profile.

The following describes the valid Login switches:

Switch   Description
------   -----------

/PATH=pathname

Sets the default path to pathname. The /PATH switch is not cumulative; if you specify the switch multiple times, only the last one has an effect.

/SETSEARCH=pathname

Pushes pathname onto the search list. The effect of this switch is cumulative; if you specify the switch multiple times, you add additional items to the search list. If the argument is a minus sign (-), the last path is popped from the list. Note that the last item specified is the head of the searchlist and the first item specified is the end of the list.

/SHELL=filename

Specifies an alternate command interpreter to run instead of the Shell. This is an aid in debugging a new or user written command line interpreter.

/SCREENBOTTOM=option

Sets the default parameters for the bottom of the screen when you change the screen size with a ScreenSize command (see the ScreenSize command description).

Valid options are:

ON - displays bits stored in area
OFF - bottom is all one color
WHITE - bottom matches background
BLACK - bottom is opposite of background

COMMAND DESCRIPTIONS

LOGIN

/POINTALLOWED=option

Specifies whether or not the system accepts input from a pointing device. The valid options are TRUE and FALSE. TRUE allows use of a pointing device and implies the use of PopUp menus. FALSE prevents all uses of a pointing device. /POINTALLOWED=TRUE is the default.

/CURSORFUNCTION=n

Sets the default cursor function. The cursor function also determines the screen color. Thus, you can use this switch to set the default screen color.

Valid cursor functions (values for n) are the integers 0 through 7 inclusive. The integers signify the following:

0 - screen is all white
1 - only cursor displays
2 - white on black, cursor is large square
3 - black on white, cursor is large square
4 - black on white, black cursor hides image
5 - white on black, white cursor hides image
6 - black on white, cursor inverts
7 - white on black, cursor inverts

If you prefer white letters on a black background, use CURSORFUNCTION=5.

The default value is 4 (black letters on a white background).

/COMMAND=string

Execute the specified string as the first Shell command after login and before accepting commands from the keyboard.

/PROFILE=filename

Specifies a file (filename) to execute as the profile file.

COMMAND DESCRIPTIONS

LOGIN

/TABLETTYPE=type

When you boot a PERQ and a tablet is connected, the tablet is used as the pointing device. If a tablet is not connected, the system assumes that the bitpad is the pointing device. Thus, if both a tablet and a bitpad are connected, only the tablet is recognized. This switch permits you to specify whether to use the tablet or bitpad.

Valid entries for type are Tablet and Bitpad.

/REALRELATIVE=option

Specifies whether the pointing device (tablet or bitpad) is read in absolute or relative mode. In absolute mode, cursor position on the screen is determined by the actual (absolute) tablet coordinates of the puck. In relative mode, cursor position is determined by the difference between previous and present tablet coordinates.

For example, assume that the puck is at the lower left corner of the tablet. In both modes the cursor follows the puck as it moves across the tablet. However, if you lift the puck and place it at the upper right corner of the tablet, the cursor moves to the upper right corner of the screen in absolute mode, but remains in the lower left corner in relative mode. The next relative cursor position is mapped from the lower left.

Valid options are True, to specify relative mode, and False, to specify absolute mode.

/HELP

This switch displays a description of the Login command and the associated switches. Note that the /HELP switch does not produce a run file.

COMMAND DESCRIPTIONS

MAKEBOOT

MakeBoot writes boot files onto hard disk and floppy disk.

Format:

MAKEBOOT [RunFile]

MakeBoot prompts for all parameters.

COMMAND DESCRIPTIONS

MAKEDIR

The MakeDir command creates a new, empty directory.

Format:

    MAKEDIR [FileSpecification][/Help]

where FileSpecification is the name for the new directory. If you do
not specify the name, MakeDir prompts for one. All directories must
have a .DR extension. Therefore, MakeDir automatically appends .DR if
you do not specify it. The /Help switch displays a description of the
MakeDir command.

MakeDir does not accept the name ROOT or the name of any existing file
that has a .DR extension; it prints an error message if you supply
such a name.

COMMAND DESCRIPTIONS

MOUNT

The Mount command attaches devices to the file system. Devices must be mounted before files on them can be accessed by the file system.

Format:

    MOUNT device

where device is H or F.

It is imperative that you dismount file system (as opposed to FLOPPY) floppies before you remove them from the floppy drive. If you fail to do this, the next floppy put into the drive may be overwritten and the file system floppy is also likely to be corrupted.

Floppies written with the FLOPPY program cannot be mounted or dismounted.

A device can be mounted more than once.

Note that if you dismount the device you're running from the system will not be able to find the Shell.

COMMAND DESCRIPTIONS

PARTITION

The Partition command creates new partitions or modifies the names and sizes of existing partitions on a device.

Format:

Partition

Partition prompts for all parameters.

COMMAND DESCRIPTIONS

PATH

The Path command changes the current directory, but does not affect the search list. Your "path" is the directory you are currently using. To find named files, the system first looks in this directory and then in directories specified in your search list (see the SetSearch command description). New files are created in this directory if a different one is not specified.

Format:

    PATH [pathname]

If you do not specify a pathname, Path displays the current path. You can then type a new path or press carriage return to exit without changing the path. The final ">" of a directory name is optional for the path command so "Path Foo" changes the path to the directory foo.Dr. Path prints an error message if you try to Path to a nonexistant directory. If you attempt to set a Path from which the Shell cannot be accessed (which can happen if you change the SearchList), Path requires confirmation.

COMMAND DESCRIPTIONS

PAUSE

Pause can be used to suspend the execution of a command file and wait for user confirmation before proceeding. It takes a message as a parameter, prints it on the screen, and then waits for a carriage return on the keyboard before continuing. This command can be given by the user, but it is most useful in command files when some user action is required before proceeding, (for example, changing floppies).

COMMAND DESCRIPTIONS

PRINT

The Print command sends files to a printer (the GPIB address for Print is one). If the file does not begin with a form-feed character, Print supplies one. Also, Print supplies a form-feed at the end of every printed file.

For printers connected to the RS232 port, Print requires Z80 PROMs version 8.5 (or higher) for proper operation. You can find these two PROMs on the "IO" board (red color-coded) labeled with the version number. For printers connected to the GPIB (IEEE-488) port, the PROM versions are not relevant. Contact Three Rivers field service to exchange earlier PROMs.

Format:

PRINT [Filename][,filename2,...filenamen][/switch(es)]

The valid switches are:

Switch   Description
------   -----------

/BAUD=n

This switch sets the baud rate for the print device. The default is /BAUD=9600.

/BREAK

This switch forces a blank page between each page of the printed listing.

/NOBREAK

This switch omits the blank page between each page of the printed listing. The default is /NOBREAK.

/COPIES=n

This switch allows you to specify the number (n) of listings you wish to print. The default is /COPIES=1.

COMMAND DESCRIPTIONS

PRINT

/DIABLO

This switch directs Print to initialize for the Diablo 630 daisy wheel printer.

/HELP

This switch displays a description of the Print command and the associated switches. Note that /HELP does not print a file.

/HP

This switch directs Print to initialize for the Hewlett-Packard 7310A printer.

/LINEPRINTER

This switch directs Print to initialize for the TI 810 (or similar) printer.

/NARROW

This switch specifies narrow characters (16.5 characters per inch). The default is /NARROW.

/PAUSE

This switch instructs Print to stop the printer after each page of the listing. Thus, /PAUSE permits user interaction (for example, to change paper). Press RETURN to continue printing.

/NOPAUSE

This switch instructs Print to print successive pages without interruption. The default is /NOPAUSE.

/PLAIN

This switch directs Print to perform no initialization for a specific printer. Use /PLAIN with generic printers. The default is /PLAIN.

COMMAND DESCRIPTIONS

PRINT

/SHIFT=n

This switch instructs Print to shift the listing a number (n) spaces to the right. You cannot specify a negative number (shift the listing to the left). The default is /SHIFT=0.

/SHORT

This switch specifies short characters (eight lines per inch). The default is /SHORT.

/START=n

This switch permits you to specify the page number (n) of the document to begin printing. The default is /START=1.

/STOP=n

This switch permits you to specify the last page number (n) of the document to print. The default for /STOP prints the last page of the document.

/TABS=n

This switch provides tab stops every n characters. The default is /TABS=8.

/TALL

This switch specifies tall characters (six lines per inch).

/TITLE

This switch instructs Print to print a title line at the top of each page. The default for all files except files with a .DOC extension is /TITLE.

COMMAND DESCRIPTIONS

PRINT

/NOTITLE

This switch instructs Print to omit the title line at the top of each page. The default for all files with a .DOC extension is /NOTITLE.

/WIDE

This switch specifies wide characters (10 characters per inch).

COMMAND DESCRIPTIONS

RENAME

The Rename command changes the name of an existing file.

You can rename a file from one directory to another (move the file) as long as both directories are in the same partition. You can rename a .Run file, but note that if you rename a .Seg file, you must re-link the programs which use it.

Format:

    RENAME SourceFile[~]DestinationFile[/switch]

Rename prompts for any missing arguments.

Valid switches are:

    Switch   Description
    ------   -----------

    /ASK

            This switch requests verification before renaming a
            file. /ASK is the default.

    /NOASK

            This switch overrides verification requests.

    /CONFIRM

            This switch requests confirmation before changing the
            name of a file to an existing file name. /CONFIRM is
            the default.

    /NOCONFIRM

            This switch overrrides confirmation requests.

    /HELP

            This switch displays a description of the Rename
            command and the associated switches. Note that /HELP
            does not rename a file.

COMMAND DESCRIPTIONS

RENAME

The source file for Rename can contain wild cards (for a description of the wild cards, see Chapter 2). If the source contains wild cards, the destination must contain the same wild cards in the same order. If you include wild cards in the source file, Rename finds all files in the directory which match the source pattern. For these files, the part of the source file name that matched each wild card is used to replace the corresponding wild card in the destination. As an example, for the command:

        RENAME foo*.abc#   anotherdir>*baz.rmn#z

The input file "FOOZAP.ABCD" would be renamed to the new file "anotherdir>ZAPbaz.rmnDz".

If wild cards are used, Rename asks for verification of each file renamed. This can be disabled with the switch "/NOASK" or enabled with the switch "/ASK." The default is /ASK.

Wild cards are not allowed in the directory part of the source file.

When the source file does not contain wild cards, the destination file can contain, at most, one occurence of the asterisk (*) wild card. In this case, the non-directory part of the source replaces the "*" in the destination. For example,

          RENAME sys:Boot>newOS>myprog.Pas dir3>new.*

renames the file "sys:Boot>newOS>myprog.Pas" to a new file named "dir3>new.myProg.Pas". This is most useful when you want to rename a file from one directory to another with the same name. For example,

          RENAME dir1>prog.Pas *

moves prog.pas from the directory "dir1" into the current directory.

If there are no wild cards in the source, an attempt to include wild card characters other than the single "*" in the destination, leads to problems later. Rename treats these extra wild cards as simple literal characters. Because a file with wild cards in its name is hard to specify, Rename requires confirmation before creating such a file.

COMMAND DESCRIPTIONS

RENAME

If the the destination file already exists, Rename requests
confirmation before deleting. You can disable the confirmation
request with the /NOCONFIRM switch. The default is /CONFIRM. When
you specify the /NOCONFIRM switch, you implicitly specify the /NOASK
switch.

If an error is discovered and wild cards were used, Rename asks
whether or not to continue processing the rest of the files that match
the input. This confirmation is required regardless of switches you
specify.

COMMAND DESCRIPTIONS

RERUN

ReRun is a convenient method to reexecute the default file remembered
by the Shell and supply new arguments to   the   runfile.   The   command
line is:

     ReRun arguments

The   ReRun   command   is   most   useful   when   the   default   file has an
especially long name.   For example, if   "Programwithlongname"   is   the
default file, the command:

     ReRun A b 3

is the same as typing:

     Programwithlongname A b 3

COMMAND DESCRIPTIONS

RUN

Run is another way to invoke the default file remembered by the Shell. If you have been editing, compiling and linking the default file, simply typing:

RUN

executes that default file. The Run command sets the default file to the specified runfile. For example, the command:

RUN Foo arg1 arg2

is the same as typing:

Foo arg1 arg2

except that the first command (Run Foo arg1 arg2) sets the default file to Foo.

COMMAND DESCRIPTIONS

SCAVENGER

The Scavenger checks the file system structures and fixes any detected errors. You can call the Scavenger any time you suspect a file system problem or whenever you need to reconstruct a directory.

Format:

    Scavenger

The Scavenger prompts for all parameters.

Utility Program Guide

COMMAND DESCRIPTIONS

SETBAUD

SetBaud allows you to specify the baud rate to the RS232 line. Valid
baud rates are: 110; 150; 300; 600; 1200; 2400; 4800; and 9600. The
command syntax is:

SetBaud 4800

COMMAND DESCRIPTIONS

SETTIME

The SetTime command allows you to change the time and/or date. SetTime can get the new time from one of three sources: the user; the network time server; and on PERQ2 machines, the EIO clock.

Format:

        SetTime [DD MMM YY] [HH:MM][:SS][/switch]

If you specify the time, note that the time notation uses a 24 hour clock. The seconds are optional. You cannot specify a year prior to 1980 (80). Correct only the time by omitting the date. The following example sets the date to June 3, 1983 and the time to 3:30 PM:

        SetTime 3 Jun 83 15:30

If you direct SetTime to acquire the time, omit the date and time and simply specify the switch to identify the source.

The valid switches are:

        Switch   Description
        ------   -----------

        /USER

                This switch permits the user to enter a date and time,
                using the notation described above. If you do not
                enter a new date or time, SetTime prompts for a value.
                /USER is the default.

        /SERVER

                This switch sets the date and time based on the current
                time on the network time server.

        /EIOBOARD (PERQ2 machines only)

                This switch, valid only on PERQ2 machines, sets the
                time from the EIO clock. Note that SetTime adds an
                offset to the EIO clock time to get the current date
                and time. You can change this offset using the
                /SetGMTOffset switch.

COMMAND DESCRIPTIONS

SETTIME

/SETGMTOFFSET

This switch sets the local to Greenwich Mean Time (GMT) offset. The GMT offset is added to the running time from the EIO board to obtain the current date and time.

COMMAND DESCRIPTIONS

STATISTICS

The operating system constantly collects statistics about the performance of the swapper. The Statistics command permits you to display the information after each process executes.

Format:

    STATISTICS option

The options are Yes and No. Yes displays the statistics after each process executes and No ends the display.

After each process executes, the Shell prints the Statistics for that program in the following format:

        Load   1.6 secs.
        Exec   3.6 secs.
        IO   0.8 secs.
        Swap   0.1 secs.
        Move   0.0 secs.
        Duty   97.2 percent.

"Load" is the time spent loading the program and its modules into memory.

"Exec" is the total time spent executing including IO, Swap and Move times.

"IO" is the time spent doing Unit-level IO not including IO time spent swapping.

"Swap' is the amount of time spent swapping.

"Move" is the amount of time spent moving segments from one place in memory to another to try to find room for new allocation.

"Duty" is the proportion of time spent in actual work. The system computes the duty factor as follows:

        100 * (Exec - Swap - Move) / Exec

COMMAND DESCRIPTIONS

SWAP

The Swap command enables or disables the virtual memory system.

Format:

    SWAP option [partition]

The options are Yes and No. Swap Yes enables the virtual memory system and Swap No disables it. If you specify Yes, you can specify a partition to use for the swap files. For example,

    SWAP YES Sys:Boot>

enables swapping to the Sys:Boot> partition. If you specify No, Swap forces all active segments into memory and then disables the virtual memory system.

When booting from the hard disk, swapping is initially enabled in the partition containing the boot file. If you simply specify SWAP YES with no partition name, Swap uses this default partition. When booting from the floppy, swapping is initially disabled. We discourage swapping to a floppy disk because floppies have a small capacity, are slow, and may be dismounted at any time.

COMMAND DESCRIPTIONS

TYPEFILE

The TypeFile command displays any file or files on the console.

Format:

        TYPEFILE [FileSpecification][,file2][...,filen][/switch]

TypeFile does extension completion on the file name you specify. If the file to print is FOO.PAS, you only need type FOO. The extensions that type knows about, in order tried, are: Pas, FOR, Micro, Cmd, Dfs, and Doc.

If you omit the file name, TypeFile displays the default file; the same default file used by other commands (refer to Chapter 1). Unlike other commands, TypeFile does not change the default file name.

Note that to display multiple files, sequentially, you must. separate the file specifications with a comma (,).

The valid switches are:

        Switch   Description
        ------   -----------

        /WAIT   This switch instructs Typefile to stop after displaying
                        a screenful of text and to stop when it finds a
                        formfeed character (CTRL/L) in the file. Thus the
                        switch permits you to read the page before the screen
                        is erased and the next page displayed. To continue,
                        type CTRL/Q. /WAIT is the default.

        /NOWAIT   This switch instructs TypeFile to display the file
                        without interruption. You can stop and start the
                        display by pressing CTRL/S to stop and CTRL/Q to
                        continue.

        /FONT=fontname

                        This switch specifies the font (fontfile) to use when
                        displaying a file.

        /HELP   This switch displays a description of the TypeFile
                        command and the associated switches. Note that /HELP
                        does not display a file on the console.

COMMAND DESCRIPTIONS

TYPEFILE

TypeFile indicates the end of the file with a solid, left pointing triangle.

COMMAND DESCRIPTIONS

USERCONTROL

UserControl maintains the System.Users file which contains information about valid users and their passwords, group identifiers, and user profiles. This information is used by the Login program (see the Login command description). The System.Users file must be in the root directory (top level) of the partition where the boot file is.

Format:

    USERCONTROL [command][/switch(es)]

If you include a command, UserControl executes the command and then exits. If you do not include a command, UserControl prompts with:

    USERCONTROL>

and waits for input.

The following are the valid commands:
    Command   Description
    -------   -----------

    HELP

            Displays a description of the UserControl command and associated switches.

    ADDUSER [username][/switch(es)]

            Adds a new user to the user file or, updates the information for an existing user. If you omit a username, the command prompts for one. Any printing character except blanks, spaces, equal signs (=), commas (,), or slashes (/) are valid usernames. The maximum number of characters is 31. AddUser accepts the following switches:

                /PASS - permits users to change their password or enters a password for a new user. Note that when you specify the /PASS switch, UserControl prompts for the password. You can enter any printing character except blanks, spaces, equal signs (=), commas (,), or slashes (/). The

COMMAND DESCRIPTIONS

USERCONTROL

maximum number of characters is 31. You can
also respond to the password prompt with CR,
which enters a null password.

If you omit the switch for a new user, the
password defaults to null. For an existing
user, the password is unchanged if the switch
is omitted.

/GROUP=[group id] - permits an existing user to
change the group id or enters a group id for a
new user. The value for group id can be any
integer from 0 to 255 inclusive. If you do not
specify a value, you are prompted for one.

If you omit the switch for a new user, the
group id defaults to 1. For an existing user,
group id is unchanged if the switch is omitted.

/PROF=[profile] - specifies the complete path
for the user's profile file.

If you omit the switch for a new user, the
default is SYS:USER>name>PROFILE. For an
existing user, the profile is unchanged if
you omit the switch.

CHECKUSER [username]

Validates   a   username   and   password.   If you omit the
username, the command prompts for one.

NEWFILE

Destroys the existing System.Users file and   creates   a
new one.

LISTUSERS

Displays a list of all of the valid users.

COMMAND DESCRIPTIONS

USERCONTROL

REMOVEUSER [username]

Delete a user from the file. If you omit the username, the command prompts for one.

QUIT

Exits the program.

You can also specify the /HELP switch (type /HELP or press the HELP key) with any command or in response to any prompt to display specific help information.