WICAT Multi-user Control System

WMCS

User's Reference Manual

188-101-201 E

May 1985

• Software •
Publications

WICATsystems

Printed in the United States of America

Receipt of this manual must not be construed as any kind of commitment, on the part of WICAT Systems Incorporated, regarding delivery or ownership of items manufactured by WICAT.

This manual is subject to change without notice.

first printing	July 1981
second printing	August 1981
third printing	July 1983
fourth printing	April 1984
fifth printing	June 1985

Information about this Manual

Review the following items before you read this publication.

The subject of this manual

This manual describes the Command Interpreter Program (CIP) that is part of the WICAT Multi-user Control System (WMCS).

The audience for whom this publication was written

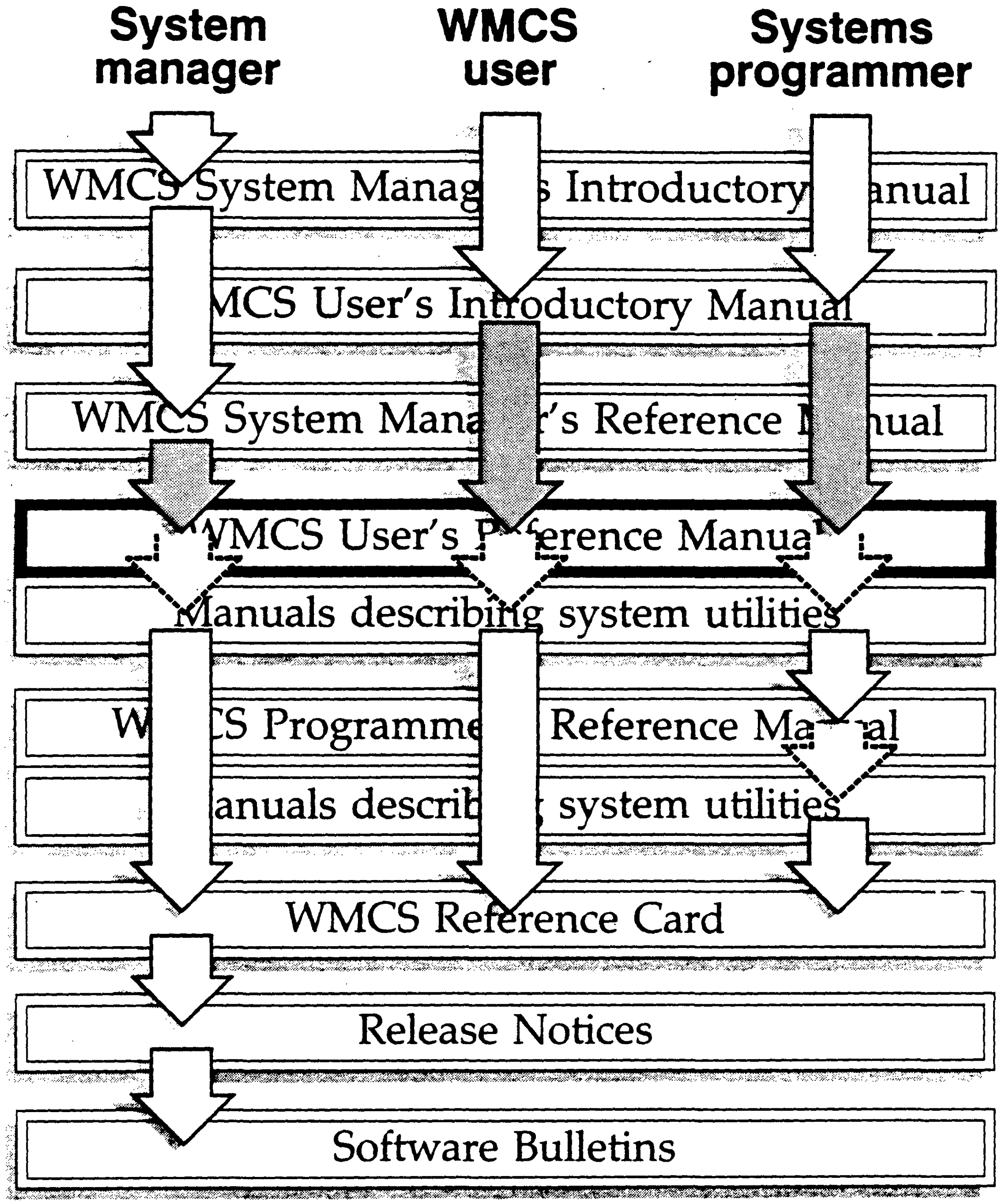
This manual is written for users who have at least completed the tutorials in the WMCS User's Introductory Manual. It is also written for system managers who have at least completed the tutorials in the WMCS System Manager's Introductory Manual.

Related publications

The Reader's Guide to WMCS Publications shows the relationship of this manual to the other WMCS manuals.

Reader's Guide to WMCS Publications

Instructions: Determine the audience to which you belong and then read *only* the publications at an *arrowhead.* Dotted arrowheads indicate optional reading.



System manager

WMCS user

Systems programmer

WMCS System Manager's Introductory Manual

WMCS User's Introductory Manual

WMCS System Manager's Reference Manual

WMCS User's Reference Manual

Manuals describing system utilities

WMCS Programmer's Reference Manual

Manuals describing system utilities

WMCS Reference Card

Release Notices

Software Bulletins

Key to color code:

Vol. 1	Vol. 2	Vol. 3	Vol. 4	Card

Insert this guide behind the first divider in volume 1. 188-190-105 B

Typographical Conventions Used in this Publication

Bold facing indicates what you should type.

Square brackets, [], indicate a function key, the name of which appears in uppercase within the brackets. For example, [RETRN], [CTRL], etc.

Underlining is used for emphasis.

Table of Contents

Chapter 1 Communicating with the WICAT Multi-user Control System

Chapter 2 The Geography of the System

Chapter 3 Using the CIP Command Line

Chapter 7   Using CIP Commands in Files

Chapter 8   Parameter Files

Chapter 9   Background Processing

Chapter 10   Input/output Redirection

Chapter 11   Logical Name Assignment

Chapter 12 Diagnostic Messages

Appendix A Glossary of WMCS Diagnostic Messages

Appendix B System Logical Name Assignments

Appendix C System Time

Chapter 1

Communicating with the WICAT Multi-user Control System

The WICAT Multi-user Control System (WMCS) is the program, or software, that superintends all aspects of your system, ultimately causing the hardware to do what you tell it to do.

However, communicating directly with the WMCS each time you want to create a file or directory, switch to another disk, etc., would require the instructions for that operation to be in terms the WMCS understands. In other words, in order for the WMCS to communicate with your system's hardware, the WMCS must speak a language that the hardware can understand. Using this language every time you want to tell the WMCS to do something would be tedious and time consuming, and would require considerable expertise in dealing with computers.

Therefore, to expedite your interaction with the WMCS (particularly if you are unfamiliar with computers), a program has been developed that translates user commands into terms that the WMCS understands. It is called the Command Interpreter Program (CIP).

Figure 1.1 illustrates CIP's function as the interpreter that allows you to communicate with the WMCS.

Thus, when you type the mnemonic DIR onto the CIP command line and strike [RETRN], the CIP translates that mnemonic into the following instructions for the WMCS:

1. Find the DIR Program. It is stored in a file named DIR.EXE on one of the system's devices.

2. Do what the contents of DIR.EXE tell you to do.

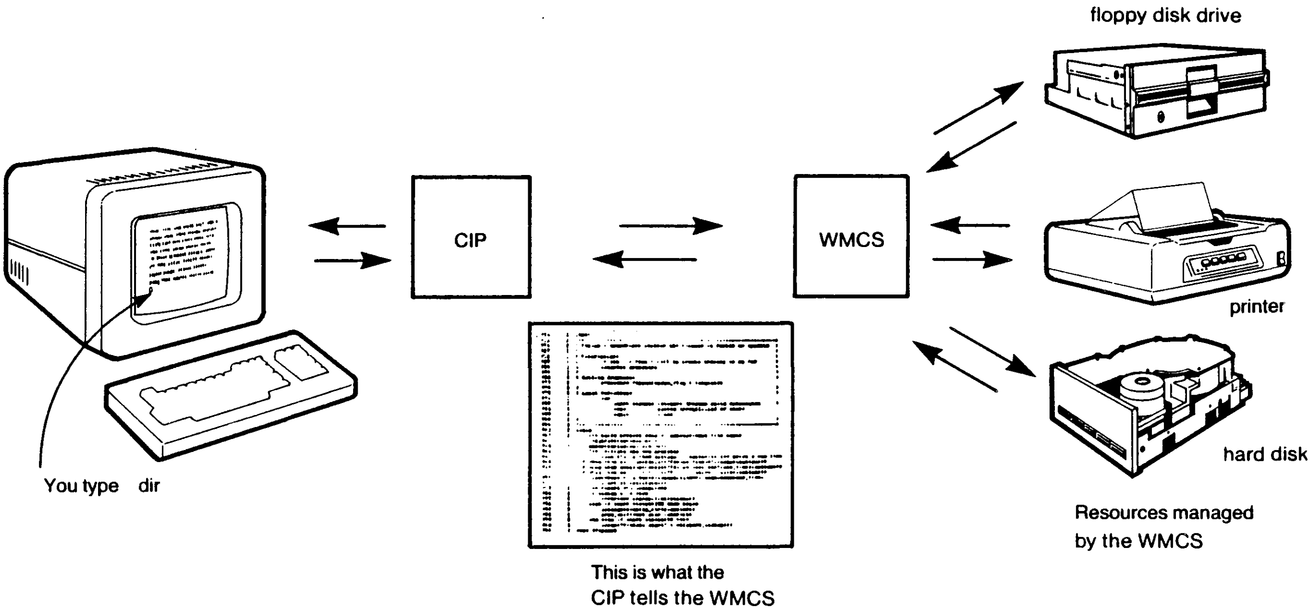

floppy disk drive
CIP
WMCS
printer
hard disk
You type dir
Resources managed by the WMCS
This is what the CIP tells the WMCS

Fig. 1.1 The CIP: Your means of communicating with the WMCS

All of the work in a computer is done by means of programs (files like DIR.EXE) that tell the system how to do what you want it to do. When you type a CIP command onto the command line and strike [RETRN], you are actually telling the WMCS to find and execute the program that in turn tells the system how to perform that particular function. In other words, the CIP is a program that tells the WMCS how to perform various functions.

Executing the CIP Itself

When your system manager assigns you a user account, he designates a program that allows you to communicate, or interface, with the WMCS. So, when you log onto the system, the file containing the interface program is executed. If CIP is your interface, a copy of CIP.EXE is executed whenever you log on.

Follow this procedure to log on to your system:

Step 1 Turn the power switch on your terminal to the on position.

After the necessary files have been executed, this kind of report appears on the screen:

```
(boot$cip) WARLOCK is up and running.
```

NOTE: WARLOCK represents the name the system manager has given your system.

Step 2 Strike RETRN

This prompt appears on the screen:

```
Welcome to Warlock
Username:
```

Step 3 Type the username assigned to you by your system manager.

Step 4 Strike RETRN

This prompt appears on the screen:

```
Password:
```

NOTE: If the password prompt does not appear read the information under step 6.

Step 5

Type the password assigned to you by your system manager.

Do not be concerned that your password does not appear on your terminal screen as you type. The password is transmitted, but not to the screen.

Step 6

Strike RETRN

If you make a mistake in typing either the username or the password, this diagnostic message appears on your screen:

```
LOGON - User validation error.
```

Furthermore, if you wait more than several seconds to type either the username or the password, this diagnostic message appears on your screen:

```
LOGON - Input time-out error.
```

If either of these messages appears, strike [RETRN], and repeat the logon procedure, i.e., go back to step 2.

The following kind of display appears on the screen when the username and the password are accepted:

```
=======================================================
System Bulletins

=======================================================
GRACE>
```

When both the username and the password have been correctly typed, and you strike [RETRN] after having typed your password, three command files are executed:

```
LOGON.COM

LOCALLOG.COM

USERUP.COM
```

What then appears on the screen depends upon the contents of these three files.

Only the system manager has access to LOGON.COM and LOCALLOG.COM, command files containing logical name assignments, messages, etc., that pertain to all the users on the system.

A USERUP.COM file should appear in each user-account default directory because the system manager usually creates this file when he creates your user-account default directory. Use the VEW Command to place logical name assignments, etc., in the files that pertain to your working environment (read chapter 6 for detail).

When your logon is completed and the cursor appears next to your username and a right angle bracket, >, at the bottom of the screen, the commands and keyboard operation explained in this manual apply.

The User-account Default Directory

When the system manager creates a user account for you, he also assigns you a _user-account default directory_, a directory in which you find yourself whenever you log on to the system with the username and password assigned to that account.

In other words, just as your user account determines the nature of your interface with the WMCS, your account also assigns you to a _geographic_ location within the device and directory structure that together constitute the system. Chapter 2 explains the geography of the devices and the directory structure, and Chapter 3 tells you what a default is.

Chapter 2

The Geography Of The System

There are two aspects to the geography of your system:

1. Physical location.

2. Conceptual location.

Physical location refers to your system's hardware. This means that when you log on to the system, you log on to a particular terminal, or device, establishing a specific physical location for your use. For example, when you insert a tape into a tape drive, that tape is located in one of the mechanical devices that constitute your system's hardware, establishing a specific physical location for the tape.

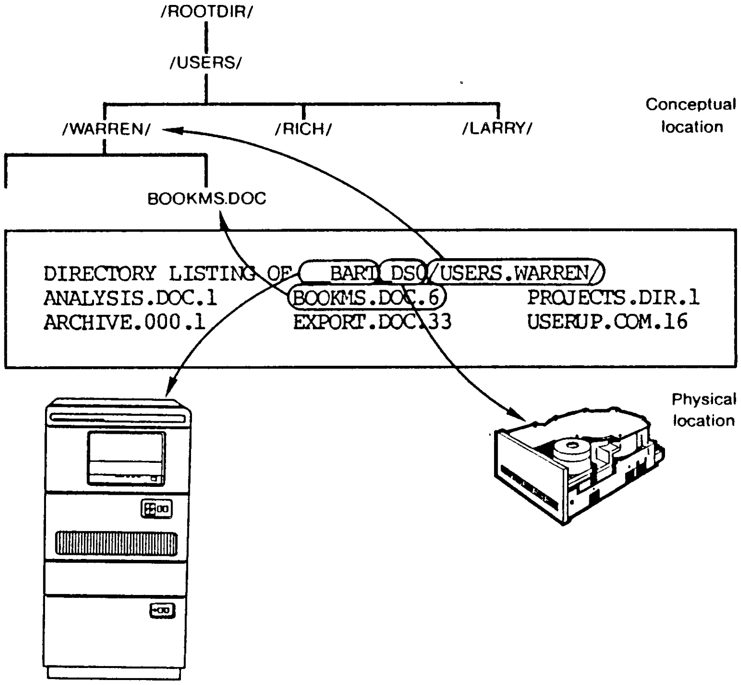
Therefore, the first step in finding your way about the system is knowing the names of the devices that make up the system. If your system is part of a network, you should also know the names of the nodes (other computers) on your network. Beyond that, however, the WMCS, not the user, is primarily concerned with the physical location of data.

For example, suppose that you log on, type **dir** onto the CIP command line, and strike [RETRN] to find out what your user-account default directory contains. This kind of report appears on your terminal screen:

```
DIRECTORY LISTING OF __BART_DS0/USERS.WARREN/
ANALYSIS.DOC.1      BOOKMS.DOC.6        PROJECTS.DIR.1
ARCHIVE.000.1       EXPORT.DOC.33       USERUP.COM.16
```

To provide this kind of report, the WMCS must determine which data, of all the data on a particular device, constitute the directory. The grouping of data into files, and the grouping of files into a directory are conceptual matters, and the manner in which the data are stored on the device has little to do with the user's perspective. In other words, the user perceives such a relationship among the data, the WMCS does not.

Therefore, the nodename and devicename that appear as the first elements on the first line of this display are the only part of the report that pertains to the physical location of the data. The remainder of the display, and the files themselves, are conceptual associations of data.



```
DIRECTORY LISTING OF __BART_DS0/USERS.WARREN/
ANALYSIS.DOC.1          BOOKMS.DOC.6          PROJECTS.DIR.1
ARCHIVE.000.1           EXPORT.DOC.33         USERUP.COM.16
```

Fig. 2.1 Physical and conceptual location of data

How the WMCS determines the physical location of data, based upon what you type onto the CIP command line, is explained in other documentation for those users and programmers concerned with it.

This chapter is concerned primarily with the conceptual geography of your system.

Location of the Home Directory

When the system manager creates a user account for you, he assigns you to a default node, a default device and a user-account default directory (your home directory).

The node is the computer to which all the devices are connected, e.g., __BOB.

The device on which your home directory is located is most likely the hard disk that is the primary device on your system, e.g., _DC0.

The directory in which you find yourself when you log on is your user-account default directory, or home directory. The system manager creates a /USERS/ directory in which your home directory is usually located. Figure 2.2 illustrates the typical relationship to the home directories of each user on a system.

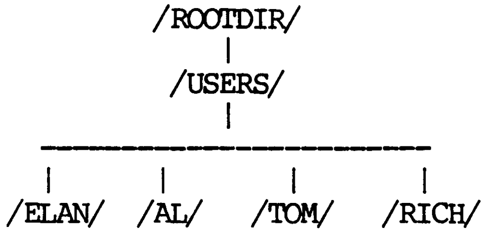


Fig. 2.2 Conceptual location of home directories

In other words, each user's home directory is a subdirectory of /USERS/.

/USERS/ is usually a subdirectory of /ROOTDIR/, the main, or root, directory on the system. Figure 2.3 illustrates the relationship between /USERS/ and the typical contents of /ROOTDIR/.

```
Directory listing of __BART_DS0/ROOTDIR/
BITMAP.SYS.1        KERNEL.156.4        SYSHLP.DIR.1
BOOTDISK.156.4      KERNELBUG.156.4     SYSLIB.DIR.1
DISK.156.4          KSAM.156.2          SYSEXE.DIR.1
FCB.SYS.1           MATH1010.156.2      TTY.156.4
FCBBITMAP.SYS.1     OSINIT.156.2        USERS.DIR.1
```

Fig. 2.3 Standard files and directories in /ROOTDIR/

The Contents of /ROOTDIR/

The following sections describe the contents of each major subdirectory and kind of file in /ROOTDIR/.

/SYSEXE/

This directory contains the executable image file for each CIP command, i.e., the files in this directory contain the programs you execute when you type a command mnemonic onto the CIP command line.

/SYSLIB/

This directory is reserved for data files used by the system.

/SYSHLP/

This directory contains the help display for each CIP command. In other words, when you type DEL? onto the CIP command line, the contents of DEL.HLP (located in /SYSHLP/) appear on the screen.

/SYSDSR/

This directory contains device drivers. A device driver is the program assigned to each mechanical device allowing the CPU to communicate with it. These programs are part of the WMCS.

/SYSINCL/

This directory contains record definitions for the development of programs that use system data structures.

/SYSTMP/

This directory is accessible to all users and is for temporary files only. Standard operating files should not be stored in it because it is purged each time the system is booted.

Application Directories

The number of applications on your system determines the number of application directories you have.

System Files

Do not tamper with system files because they define the file structure maintained by the system. Two such files are FCB.SYS, and FCBBITMAP.SYS.

WMCS Files

These files contain the WICAT Multi-user Control System itself, and are loaded into memory when the system is booted. The standard file extensions for WMCS files are .100 and .156, depending upon which system you are on. For detail on which file extension goes with which system, see the table of file extensions in this chapter.

File Identification

Files are identified by a file designation. This designation consists of six elements:

1. The name of the node to which the device is connected.
2. The name of the device on which the file is stored.
3. The name of the directory in which the file is listed. If a file is in a subdirectory, the directory-name field of the file designation includes all directories through which the system must search to find the file.
4. The filename.
5. The file extension.
6. The version number of the file.

These six parts of the file designation constitute the pathname for a directory or file. In other words, the name itself is the path among the nodes in your network, the devices on your system, and through the structure that leads to a directory or file.

Nodename	Devicename	Directory name	Filename	File extension	Version no.
__bob	_dc0	/users.al/	report.	doc.	10

Fig. 2.4 The parts of the file designation

Nodename

Nodename	Devicename	Directory name	Filename	File extension	Version no.
__bob	_dc0	/users.al/	report.	doc.	10

A network is a group of two or more computer systems linked together. Devices and the files stored on them can be shared among the different computers, called nodes, on the network. Nodenames identify each computer in a network. Even though your computer may not currently be part of a network, it will have a nodename assigned to it.

A nodename can contain up to sixteen alphanumeric characters, as well as the tilde, ~, and the dollar sign, $. Nodenames are preceded by two underscores. e.g., __BOB.

Devicename

Nodename	Devicename	Directory name	Filename	File extension	Version no.
__bob	_dc0	/users.al/	report.	doc.	10

A computer system is composed of a Central Processing Unit (CPU), a memory, I/O (input/output) devices, and various peripheral devices.

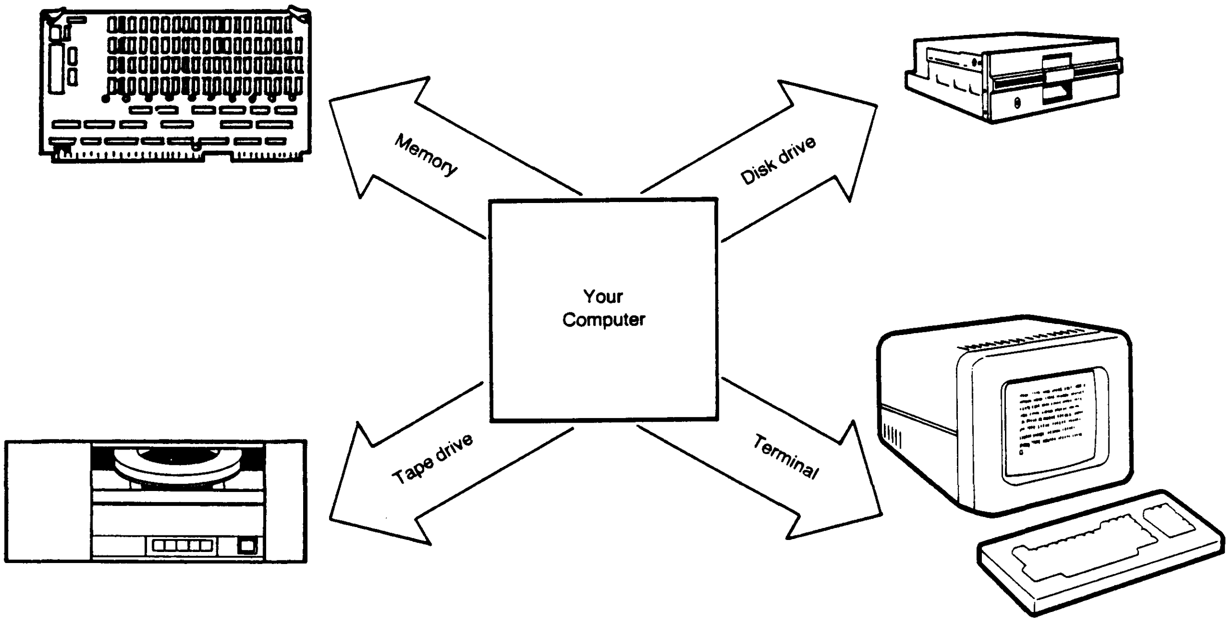


Fig. 2.5 The fundamental parts of a computer

There are several kinds, or classes, of devices, but the two most common are:

1. Character-oriented devices such as terminals, printers, etc. These are often referred to as TTY devices.

2. Directory-oriented devices such as disks and tapes.

Character-oriented devices do not contain files, but the WMCS treats them as though they were files. A terminal is not a device on which a file can be created or permanently stored; it is character- rather than directory-oriented.

Directory-oriented devices, such as disks, tapes, etc., are the devices on which files are stored.

The name of a device generally indicates which of these two categories the device belongs to. For example, _DC0 and _DX0 are common names for disks, while _TT1, _TT2, are common names for terminals.

The devicename can consist of up to eight alphanumeric characters after the underscore. For each of the standard devices on a system, devicenames have been suggested. However, you can name devices anything you want, within the character limit.

Directory Name

Nodename	Devicename	Directory name	Filename	File extension	Version no.
__bob	_dc0	/users.al/	report.	doc.	10

Directories are groups of files located on a particular device. For example, __BOB_DC0/SYSLIB/ indicates that the directory, /SYSLIB/, is located on the disk named _DC0 and the node named __BOB. Note that slashes are used to enclose directory names.

Furthermore, even though /ROOTDIR/ is the first-level directory for all files on a device, it need not be given as part of the file designation unless you refer to a file in /ROOTDIR/ itself. In other words, even though /SYSLIB/ is a subdirectory of /ROOTDIR/, it is not necessary to specify __BOB_DC0/ROOTDIR.SYSLIB/ for /SYSLIB/.

The file designation must contain the names of each directory on the path to the file. For example, __BOB_DC0/SYSLIB.CORRES/MEMO.TXT indicates that MEMO.TXT is located in /CORRES/, a subdirectory of /SYSLIB/, on device _DC0, on node __BOB.

Figure 2.7 illustrates the nature of the conceptual directory structure for the data on a device.

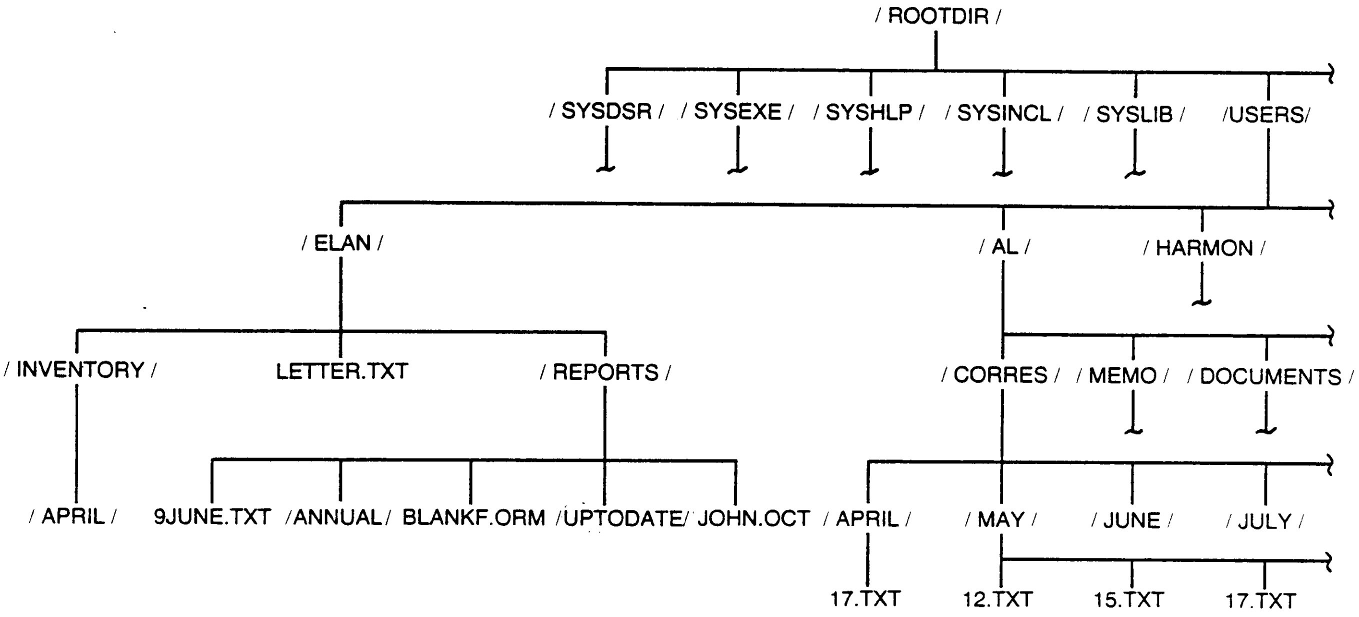


Fig. 2.7 Conceptual directory structure for a device

The DEF Command allows you to find out what device you are on and what directory you are in.

Filename

Nodename	Devicename	Directory name	Filename	File extension	Version no.
__bob	_dc0	/users.al/	report.	doc.	10

The filename you assign can contain no more than nine alphanumeric characters. The dollar sign, $, and tilde, ~, can also be included, and any of these characters can be the first in the filename.

Any lowercase characters in the filename appear as uppercase letters when the filename appears in directories and system-generated messages, i.e., lowercase characters are automatically converted to uppercase.

File Extension

Nodename	Devicename	Directory name	Filename	File extension	Version no.
__bob	_dc0	/users.al/	report.	doc.	10

Each file designation also contains a user-assigned extension consisting of from zero to three alphanumeric characters, as well as the dollar sign, $, and the tilde, ˜. The file extension can be used to indicate file type, e.g., .DOC for a file containing a document, .TMP for a temporary file, .DAT for a data file, etc.

Although you can formulate your own scheme for file extensions, the following list may be helpful in systematizing the designations you use:

Standard File Extensions

.100	WMCS for the System 200,220,300
.156	WMCS for the System 150MU,150WS,155,160
.A	Library file
.A68	C assembler source file
.ARC	Archive file
.ASM	Assembly language source file
.BAK	Backup file
.BAS	BASIC source file
.BTF	Batch terminal emulator file
.C	C-language source file
.CBL	COBOL source file
.CKS	File containing checksums
.COM	Command file
.DAT	Data file
.DEF	Library definition file
.DIR	Directory file
.DSP	Display screen file for WISE
.DSR	Device Service Routine (device driver)
.DOC	File containing a document
.EXE	Executable image file
.F77	FORTRAN77 source program
.H	Include-files for C
.HLP	Help display
.IDX	CIS COBOL ISAM file
.INT	Intermediate code file (Pascal and COBOL)
.IT	CIS COBOL data ISAM file
.KEY	KSAM key file
.LST	Program listing file
.MCR	Linker cross reference file
.MRL	Relocatable object module
.NAM	Name association table for WISE frames
.O	Object files for C and FORTRAN
.PAS	Pascal source file
.PRM	Parameter file
.PRN	Program listing file
.STP	Setup files for WISE
.SYS	File maintained by the system (FCB.SYS, BITMAP.SYS, etc.)
.TMP	Temporary file
.TXT	File containing text
.UCP	UltraCalc pointer file
.UCS	UltraCalc string file
.UCX	UltraCalc script file
.WIS	WISE graphic and logic frames

File Version Number

Nodename	Devicename	Directory name	Filename	File extension	Version no.
__bob	_dc0	/users.al/	report.	doc.	10

The original version or copy of a file appears as version number one in a directory. Subsequent copies or versions are automatically assigned a version number that is one integer greater than the highest existing version number for a copy of that file in the directory.

Furthermore, you can delete old versions of a file without affecting the version number of the file that is kept. For example, if you have four versions of a file, and you delete the first three versions, the fourth version retains four as its version number, and the numbering of subsequent versions of that file begins with 5.

Unless you specify a version number, when naming a file as part of the CIP command line syntax for a particular command, the most recently created version (or the file with the highest version number) is used in the execution of that command.

The WMCS allows up to 65,535 versions of a file.

Relative Addressing

Typing the complete file designation each time you specify a directory or file can be tedious. Therefore, the CIP provides relative addressing, i.e., the specification of a directory or file relative to "where you are" among the devices and directories that make up your system.

For example, when you log on, you are located in your home directory, e.g., __BOB_DC0/USERS.AL/. If you had to type the complete file designation when calling for a listing of the contents of that directory, the command line character string would look like this:

```
dir __BOB_dc0/users.al/
```

Therefore, any CIP command requiring a file designation for a parameter automatically assumes (unless you specify otherwise) your working-default node, device, and directory.

Consequently, you can log on, type DIR onto the CIP command line, strike [RETRN], and a list of the contents of your home directory appears on your terminal screen. The CIP assumes (since DIR was all you typed on the command line) that you want a directory listing for the default directory.

Suppose you want to rename one of the files in your home directory. You might type this kind of command-line character string to do so:

```
ren report.doc april.rpt
```

Inasmuch as you specify only the filename and file extension of the file to be renamed (the first file listed), the CIP assumes that you mean the REPORT.DOC located in your working-default directory and on your working-default device and node. Furthermore, since the second file designation consists of only a filename and file extension, the CIP assumes that you want the default device and directory to be assigned to the second file designation.

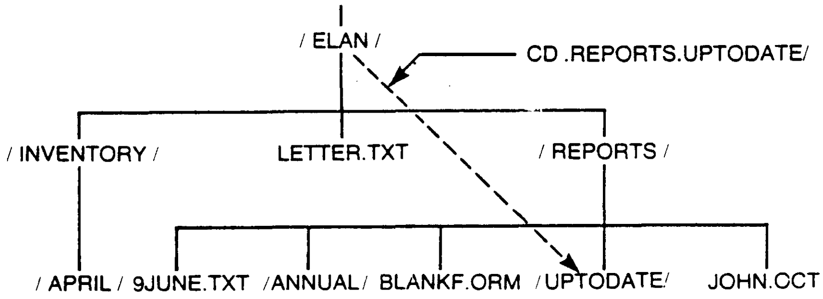
Finally, suppose you want to rename REPORT.DOC and assign it to another directory on the default device. The following kind of character string performs that operation:

```
ren report.doc tom/april.rpt
```

In other words, the default device is assumed for both designations, but the default directory is assumed for the first file only.

The use of the CD Command further illustrates the significance of your default device and directory.

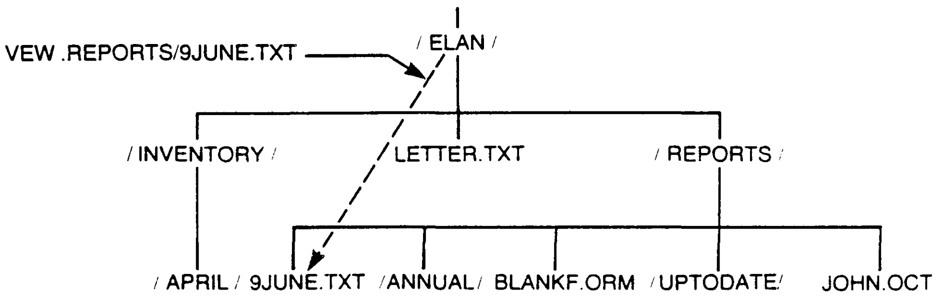
Were /ELAN/ your default directory, and you wanted to move to __BOB_DC0/ ELAN.REPORTS.UPTODATE/, you need type only CD .REPORTS.UPTODATE/in order to do so.



Had you already used the CD Command to go to, or to change your default directory to, /ELAN.REPORTS/, you would be in the /ELAN.REPORTS/ directory, and need type only CD .UPTODATE/.

If you wish to specify a file that is farther down the directory structure from where you are, give that portion of the file designation, preceded by a period, ., that is not part of the default.

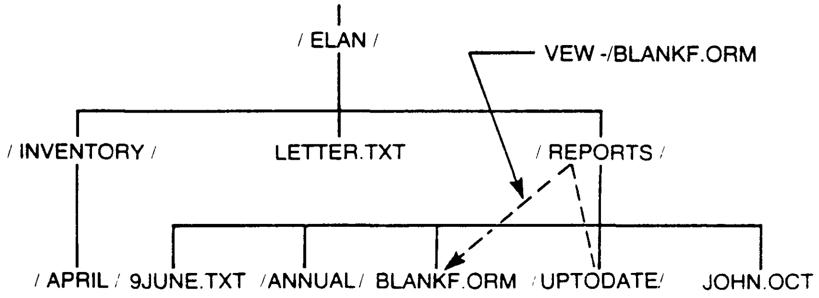
For example, suppose you want to edit a file whose pathname is __BOB_DC0/ ELAN.REPORTS/9JUNE.TXT, and that you are in /ELAN/. You need type only VEW .REPORTS/9JUNE.TXT to access that file.



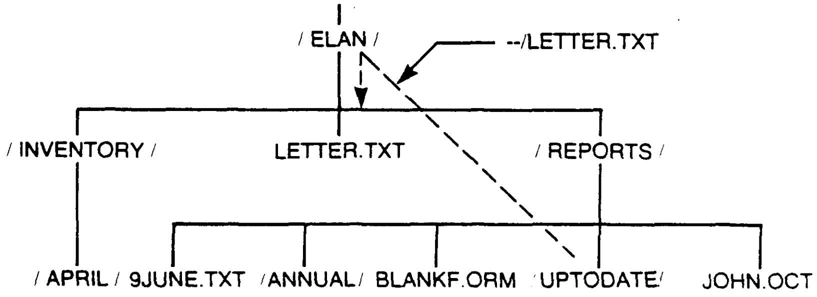
Were you already in /ELAN.REPORTS/, you would need to type only VEW 9JUNE.TXT to access the file.

To access a file that is farther up the directory structure from where you are, you can use the minus sign, -. For example, suppose you are in /ELAN.REPORTS.UPTODATE/, and want to access a file whose designation is _BOB_DC0/ELAN.REPORTS/BLANKF.ORM.

You need type only -/BLANKF.ORM as the pathname to the file. In other words, the minus sign indicates that you wish to move one directory up the directory structure, and then access BLANKF.ORM.

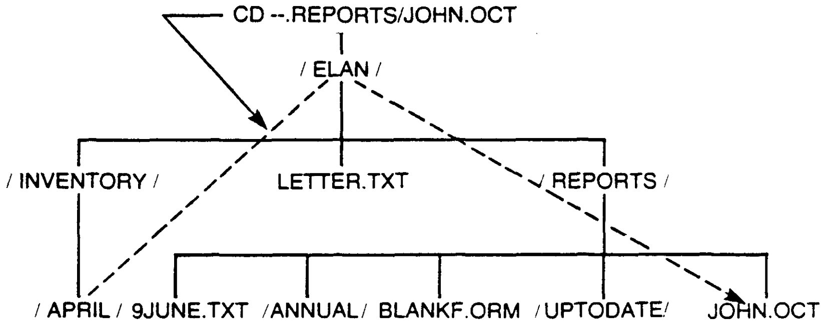


If you wanted to specify a file in /ELAN/, and /ELAN.REPORTS.UPTODATE/ were your default directory, you could give --/LETTER.TXT as the pathname. Note that you need not put a period between the two minus signs.



Finally, suppose you want to specify a file whose designation is __BOB_DC0/ELAN.REPORTS/JOHN.OCT, and that __BOB_DC0/ELAN.INVENTORY.APRIL/ is the default.

You could specify the file by typing --.REPORTS/JOHN.OCT as the pathname. This pathname indicates that you wish to move up the directory structure two directories, and then down one to the directory in which JOHN.OCT is located.



Chapter 3

Using the CIP Command Line

CIP commands are used on a CIP command line or in a file. This chapter tells you about the command line. Chapter 9 tells you about parameter files and Chapter 10 about command files.

The CIP assumes that anything typed onto the command line is to be executed. This chapter tells you how to type things onto the command line correctly, so the CIP can understand what you want done.

Command-line Syntax

The information in this section pertains to the CIP command line generally. Chapter 6 details each command.

1. The CIP executes a command-line character string precisely as the character string appears on your terminal screen. The command-line character string is the string of characters you type to execute a command, and is read and executed from left to right. For example:

   ```
   print x25.doc
   ```

2. The character limit for the CIP command line is 1023.

3. You can have any number of commands on a command line as long as the line does not exceed 1023 characters.

4. The command mnemonic is always the first element of a command-line character string, and consists of from 1 to 9 letters.

5. Parameters and switches delimit the execution of a command.

6. Put at least one space between the mnemonic and the first parameter, each parameter value, each switch, and each I/O redirection character.

7. [RETRN] is the command-line terminator, (i.e., the key you strike when you want the CIP to execute what you have typed onto the command line).

8. Table 3.1 lists characters that the CIP uses for special purposes. Each of these characters is explained in the section of this manual that deals with the function the character performs.

   Other characters such as the asterisk, *, equal sign, =, number sign, #, percent sign, %, etc., are used by some of the CIP commands. That usage is explained in the description of the command that uses the character.

Table 3.1. Significant Characters

Character	Purpose
?	Display a CIP help file on the screen
!	Comment flag
	Parameter separator (space character or blank)
;	Command separator
,	File-designation separator in file lists
@	Create another copy of CIP.EXE for the purpose of executing command files and parameter files
&	Background execution of commands or command files
<	Redirect the standard input of a process
>	Redirect the standard output of a process
>>	Append the standard output to a file
^	Redirect the standard error output of a process
^^	Append the standard error output to a file
|	Pipe
-	Used in relative addressing
"	Group multiple words as a single parameter
'	Translate a string of characters
\	Accept the subsequent characters literally, or input the subsequent binary values
:=	Assign a logical name to the current process
:==	Assign a logical name to the user process
:===	Assign a logical name to all processes on the system
{}	Used to enclose process creation parameters

9. Process creation parameters can be specified on the command line. They are enclosed in braces and can occur in any order on the command line.

Importance of Syntax

A command-line character string is executed exactly as it appears on your screen, reading from left to right. Consequently, the sequence, spelling, and punctuation of the character string are crucial to the execution of the command.

Character Limit

The character limit for the CIP command line is 1023. If you exceed that limit, the keyboard beeps each time you strike a key. The CIP executes only the 1023 characters that are allowed. Because the terminal screen is only 80 characters wide, the CIP moves the terminal window automatically across the command line.

The Semicolon

You can list several commands on a command line by separating each command with a semicolon.

Spaces are allowed on either side of a semicolon.

The command line is executed from left to right. Were the following command line executed, the CD command would be executed before DIR:

```
cd /users.al/;dir
```

Command Limit

You can have any number of commands on a command line as long as the line does not exceed 1023 characters.

The Command Mnemonic

The CIP uses the command mnemonic to identify the program you wish to execute. For example, when you type DIR onto the command line, you indicate to the CIP that you want to execute the DIR Program.

If you type a command for which the CIP cannot find a program, a diagnostic message appears on the screen. For example, if you type DDIR onto the CIP command line, the following diagnostic message appears on the screen:

```
CIP      : Creating process DDIR.EXE
Failed   : Status = 133.
Message  : The specified file could not be found.
```

In other words, CIP read the first element of the character string (DDIR) as the command mnemonic. The CIP searched for a file named DDIR.EXE, failed to find it, and displayed a diagnostic message explaining:

a. What the utility was doing (creating process) when the error occurred.

b. A diagnostic number indicating that the CIP failed in executing the command (status=133).

c. A statement describing the error. In this case, the CIP was unable to execute the command because DDIR.EXE does not exist.

The Return Key

[RETRN] is the command-line terminator, (i.e., the key you strike to tell the CIP to execute what you typed onto the command line).

Significant Characters

Some of the significant characters in the CIP have precedence over others. The order of precedence is:

```
\
?
'
"
!
all others
```

In other words, a question mark has precedence over every other character except the accept character, and so on. Precedence refers to which character will be accepted if two or more characters appear together. Note that all the significant characters other than those listed above have approximately the same precedence.

For example, suppose you want to use the SEND command to communicate with another terminal. To do so you type send on to the command line with the message you wish to send enclosed in double quotation marks. For example:

```
> send "Are you ready"
```

The following report appears on the screen of the terminal receiving the message:

```
(CIP_LARRY) Are you ready
```

However, if you type the same message with a question mark at the end of the phrase, as soon as you type the question mark a help display for the SEND command appears on your screen.

Because the question mark has precedence over the double quotation marks, the CIP responds to it. In order to be allowed to send a message with a question mark in it, the question mark would need to be preceded immediately by an accept character. Because the accept character (\) has precedence, the CIP responds to it by accepting what follows.

On the other hand, if you send a message that is enclosed by double quotation marks, and which has an exclamation mark in it, the exclamation mark will be included in the message because double quotation marks take precedence over it. For example:

```
send "Now is the time to go!"
```

The following message appears on the screen of the terminal to which it was sent:

```
(CIP_NEAL) Now is the time to go!
```

The exclamation mark is included in the message because the double quotation marks take precedence over the exclamation mark.

? QUESTION MARK

Unless you type a backslash immediately before the question mark on the CIP command line, a help display appears on your screen. Help displays are discussed later in this chapter.

CIP displays the help file associated with the first command mnemonic on the command line. Therefore, the position of the cursor on the command line is inconsequential when you type a question mark.

! EXCLAMATION POINT

The exclamation point is the comment character used in command files to set off explanations for the user. Characters following the exclamation point on the same line are ignored when the file is executed.

SPACE

A space separates all parameter values, I/O redirection assignments, and other command-line elements.

; SEMICOLON

This character separates consecutive CIP commands.

For example, the following command-line character string deletes REPORT.DOC, and displays a report on the amount of space on the volume in _DX0 (the second command would not be executed without the semicolon):

```
> del report.doc;sp _dx0
```

, COMMA

This character separates the file designations in a file list.

For example, the following command-line character string deletes each of the listed files:

```
del report.doc,summary.txt,inventory.doc
```

@ AT SIGN

This symbol indicates that you want to execute a copy of CIP.EXE and use that copy to execute a command file.

& AMPERSAND

The ampersand indicates background execution of a command or command file. This is explained in more detail in Chapter 9 of this manual.

< LEFT ANGLE BRACKET

This is used to temporarily redirect input. This is explained in more detail in Chapter 10 of this manual.

> RIGHT ANGLE BRACKET

This is used to temporarily redirect output. This is also explained in more detail in Chapter 10 of this manual.

>> DOUBLE RIGHT ANGLE BRACKETS

These are used to append the output of a process to a file. See Chapter 10 for more information.

^ CIRCUMFLEX

This is used to temporarily redirect diagnostic messages. This is explained in more detail in Chapter 10 of this manual.

^^ DOUBLE CIRCUMFLEX

These are used to append the error output of a process to a file. See Chapter 10 for more information.

| VERTICAL BAR

The vertical bar represents a pipe. It is used to redirect the output of one program to the input of another. This is explained in detail in Chapter 10.

- HYPHEN

The hyphen is used in relative addressing.

" DOUBLE QUOTATION MARK

This character is used to group multiple words as a single parameter.

' SINGLE QUOTATION MARK

This character encloses an item to be translated.

\ BACKSLASH

The backslash is used as an accept character. It tells the CIP to accept the next character as a character and not assign any meaning to it (e.g., \? tells the CIP to treat the ? as a question mark and not as an indication that you want to look at a help display).

The accept character is also used to insert binary values. Binary values are specified by giving an accept character followed by 2 hexidecimal digits. The binary value of the hexidecimal sequence is inserted.

:= COLON EQUALS

This sequence is used to assign logical names to the current process. Read Chapter 11 of this manual for more detail.

:== COLON DOUBLE EQUALS

This sequence is used to assign logical names to the user process. Read Chapter 11 of this manual for more detail.

:=== COLON TRIPLE EQUALS

This sequence is used to assign logical names to all processes on the system. Read Chapter 11 of this manual for more detail.

{} BRACES

These are used to enclose process creation parameters. These parameters are discussed later in this chapter.

Process Creation Parameters

When you strike [RETRN] to execute a CIP command, you create a process. The CIP assigns a nodename, process name, privilege, priority, timeslice, owner, and other attributes to this process by default. You can change the default values of your process attributes on the command line with the following process creation parameters. These parameters must be enclosed in braces, {}, and can occur in any order on the CIP command line.

nodename/siteid — The nodename or site ID of the system on which you want the process to run. The default is the node associated with your system disk.

:processname= The process name (up to 16 characters) to be assigned to the process. The default is the command name followed by an underscore and your username.

:privilege= A privilege list (setpriv, system, readphys, writephys, setprior, chngsuper, bypass, operator, altuic, world, group, network, setattr, all, none). The default is the user's own privileges, plus any privileges the process was installed with.

:priority= A priority (number from 0 to 15). The default is the user's current priority.

:timeslice= A timeslice (in milliseconds). The default is the user's current timeslice.

:owner= A UIC or username. The default is that the process is owned by the user who created it.

:attribute= An attribute list (swappable, prezeromem, postzeromem, fastencrypt, desencrypt, watchdog, user1, user2, user3, user4). The default is the user's current attributes.

For example, suppose you typed the following process creation parameters on the CIP command line:

```
{__nodeb :prio=2 :priv=group :owner=[22,2] :time=100 :proc="joe"} dir
```

When you strike [RETRN], the CIP creates DIR.EXE on node __NODEB with priority 2, privilege GROUP, owner [22,2], timeslice 100, and processname JOE.

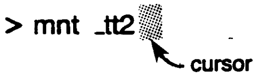
Control-key Functions

One important feature of the CIP is that you can edit command-line character strings before executing them. This saves you time when you make a mistake in typing a command-line character string, or when you execute an erroneous command-line character string and do not want to retype the whole character string.

Use these functions to edit the CIP command line, recall previous CIP command-line character strings, and control various aspects of the execution of commands.

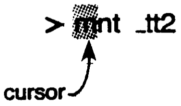
[CTRL] a Moves the cursor to the first position on the command line.

before



```
> mnt _tt2
```

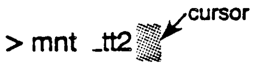
after



```
> mnt _tt2
```

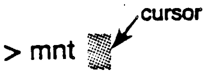
[CTRL] b Deletes the word to the left of the cursor.

before



```
> mnt _tt2
```

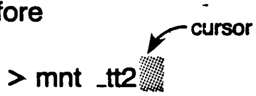
after



```
> mnt
```

[CTRL] c Stops the execution of a program whose input file is the terminal from which [CTRL] c is executed. When you execute [CTRL] c, a circumflex and an uppercase C, ^C, appear on the command line, and the cursor reappears on a new CIP command line. However, the function that is stopped is saved in the list of recently issued commands.

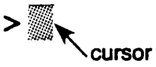
before



```
> mnt _tt2
```

after

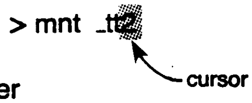
```
> mnt _tt2
```



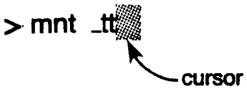
```
>
```

[CTRL] d Deletes all characters from the cursor to the end of the command line.

before


```
> mnt _tt2
```

after

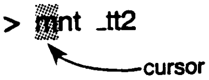

```
> mnt _tt
```

[CTRL] e Allows you to recall a previous command and have it automatically executed by typing only the first characters of a command, followed by [CTRL] e. CIP searches for the most recently issued command that begins with the characters specified. The first command that matches is executed.

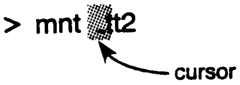
If you type a mnemonic on to the command line that does not correspond to a previous command and then strike [CTRL] e, the terminal will beep, but will not execute a command.

[CTRL] f Moves the cursor to the front of the next word on the command line. [CTRL] f is ineffectual when the cursor is at the end of the command line.

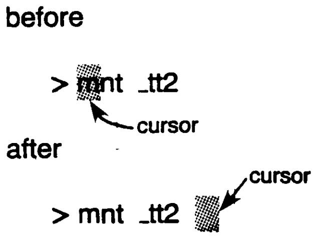
before


```
> mnt _tt2
```

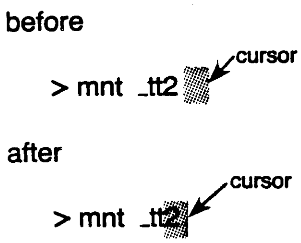
after


```
> mnt _tt2
```

[CTRL] g Moves the cursor to the end of the command line. [CTRL] g is ineffectual when the cursor is already there.



[CTRL] h Moves the cursor one character position to the left. This is the same function that the left arrow key and the backspace key perform. If for some reason the left arrow does not function, use [CTRL] h instead. However, [CTRL] h is ineffectual when the cursor is in the left-most position on the command line.

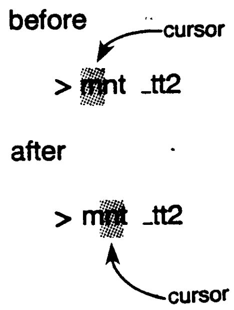


[CTRL] i Functions as the [TAB] key.

[CTRL] j When searching the command list, [CTRL] j is pressed to recall the most recently issued command. If [CTRL] j is pressed again, the CIP searches for the the next command in the list. In other words, [CTRL] j performs as the down-arrow key. However, [CTRL] j is ineffectual if you have not already used [CTRL] n.

[CTRL] k Performs the same function as the up-arrow key. In other words, [CTRL] k recalls the most recently issued command to the command line.

[CTRL] l Moves the cursor one character position to the right. The right-arrow key performs the same function.



[CTRL] m This is equivalent to striking [RETRN].

[CTRL] n Searches for the most recently issued command that begins with the mnemonic specified, within a 2 Kbyte buffer. The first command matched is recalled to the command line. If [CTRL] n is pressed again, the CIP searches for the command with the specified mnemonic prior to the one just recalled.

When no character is on the command line, [CTRL] n performs as [CTRL] k.

[CTRL] o Stops output to the terminal screen, suppressing only the display of output from the process. Strike [CTRL] o a second time to cause the displaying of output to resume.

NOTE: If the process whose output was interrupted concludes before you strike [CTRL] o again, the cursor may appear to be frozen. If this happens, strike [CTRL] o and then [RETRN] The cursor reappears next to a right angle bracket on a new CIP command line.

[CTRL] p When a mnemonic is on the command line, striking [CTRL] p shows the path of the image file or command file to be executed.

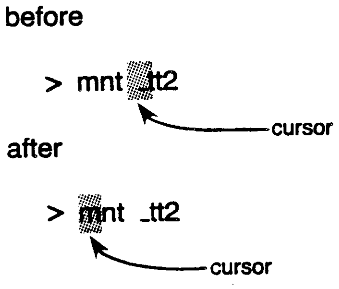
For example, if you type DIR on to the command line and strike [CTRL] p, the following display appears immediately below the command line:

```
sys$disk/sysexe/dir.exe > dir
```

However, if you strike [CTRL] p when no mnemonic is on the line, a new command line appears.

[CTRL] q Resumes output to the screen after [CTRL] s. Unlike [CTRL] o, this function allows you to see all the output even though it has already taken place.

[CTRL] r Moves the cursor to the end of the previous word on the command line. If the cursor is already at the front of the command line, this function is ignored.



before

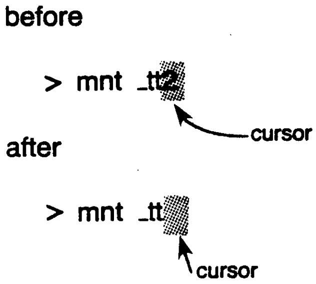
```
> mnt _tt2
```

after

```
> mnt _tt2
```

[CTRL] s Stops output to the screen. Output can be resumed by striking [CTRL] q. Unlike [CTRL] o the command-line character string you have typed on to the command line is not lost.

[CTRL] v Deletes the character or space occupied by the cursor. All characters to the right of the cursor on the command line are, upon the execution of this function, shifted one space toward the left margin of the command line.



before

```
> mnt _tt2
```

after

```
> mnt _tt
```

[CTRL] w When you type a command that is longer than the 80-character width of the terminal screen, the command line shifts in order to type the entire command. When you strike [CTRL] w, the entire command is displayed (even if it is longer than 80 characters), by wrapping the line around to subsequent lines as needed.

[CTRL] y Deletes the current line. Unlike [CTRL] c, which saves the command in the list of recently issued commands, [CTRL] y simply erases the line.

The Keyboard and the CIP

This section explains the manner in which the keyboard behaves while you are in the CIP. Figure 3.2 shows the standard WICAT keyboard.

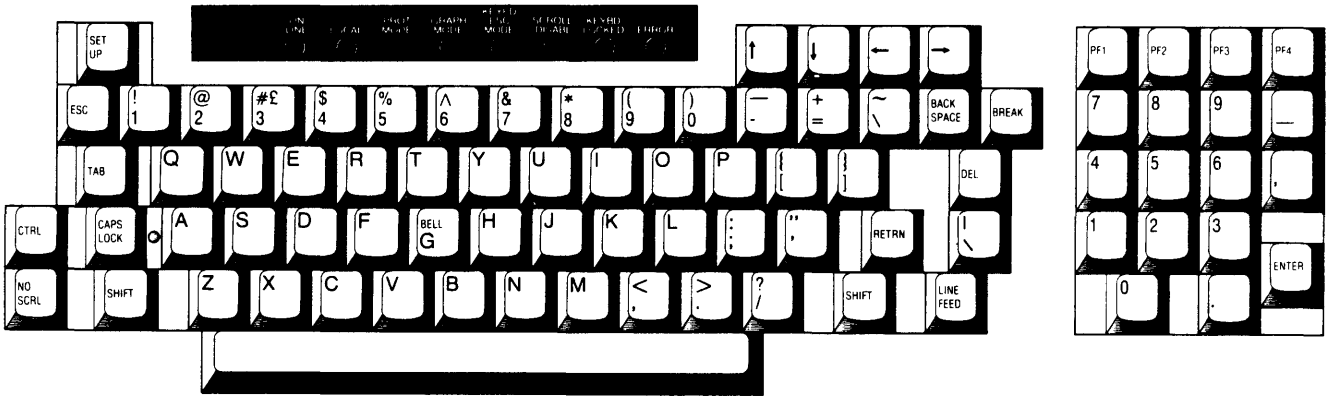


Fig. 3.2 The keyboard

Character Keys

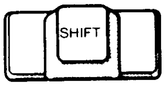
Any character (except the question mark) typed onto the CIP command line appears at the point at which the cursor is located when you type the character, and the cursor appears to the right of the newly typed character.

If there are characters to the right of the cursor when characters are typed onto the command line, or when you strike the spacebar, the character (or space) underneath the cursor, and all characters and spaces to the right of the cursor, shift one space toward the right margin of the command line.

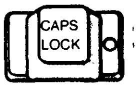
The numeric character keys in the keypad, as well as the dash, comma, and period keys also function as character keys.

Function Keys

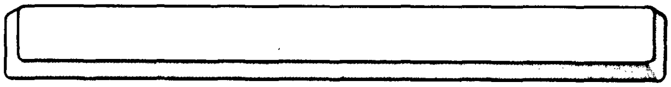
The function keys allow you to move the cursor, execute command line character strings, etc.



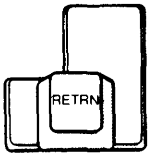
Use the shift key in conjunction with the alphanumeric keys as you would the shift key on a typewriter.



The caps lock key functions only with the alpha-character keys.

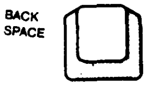


The spacebar moves the cursor toward the right margin of the command line. Any spaces or characters to the right of the cursor are also moved toward the right end of the command line.

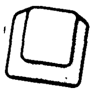


The return key is used to type commands positionally, to tell the CIP that the command string is finished and the command is to be executed. If the command is typed by prompt, the return key tells the CIP that you have finished typing a particular parameter and wish to be prompted for the next parameter. However, when you strike the return key after having typed the last required parameter for the command, CIP executes the command.

NOTE: The cursor can be anywhere on the CIP command line when you strike the return key to execute the command.



The backspace key function is the same as the left-arrow key (see the information on the left-arrow key, below).



The delete key deletes the character immediately to the left of the cursor. If you hold the delete key down, it repeats, that is, continues to delete characters as long as you press the key. However, if the cursor is in the first position on the command line, the delete key has no effect on the cursor. If a character is underneath the cursor, in the first position on the line, and the cursor is over that character, you cannot use the delete key to delete the character. Finally, any characters and/or spaces to the right of the cursor, as well as the character or space underneath the cursor, are moved one space toward the left margin of the command line with each press of the key.

↑

The up-arrow key performs the same function as [CTRL] k.

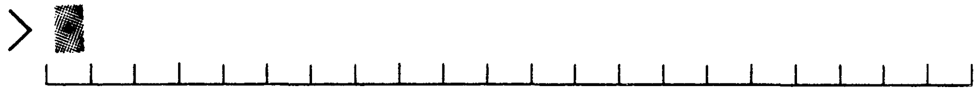
↓

The down-arrow key performs the same function as [CTRL] j.

→

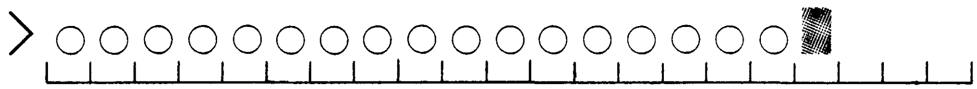
The right-arrow key ([CTRL] l on some terminals) moves the cursor toward the right margin of the command line, one character space at a time. Nevertheless, you cannot use this key to create spaces.

For example, when the cursor appears next to a right angle bracket on the CIP command line, none of the cursor positions on the command line has been assigned either a space or a character. If you think of such a command line as a string of unassigned cursor positions, it might look like this:



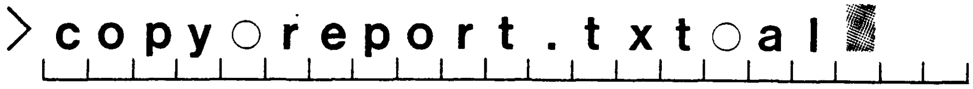
Inasmuch as the right-arrow key moves the cursor only to a position that has been assigned either a space or a character, pressing the right-arrow key will not move the cursor on the above command line.

Were you to use the spacebar to move the cursor, and were a space represented by a circle, the command line would look like this:



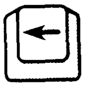
On such a command line, you could use the right-arrow key to move the cursor to any position occupied by a circle.

Finally, any cursor position in which the cursor is located, is assigned a space. Thus were you to type the following command string onto the CIP command line, you could use the left-arrow key to move the cursor toward the left end of the line, and the right-arrow key to move the cursor toward the right end of the line, until the cursor is located in the space immediately to the right of the last character in the line.

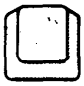


```
>copy○report.txt○al█
```

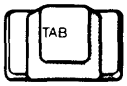
Once the cursor is located in the right-most position on the command line, the right-arrow key is ineffectual.



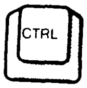
Use the left-arrow key ([CTRL] h on some terminals) to move the cursor toward the left margin of the command line. Once the cursor is located in the first position on the line, the left-arrow key is ineffectual.



Ineffectual in the CIP.

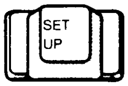


The [TAB] key moves the cursor eight spaces to the right, or to the next tab stop (which is every eight characters).

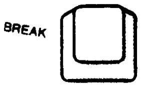


Use the control key to execute control key functions. Do this by pressing [CTRL] and, keeping it depressed, striking the character key assigned to the function you wish to perform.

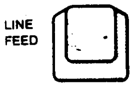
Press this key once to temporarily halt the display of output to the terminal screen. Strike the key a second time to cause the display of output to resume.



Use this key to enter terminal setup mode. This mode is explained in the operator's guide for your system, or in the Terminal Reference Manual.



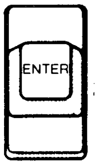
Ineffectual in the CIP.



Functions as [CTRL] j.

Keypad

The foregoing keys are not used by the CIP.



Functions as the return key.

Help Displays

To make your transition from new to experienced user as smooth as possible, and to aid experienced users who want a synopsis of a particular command, the CIP provides a set of help displays that can be called to the screen from the CIP command line by typing a question mark, ?.

If the question mark is the only character you type onto a command line, a display listing all the available help files appears on the screen.

If a command mnemonic begins the command line and you type a question mark, the help display for that command appears on the screen, and your command line character string reappears below the help display. This allows you to finish typing the command while you refer to the help display.

Chapter 4

Parameters and Switches

In most cases, the mnemonic for a CIP command is followed by instructions that delimit the execution of the command. There are three kinds of instructions, or delimitations:

1. Required parameters.

2. Optional parameters.

3. Switches.

Required Parameters

A required parameter has no default value. Therefore, the command cannot be executed until a value is specified for all the required parameters that delimit the execution of the command. For example, the CHKD Command that checks the sectors on a disk has one required parameter--the devicename for the disk whose sectors you want checked. CHKD cannot be executed until a devicename is supplied for that required parameter.

The Command Line Syntax section in each CIP command description (in chapter 6 of this manual) tells you whether a command has any required parameters, and the order in which parameter values for required parameters are to be typed onto the command line. Note that the CIP reads the command line from left to right and that the values for required parameters must appear immediately after the command mnemonic in the order in which they are listed under the Command Line Syntax section in the description of the command.

If you type a command mnemonic for a command that has required parameters, but do not type values for the required parameters, a prompt appears on the screen when you strike [RETRN]. For example, were you to type DEL onto the command line and strike [RETRN], this is what would appear on your screen:

```
del
File list     >
```

"File list" is a prompt that tells you DEL cannot be executed until you specify what you want DEL to delete. Note that were you to strike [RETRN] without having typed a value for the required parameter the prompt would reappear; and continue to reappear each time you struck [RETRN] without having typed a parameter value.

If you decide you do not want to execute a command, and you are specifying parameter values as you are prompted for them, use [CTRL] c.

Finally, suppose that you type the following onto the command line (:CONFIRM is what is called a switch; these are explained later in this chapter):

```
del :confirm
```

This is what your screen looks like when you strike [RETRN]:

```
del :confirm
File list     >
```

In other words, even though a value appears next to the command mnemonic, DEL recognizes (because in this case the value begins with a colon, indicating that the value is a switch) that no value is specified for the required parameter. When you specify the required value, DEL is executed and the switch is included as though it and the required parameter value has been specified in the first place.

Prompts for required parameter values are provided so that the transition from new to experienced user can be as smooth as possible.

Finally, even though the prompt is for a single required parameter value, you can type not only the required parameter value but any other parameter values and switches you wish to include on the command line. For example:

```
del :confirm
File list     > ch01.txt :auto
```

There are four wildcard symbols:

*   The asterisk is a multiple-character wildcard that can be used to specify zero or more characters in a particular file designation.

For example, DEL REP*.*.* indicates that you wish to delete all files that match the specified pattern. In other words, all files with R E P as the first three characters in the filename are to be deleted, regardless of what follows R E P in the filename, and regardless of the file extension.

=   The equal sign is a single-character wildcard symbol that can be used to specify any single character.

For example, ST= might stand for STA, STB, ST1, etc., where ST is followed by one character. =E* could stand for anything where E is the second letter in a character string.

[]   The set of characters specified within the square brackets will match any single character.

A set, or range, can also be specified as in the following example:

```
del report.doc.[1-46-9]
```

In the foregoing example, versions 1, 2, 3, 4, 6, 7, 8, and 9 of the file report.doc will be deleted. Version 5 will not be deleted.

One or more sub-ranges can even be specified. For example:

```
del [a-dmx-z].txt.*
```

The foregoing example will delete all versions of files which begin with the letters a, b, c, d, m, x, y, and z that have the extension txt.

You can also exclude characters from a set or subset as shown in the following example:

```
del [a-m^f].txt.*
```

The foregoing command-line character string allows you to delete all versions of files that begin with the letters a through m excluding f. Files beginning with optional parameters. If you wish to specify a value for the second optional parameter, you must also specify a value for the first.

Switches

Switches also delimit command execution in the same manner as required and optional parameters. However, they can appear anywhere on the command line following the command mnemonic. The order in which switches appear on the command line is also immaterial.

All switches are optional (i.e., a default value has been assigned to every switch).

Any switch can be abbreviated as long as the abbreviation cannot be confused with any other switch.

These are the three kinds of switches:

1. Boolean switches.

2. Valued switches.

3. Set switches.

A **boolean switch** is a delimitation that is either on or off; that is, the delimitation that the switch represents is either enabled or disabled. For. example, the following command-line character string indicates that you want to delete RANDOM.TXT from the default directory:

```
del random.txt
```

When you strike [RETRN], this kind of prompt appears:

```
Delete (Y or N)? >
```

If you want RANDOM.TXT deleted as soon as you strike [RETRN] (i.e., you do not want to have to confirm your request to delete the file), you would type the following kind of command-line character string:

```
del random.txt :auto
```

The :AUTO Switch indicates that you want RANDOM.TXT deleted as soon as you strike [RETRN]. The switch is boolean (e.g., there are only two values you can type for the switch: :AUTO or :NOAUTO. The first of these enables the delimitation represented by the switch, the second disables the delimitation. Therefore, for any boolean switch type the the positive form of the switch to enable the delimitation and the negative (type NO followed by what you would type for the positive) to disable the delimitation.

This character string tells the CIP to delete all files (*) with an extension of .DOC. Nevertheless, the foregoing character string would allow you to delete only the most recent version of each file. To delete all versions of those files with a .DOC extension, type the following character string:

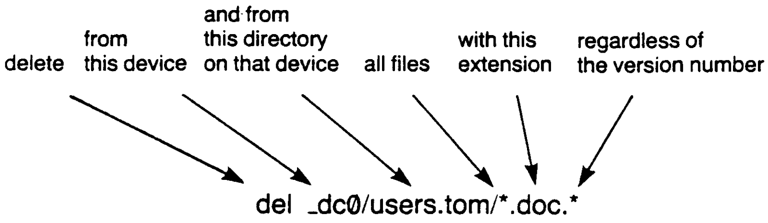
```
del *.doc.*
```

This indicates that all versions(*), of all files(*.) with an extension of .DOC are to be deleted.

If you want to delete all files, from /USERS.TOM/, that have a .DOC file extension, you could do so with the following character string:

```
del _dc0/users.tom/*.doc.*
```

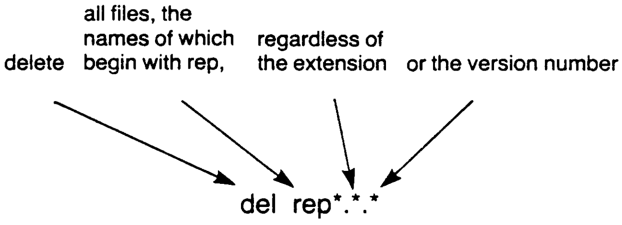
In other words, you are telling the CIP to:


delete
from this device
and from this directory on that device
all files
with this extension
regardless of the version number
del _dc0/users.tom/*.doc.*

In addition, suppose that you want to delete all files from a particular directory whose filename begins with REP. The following character string would allow you to do so with a single command:

```
del rep*.*.*
```

This tells the CIP that you wish to


delete
all files, the names of which begin with rep,
regardless of the extension
or the version number
del rep*.*.*

Specifying More Than One File Designation in a Parameter

As mentioned above, many commands require a specified parameter value. The DEL Command requires that you specify the file to be deleted, the COPY Command requires that you specify not only the file to be copied, but the destination as well.

Chapter 2 explained the file designation, and the syntax used to specify a device, directory, or file as a parameter value.

The following section tells you how to designate more than one device, directory, or file for a parameter. Two methods exist for specifying more than one file designation in a parameter value:

1. File lists.

2. Wildcard, or pattern matching, symbols.

These two methods can also be combined.

File Lists

While parameters must be separated by one or more spaces, a series of parameter values can be designated, in a single parameter value, with commas. For example:

```
del report1.txt,report2.txt,report3.txt
```

File lists can be used in all command parameters that call for a file list.

Wildcard, or Pattern Matching, Symbols

When you type a **wildcard symbol** as part of a file designation, you indicate that any character, or group of characters, that correspond to the position of the symbol in the file designation will be included in the execution of the command.

Wildcard symbols can be used in any field of the file designation, including the nodename, devicename, and directory name fields.

:sort= Specifies whether to sort and which attribute to sort on. If no switch value is specified, it will sort the default. The accepted values for this switch are:

none - do not sort at all
name - sort on filename
date - sort on creation or modification date
ext - sort on extension
uic - sort on uic

the letter f are excluded because it is preceded by the exclusion character ^.

() Numbers as wildcards inside parentheses makes it possible to select more than single digit numbers. For instance, while the brackets [] allow the selection of the numbers 1 through 9, they do not allow the use of a wildcard for numbers 1 through 10. The use of parentheses () allows the use of a wildcard for 1 through any number, as well as for numbers such as 002, 092, etc.

For example, if you want to delete versions 1 through 19 of file report.doc, you need only type the following:

```
del report.doc.(1-19)
```

Ranges are also allowed inside parentheses. For instance, to delete selected files from a series, 0 to 10, you should type:

```
del (0-4,6,8-10)
```

This command deletes files 0,1,2,3,4, 6,8,9, and 10.

However, if you attempt to use the parentheses incorrectly, the CIP displays the following diagnostic message:

```
No files found.
```

This tells you only that you must repeat the command correctly and does not mean that your files have disappeared.

Use the wildcard symbols in any parameter value where you want to specify more than one file. For example, suppose that you want to delete all the extra copies of all the files in a particular directory, and that the copies you wish to delete have .DOC as a file extension. Instead of deleting the files one at a time, you can type the following character string onto the command line:

```
del *.doc
```

Chapter 5

Directory of CIP Commands

The commands in this directory are divided into two sections. The first section consists of frequently used programs that the novice should learn first. The second section consists of the CIP commands grouped according to function. The commands in the first section will also appear in the appropriate group of the second section.

Initial Commands

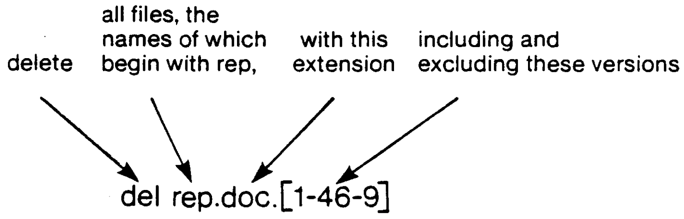
New WMCS users should become familiar with the following CIP commands because they are used more frequently than other commands:

cd	Change default directory
copy	Copy files
crd	Create directories
def	List default device and directory
del	Delete files
dir	List directory contents
err	Display a diagnostic message
log	Log off
print	Print files
pu	Purge a directory
ren	Rename files
type	Display the contents of text files
vew	Edit text files

Finally, suppose you wanted to delete only some of the files named REP.DOC in a particular directory. If you have versions REP.DOC.1, REP.DOC.2, REP.DOC.3, etc., through REP.DOC.9, and you want to delete all versions except REP.DOC.5, you would type the following character string:

```
del REP.DOC.[1-46-9]
```

This tells the CIP that you want to


delete
all files, the names of which begin with rep,
with this extension
including and excluding these versions
del rep.doc.[1-46-9]

Wildcard symbols are allowed with the following switches:

:exclude= This is used for a list of file specifications that are not to be included even though they may match a file list.

:mod Any dates specified will check the modification date rather than the creation date.

:since= Specifies a date and time. Only those files which were modified or created on or after the specified date and time will be wildcarded. All dates will be used if not specified.

:before= Also specifies a date and time. Only those files modified or created before the specified date and time will be wildcarded. All dates will be used if not specified.

:uic= Specifies the owner of a file. Only those files which match the specified user and group portion of the UIC will be wildcarded. Both the group and user portions of UIC can be any valid wildcard pattern. A valid username can also be specified. This switch is used predominantly by the system manager. Detailed information about its use is in the System Manager's Reference Manual.

Directory Commands

cd	Change default directory and/or device
crd	Create a directory
dir	List contents of directory

Device Commands

alloc	Allocate a device
btup	Update the boot block
chkd	Check disk sectors
dealloc	Deallocate a device
dev	List mounted devices
dinit	Initialize device
dmnt	Dismount a device
mnt	Mount a device
recover	Rebuild system files
rew	Rewind tape
rotor	Create and manager rotor lists
send	Display a message on other terminals
skip	Position the tape
swap	Exchange the system disk
xfer	Transfer disks or files

Status Commands

cmdst	Display previous command line character strings
def	Display name of default device and directory
dm	Display memory status
dstat	Display and change device status
err	Display an error message
fstat	Display and change file status
mstat	Name shared memory status
nstat	Display network status
pstat	Display and change process status
serial	Display hardware serial number
sp	Display report on space allocation on a device
time	Display and set system time
who	Display usernames and UICs

Running Programs

cip	Create a CIP
option	Specify a CIP option
submit	Submit jobs to the batch queue

Miscellanea

dispatch	Send the output of a process to several destinations
gprint	Print screen graphics on a printer
logline	Append log message to end of log file
memtest	Test memory
prompt	Prompt user for input
shutdown	Prepare system for power shutdown
swapper	Remove inactive processes from system memory
typeb	Display lines in block letters
wait	Wait until specified time
watchdog	Logoff unused interactive processes

CIP Commands

The remainder of this chapter is a directory of CIP commands grouped by function.

Accessing the System

log	Log off
logflush	Initiate logon, write disk buffers to disk
logon	Log on to the system
nuserprof	Edit Network user authorization file(s)
password	Change user password
userprof	Edit user authorization file(s)

Logical Name Assignment

:=	Logical name assignment for current process
:==	Logical name assignment for user process
:===	Logical name assignment for system
shlog	Display logical name assignments

Editing

vew	Edit a text file
zap	Inspect and change file/device contents

File Commands

arch	Archive
copy	Copy files
count	Count words, lines, characters in files
create	Create files
crypt	Encrypt/decrypt files
del	Delete files
dump	Display file contents in hexadecimal
keygen	Generate keys for encryption usage
print	Print files
pu	Purge old versions of files
ren	Rename files
scan	Search files for a pattern
sort	Sort a file (old version)
tcopy	Copy files to/from foreign tape
translit	Character transliteration
type	Display contents of text files
wscan	Search files for a wildcard pattern
wsort	Sort a file

Chapter 6

Dictionary of CIP Commands

This chapter contains a detailed explanation of each CIP command.

The command descriptions are presented in alphabetical order (according to the mnemonic for each command).

Program Development

dumpdiff	Display differences between two files
incl	Merge text files
link	Linkage editor
makedsr	Convert an image to a driver
peek	Examine physical or logical memory
poke	Modify physical or logical memory
prelink	Prepare libraries for linking
typemrl	Display linker information records from .MRL file
wibug	Symbolic debugger

Backup and Restore

backup	Incremental system backup
bkup	Create a backup of a software product
load	Load a software produce generated with bkup
restore	Selective file restoration

Communication

nsp	Network surrogate process
sscopy	System-to-system file transfer (old version)
talkt	Communicate through a specified port
usscopy	Updated system-to-system file transfer

Configuration Control

checksum	Compute file checksums
config	System configuration
fpmgr	Install system floating point packages
install	Install privileged files
nsysprof	Network system profile program
sysprof	System profile program
verify	Verify file checksums
version	Display the version string of a file

alloc

Functional Description

Use this command to allocate (or reserve) an available device for exclusive use by a process.

Command Line Syntax

Mnemonic	alloc			
Required parameter	Device or Rotor-list Name (Optional when Function is :list)			
Optional parameter	Function (:allocate, :list)			
Switches				
Function modification	:access=	:logical=	:pid=	:timeout=

Parameters

Device Name	Function	Required. Use this parameter to specify the devicename or rotor list from which an available device will be allocated.
	Default	None. (If the Function parameter is :list, the default is all mounted devices.)
	Syntax	Type any devicename or rotor-list name.
Function	Function	Optional. Use this parameter to specify whether to allocate the specified device (:allocate) or to list all allocated devices (:list).

Examples

```
> alloc _tt2
```

This command attempts to allocate _TT2 for the use of the process (as well as any of its child processes) making this request. A display like the following appears on your screen if the device _TT2 is available:

```
    _TT2 allocated to CIP_GRACE 00030042.
```

```
> alloc modemlst :logical=my_modem :access=W :timeout=300
```

This command attempts to gain exclusive write access to a modem port in the rotor list MODEMLST. If no devices in MODEMLST are immediately available for write access, it means all devices are either write protected to this process or are not currently free. The :timeout= switch tells ALLOC to wait for up to 5 minutes (300 seconds) for a modem port to become free. When a port is finally allocated, the name of the device is placed in the logical name MY_MODEM for future reference.

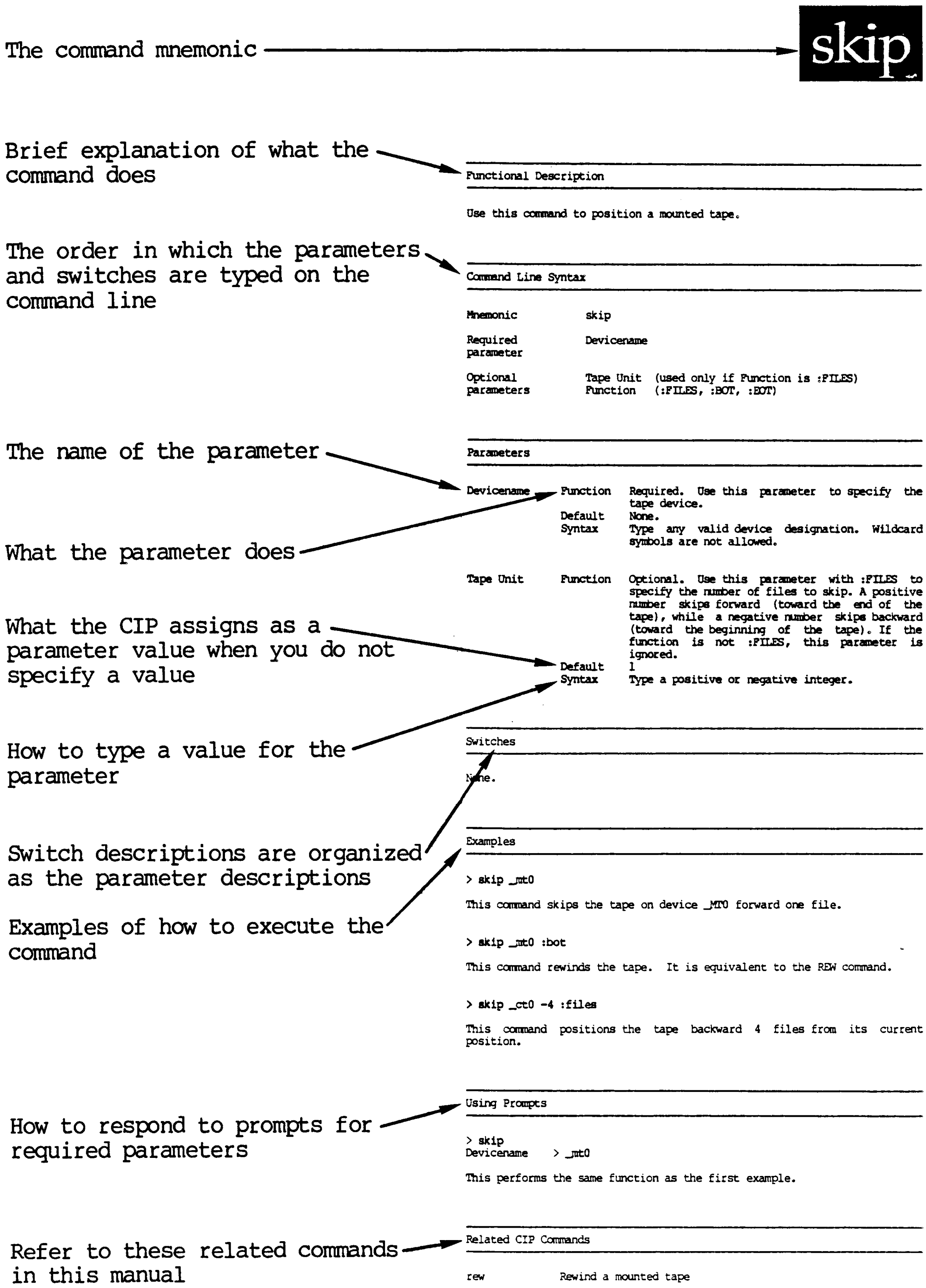
In the absence of a :pid= switch the device is allocated to the parent process of ALLOC, probably CIP.

```
> alloc :list :pid=00030042
```

This command lists all allocated devices that are accessible to process 00030042. A display like the following appears on your screen:

```
       PID       Process name     Allocated devices
    --------   ---------------   -----------------------------------------------
    00030042   CIP_GRACE          _TT2
```

The process to which each device is allocated is also listed. Process names are unsupported.



The command mnemonic

skip

Brief explanation of what the command does

Functional Description

Use this command to position a mounted tape.

The order in which the parameters and switches are typed on the command line

Command Line Syntax

Mnemonic	skip
Required parameter	Devicename
Optional parameters	Tape Unit (used only if Function is :FILES) Function (:FILES, :BOT, :EOT)

The name of the parameter

What the parameter does

What the CIP assigns as a parameter value when you do not specify a value

How to type a value for the parameter

Parameters

Devicename	Function	Required. Use this parameter to specify the tape device.
	Default	None.
	Syntax	Type any valid device designation. Wildcard symbols are not allowed.
Tape Unit	Function	Optional. Use this parameter with :FILES to specify the number of files to skip. A positive number skips forward (toward the end of the tape), while a negative number skips backward (toward the beginning of the tape). If the function is not :FILES, this parameter is ignored.
	Default	1
	Syntax	Type a positive or negative integer.

Switch descriptions are organized as the parameter descriptions

Switches

None.

Examples of how to execute the command

Examples

```
> skip _mt0
```

This command skips the tape on device _MT0 forward one file.

```
> skip _mt0 :bot
```

This command rewinds the tape. It is equivalent to the REW command.

```
> skip _ct0 -4 :files
```

This command positions the tape backward 4 files from its current position.

How to respond to prompts for required parameters

Using Prompts

```
> skip
Devicename    > _mt0
```

This performs the same function as the first example.

Refer to these related commands in this manual

Related CIP Commands

rew	Rewind a mounted tape

arch

Functional Description

Use this command to insert, replace, delete, extract, or list files in an archive file.

Command Line Syntax

Mnemonic	arch			
Required parameters	Archive File			
	File List	(Optional if Function is :list)		
	Destination	(Allowed only if Function is :extract)		
Optional parameter	Function	(:new, :insert, :replace, :delete, :extract, or :list)		
Switches				
Files to be included in the function	:abefore= :aexclude= :amod :asince=	:auic= :auto :before= :exclude=	:mod :override :since=	:arc :confirm :uic=
Content of the display	:full	:log	:bytes	
Format of the display	:pause			
Interaction with archive media	:init= :preinit	:volume= :openfiles=	:rewind	:mount=
Creation of extracted files	:builddir :edit=	:owner=	:protection=	:preserve=

Default :allocate
Syntax Type **:list**

Switches

:access=

Function Use this switch to specify the intended access to the allocated device.
Default :access=RW.
Syntax Type **:access=** followed by one of the following:

R to indicate read access, W to indicate write access, or RW to indicate read/write access.

:logical=

Function Use this switch to assign the name of the allocated device to a logical name instead of displaying it on sys$output.
Default The name of the allocated device is displayed on sys$output.
Syntax Type **:logical=** followed by a logical name.

:pid=

Function Use this switch to specify the process to which the device will be allocated. (If the Function parameter is :list, all allocated devices which belong to this PID, or any of its ancestors, will be displayed.)
Default :pid=-1 (the parent process of the utility)
Syntax Type **:pid=** followed by any valid Process Identification Number.

:timeout=

Function Use this switch to specify (in seconds) how long to wait for the requested device to become available before reporting an error.
Default :timeout=0
Syntax Type **:timeout=** followed by the number of
seconds.

Syntax Type a single file designation. Wildcard symbols are allowed in the directory name, filename, file extension, and version number fields of the file designation. However, the asterisk must be the only character in the field in which it appears.

Function

Function Optional. Use this parameter to specify what you want to do to the archive file. Six Functions are available:

:insert Insert the files specified by the File List parameter into the archive file. If the target archive file does not already exist, it is created.

NOTE: If the archive file is on a tape, ARCH always creates a new archive file, even though an archive file of the same designation already may exist.

:replace Replace the files already in the archive file (whose names match those specified by the File List parameter) with the files specified by the File List parameter. If the File List parameter contains a filename that does not match a filename in the archive file, the specified file is inserted in the archive file as though :insert had been specified.

:delete Delete from the archive file the files that match the File List parameter.

:extract Extract from the archive file the files that match the File List parameter.

:list List the contents of the archive file.

:new Create a new version of the archive file, even if the specified archive file already exists, and insert the files specified in the File List parameter into the new archive file.

Default :list
Syntax Type one of the six Functions.

Using Prompts

```
> alloc
Device name   > _tt2
```

This performs the same function as the first example above.

If the Function parameter is :list, no prompts appear.

Notes on Usage

ALLOC is typically used with rotor lists to select a device based on its intended use and availability. Hence, a pool of similar devices (a set of modem lines, a set of printer lines, etc.) are often listed together in a rotor list and ALLOC is used to select a free device from among those named.

A device can be allocated only when the device is not being referenced by any other process and when the process to which it will be allocated has the desired access to the device. Once a device is allocated, processes which are not among its child processes are prohibited from opening the device or altering it in any way.

Related CIP Commands

dealloc	Deallocate a device
rotor	Add, delete, or list devices in a rotor list

:asince=

Function Use this switch to select files from the Archive File list that were created, modified, or archived since the specified date and time.

Default All files that match the Archive File list are selected.

Syntax Type **:asince=** followed by the date or time or both.

Use this switch only with the :delete, :extract, or :list Functions.

:auic=

Function Use this switch to select files from the Archive File list that are owned by the specified user.

Default All files that match the Archive File list are selected.

Syntax Type **:auic=** followed by a UIC or the username of an authorized system user.

Use this switch only with the :delete, :extract, or :list Functions.

:auto

Function Use this switch to perform the desired operation without user confirmation.

Default :noauto, i.e. operator confirmation is required before the operation is performed.

Syntax Type **:auto**

Use this switch only with the :delete Function.

:before=

Function Use this switch to select files that match the File List and were created or modified before the specified date and time.

Default All files that match the File List are selected.

Syntax Type **:before=** followed by a date or time or both.

Use this switch with any of the six ARCH Functions. If you use the :insert, :replace, or :new Functions, this switch applies to the File List parameter. If you use the :delete, :extract, or :list Functions, this switch applies to the Archive File parameter.

:builddir

Function Use this switch to create destination directories automatically if they do not exist.

Creation of archive file	:space=
Verification of files	:verify

Parameters

Archive File

Function: Required. Use this parameter to specify the target archive file or files.

NOTE: If the archive file is to be placed on a device other than the default device and the :preinit switch is not used, you must mount the device before you execute ARCH. The directory path that contains the archive file will be created, if it does not already exist.

Default: None.

Syntax: If you use the :new, :insert, or :replace Functions, you must type a single file designation. Wildcard symbols are not allowed.

If you use the :delete, :extract, or :list Functions, type a filename or a file list. Wildcard symbols are allowed.

File List

Function: Required unless Function is :list. Use this parameter to specify the file(s) to be inserted into the target archive file, to replace files already in the target archive file, to be deleted from the archive file, to be extracted from the archive file, or to appear in a list of the contents of the archive file.

Default: If Function is :list, then default is *; otherwise no default.

Syntax: Type a filename or a file list. Wildcard symbols are allowed.

Destination

Function: Required if Function is :extract; not allowed otherwise. Use this parameter to specify the destination for the files to be extracted.

Default: None.

Use this switch with any of the six ARCH Functions. If you use the :insert, :replace, or :new Functions, this switch applies to the File List parameter. If you use the :delete, :extract, or :list Functions, this switch applies to the Archive File parameter.

:full

Function Use this switch to display all information about the file.
Default :nofull
Syntax Type :full

Use this switch only with the :list Function.

:init=

Function Use this switch to specify the parameters that are used when media are initialized. (This differs from :preinit, which is used to initialize the first volume, not to specify parameters.) In other words, when you specify :init=, you are not prompted for the parameters under which a diskette or tape is initialized.

This switch does not mean that media are automatically initialized. It allows you to specify (when ARCH is executed) what parameters are to be used when a diskette or tape is initialized. Thus, for each volume, you are prompted whether or not to initialize it. If you decide to initialize it (and you specified :init=) you are not prompted for initialization parameters.

For example, the following value for this switch provides for the automatic initialization of all volumes required for the execution of a particular version of ARCH:

```
:init="dinit dx0 arch_1"
```

Read the DINIT command description for details on what to specify.

Use two number signs as part of the volume label portion of the :init= character string in order to allow the incremental assignment of volume numbers to each diskette. For example, the following character string produces the labels shown for each volume:

Switches

:abefore= Function Use this switch to select files from the Archive File list that were created or archived before the specified date and time.

Default All files that match the Archive File list are selected.

Syntax Type **:abefore=** followed by the date and time or both.

Use this switch only with the :delete, :extract, or :list Functions.

:aexclude= Function Use this switch to exclude files from the Archive File list.

Default All files that match the Archive File list are selected.

Syntax Type **:aexclude=** followed by a list of file designations separated by commas. Wildcard symbols are allowed.

Use this switch only with the :delete, :extract, or :list Functions.

:amod Function Use this switch to specify that the modification date is to be used in all date and time considerations by the :abefore= or :asince= switches.

Default :noamod

Syntax Type **:amod**

Use this switch only with the :delete, :extract, or :list Functions.

:arc Function Use this switch to specify that the archival date is to be used in all date and time considerations by the :abefore= or :asince= switches.

Default :noarc

Syntax Type **:arc**

Use this switch only with the :delete, :extract, or :list Functions.

:openfiles=
Function Use this switch to specify how many files to keep open simultaneously.
Default :openfiles=100
Syntax Type **:openfiles=** followed by a number from 1 to 200.

:override
Function Use this switch to override the requirement that ARCH works only on archive-type files.
Default :nooverride
Syntax Type **:override**

Use this switch only with the :delete, :extract, or :list Functions.

:owner=
Function Use this switch to change the owner (UIC) of the file.
Default The UIC assigned when the file was created or last modified.
Syntax Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC that they represent.

Use this switch only with the :extract Function.

:pause
Function Use this switch to stop the display after each screen of information. The display resumes after the user presses any key.
Default The value specified by the OPTION Command.
Syntax Type **:pause** or **:nopause** to override the default.

Use this switch with any of the six ARCH Functions.

:preinit
Function Use this switch to automatically initialize the first archive volume.
Default :nopreinit
Syntax Type **:preinit**

Use this switch only with the :insert and :new Functions.

:preserve=
Function Use this switch to preserve certain attributes of the original file when a new copy of it is created. The preservable aspects are: creation date, owner, protection.

Default :nobuilddir, i.e. the destination directories are not created and an error is reported if they do not exist.

Syntax Type **:builddir**

Use this switch only with the :extract Function.

:bytes

Function Use this switch to display file sizes in bytes.

Default :nobytes, i.e., file sizes are displayed in tenths of a kilobyte.

Syntax Type **:bytes**

Use this switch only with the :list Function.

:confirm

Function Use this switch to confirm or deny the desired operation on each file.

Default :noconfirm, i.e., the operation is performed without user confirmation.

Syntax Type **:confirm**

Use this switch with any of the six ARCH Functions.

:edit=

Function Use this switch to specify a series of edits to be performed on the Destination file. Edits are specified as a series of "find and switch" strings. The Destination file is searched for the first occurrence of the "find" string. If it is found, it is replaced by the "switch" string.

Default No edits are performed on the Destination file.

Syntax Type **:edit=** followed by a list of "find and switch" strings separated by commas. The find string is separated from the switch string by a colon. One or both of the strings may be empty.

Use this switch only with the :extract Function.

:exclude=

Function Use this switch to exclude files from the File List parameter or from the Archive File parameter.

Default All files that match the list are selected.

Syntax Type **:exclude=** followed by a list of file designations separated by commas. Wildcard symbols are allowed.

If you use the :delete, :extract, or :list Functions, this switch applies to the Archive File parameter.

:space=

Function Use this switch to reserve space within the directory portion of an archive file for future insertion of files.
Default :space=48
Syntax Type :space= followed by a decimal.

NOTE: The value you type represents the number of files for which space will be reserved. Each reserved space will use 64 bytes within the archive file. After an archive file is created, the amount of space used for directory entries cannot be changed. If it is probable that you will need to insert files at a later date, then you must ensure that there will be space for these insertions at the time the archive file is created.

Use this switch only with the :new and :insert Functions.

:uic=

Function Use this switch to select files that match the File List parameter or the Archive File parameter and are owned by the specified user or list of users.
Default All files that match the list are selected.
Syntax Type :uic= followed by a list of UICs or usernames.

Use this switch with any of the six ARCH Functions. If you use the :insert, :replace, or :new Functions, this switch applies to the File List parameter.

If you use the :delete, :extract, or :list Functions, this switch applies to the Archive File parameter.

:verify

Function Use this switch to report whether the command line character string has been correctly executed.
Default :noverify
Syntax Type :verify

Use this switch only with the :insert, :replace, :extract, and :new Functions.

```
:init="dinit dx0 2/feb/83_##"
```

These are the kind of labels produced by the foregoing character string:

```
2/feb/83_1, 2/feb/83_2, ...
```

Default :init=dinit, i.e., DINIT will prompt for the needed parameters if a volume is initialized.
Syntax Type **:init=** followed by a DINIT command string enclosed in double quotation marks.

Use this switch only with the :insert and :new Functions.

:log

Function Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
Default The value specified by the OPTION command.
Syntax Type **:nolog** or **:log** to override the default.

Use this switch with any of the six ARCH Functions.

:mod

Function Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
Default :nomod, i.e. the creation date is used in all date and time considerations by the :before= or :since= switches.
Syntax Type **:mod**

Use this switch with any of the six ARCH Functions. If you use the :insert, :replace, or :new Functions, this switch applies to the File List parameter. If you use the :delete, :extract, or :list Functions, this switch applies to the Archive File parameter.

:mount=

Function Use this switch to mount the archive media.
Default The parameters specified in the DEVCONFIG.xxx file are used.
Syntax Type **:mount=** followed by a MNT command string enclosed in double quotation marks.

Use this switch only with the :new, :insert, or :replace Functions. This switch is most often used in conjunction with the :init switch to specify how to mount the volumes after initializing them in a multi-volume archive.

for the first available, contiguous 38 kilobytes, and therefore assigns ARCH.PAS to the space formerly occupied by ARCH.DOC. Now ARCH.PAS starts where ARCH.DOC used to, and has 56 kilobytes allocated to it (the same space allocated to ARCH.DOC) even though its actual size is only 38 kilobytes.

The Archived column lists the date and time the file was archived.

> arch archive.arc arch.* :new :verify

This command generates the following display:

```
        1  ARCH.EXE.1              entered.
        2  ARCH.HLP.5              entered.
        3  ARCH.MCR.2              entered.
        4  ARCH.MRL.3              entered.
        5  ARCH.PAS.4              entered.
        6  ARCH.PRN.5              entered.
     ARCH.EXE.1              verified.
     ARCH.HLP.5              verified.
     ARCH.MCR.2              verified.
     ARCH.MRL.3              verified.
     ARCH.PAS.4              verified.
     ARCH.PRN.5              verified.
```

Lines 1-6 begin with the entry number of the file in the newly created archive file. The entry number shows the order the files are entered into the archive file. This is followed by the filename and the word "entered." Following these six lines are six more which indicate whether or not each file is verified. A file is verified when ARCH compares the original to the archived copy and the two files agree.

> arch archive.arc * * :extract :verify

This command generates the following display:

```
     ARCH.EXE.1          extracted as _DS0/USERS.WORKBENCH.PROJECTS/ARCH.EXE
     ARCH.HLP.5          extracted as _DS0/USERS.WORKBENCH.PROJECTS/ARCH.HLP
     ARCH.MCR.2          extracted as _DS0/USERS.WORKBENCH.PROJECTS/ARCH.MCR
     ARCH.MRL.3          extracted as _DS0/USERS.WORKBENCH.PROJECTS/ARCH.MRL
     ARCH.PAS.4          extracted as _DS0/USERS.WORKBENCH.PROJECTS/ARCH.PAS
     ARCH.PRN.5          extracted as _DS0/USERS.WORKBENCH.PROJECTS/ARCH.PRN
     _DS0/USERS.WORKBENCH.PROJECTS/ARCH.EXE verified.
     _DS0/USERS.WORKBENCH.PROJECTS/ARCH.HLP verified.
     _DS0/USERS.WORKBENCH.PROJECTS/ARCH.MCR verified.
     _DS0/USERS.WORKBENCH.PROJECTS/ARCH.MRL verified.
     _DS0/USERS.WORKBENCH.PROJECTS/ARCH.PAS verified.
     _DS0/USERS.WORKBENCH.PROJECTS/ARCH.PRN verified.
```

Default — The original file's attributes are not preserved in the new file, i.e., the new file is created with the current date and time, the owner is the current user, and the protection is the user's default protection mask.

Syntax — Type **:preserve=** followed by any combination of the following keywords, separated by commas:

CREATED, OWNER, PROTECTION, ALL

Use this switch only with the :extract Function.

:protection=

Function — Use this switch to modify the protection mask associated with a file.

Default — The owner's protection mask.

Syntax — Type **:protection=** followed by a protection mask. Note that only those fields of the protection mask that are specified are altered. Fields not specified keep their original value.

For example, the switch :protection=s:,o:rwe allows the system no privilege, while the owner is granted read, write and execute privilege. The public and group privileges are unchanged.

Use this switch only with the :extract Function.

:rewind

Function — Use this switch to rewind the tape device.

Default — :norewind

Syntax — Type **:rewind**

Use this switch with any of the ARCH six functions, if you are working with a tape device.

:since=

Function — Use this switch to select files that match the File List parameter or the Archive File parameter and were created or modified since the specified date and time.

Default — All files that match the list are selected.

Syntax — Type **:since=** followed by a date or time or both.

Use this switch with any of the six ARCH Functions. If you use the :insert, :replace, or :new Functions, this switch applies to the File List parameter.

```
_DS0/UTILITY.WORK.SUPPORT.ARCH/ARCH.PAS.4
Logical :       56.0                            Created : 23-Dec-1985 10:50:10
Physical:       56.0                            Modified: 23-Dec-1985 10:50:13
Start   :     454741  File type:     Data       Archived: 23-Dec-1985 11:04:26
Record  :          1  File id  :        0       Owner   : [0003,0002]
Protection   : S:  RE,P:    ,G:DWRE,O:DWRE      Username: JIM
Archive file : _DS0/UTILITY.WORK.SUPPORT.ARCH.TST/ARCHIVE.ARC.7

               1 files.
```

Line 3 shows the file's physical size. This is the amount of space within the archive file that is allocated to the file. It is displayed in kilobytes rounded to the nearest tenth. This value may not always be equal to the logical size.

The Modified Field shows the date and time the file was last modified before it was archived.

Line 4 begins with the Start Field. This value is the number of the byte at which the file starts within the archive file.

The value in the File Type Field tells you which of the 10 WMCS file types the file belongs to:

Data	Any data file
Directory	A directory file
Image	A program image file
KsamData	A KSAM file containing data
KsamKey	A KSAM file containing keys
LLImage	An alternate image file
ArchiveCont	The continuation of an archive file from a previous volume
Encrypted	An encrypted file
System	An operating system file
Archive	An archive file

Numbers 256 to 65535 are available for user-defined file types.

The information next to "Archived:" on line 4 tells you the date and time the file was archived.

:volume= Function Use this switch to specify the volume number of the first volume used by ARCH.
Default :volume=1
Syntax Type **:volume=** followed by a numeral.

Use this switch only with the :insert, :replace, :extract, and :new Functions.

Examples

```
> arch archive.arc *
```

This command generates the following kind of report:

```
Archive listing of _DS0/USERS.WORKBENCH.PROJECTS/ARCHIVE.ARC.5
File name                 Logical/  Physical        Archived
-------------------     ---------------------  -------------------
ARCH.EXE.1                  129.0/     129.0  22-Dec-1985 17:06:54
ARCH.HLP.5                    1.8/       1.8  22-Dec-1985 17:06:54
ARCH.MCR.1                  102.1/     102.1  22-Dec-1985 17:06:54
ARCH.MRL.2                  206.2/     206.2  22-Dec-1985 17:06:54
ARCH.PAS.2                   38.0/      56.0  23-Dec-1985 10:23:10
ARCH.PRN.2                  185.9/     185.9  22-Dec-1985 17:06:54
                6 files.
```

Because no Function is specified, :list (the default Function) is assumed.

The first line of the display tells you the name of the archive file containing the files in the list. The display is divided into four columns and each column has a heading. Each file is listed in the File Name column.

The number in the Logical column tells you how much space the file actually occupies in the archive file.

The Physical column lists the space reserved (allocated) within the archive file for that particular file. Logical and physical size is listed in kilobytes. These two sizes need not be the same, as in the example of ARCH.PAS.2.

The difference in logical and physical size can be caused by the deletion and insertion of files. For example, suppose ARCH.DOC was the file which originally occupied the space where ARCH.PAS is now located. ARCH.DOC was 56 kilobytes long. After ARCH.DOC is deleted, ARCH.PAS is inserted into the archive file. But ARCH.PAS is only 38 kilobytes long. ARCH searches

Notes on Usage

If you have an archive file created prior to WMCS 5.0, you may need to use the :override switch because previous versions of ARCH sometimes created an archive file that was not an archive-type file.

ARCH combines several files into a single archive file. This allows you to group files and save disk space. ARCH is especially useful when writing small files to tape.

An archive file can fill more than one volume. Suppose your system manager inserts a diskette into drive _DX0 and mounts it. Then you type the following onto the command line and strike [RETRN]:

```
arch _dx0/rootdir/archive.000 *.*.* :new
```

When the diskette in drive _DX0 is full, the following prompt appears on your terminal screen:

```
Insert volume 2
Press RETRN when ready
```

Remove the diskette in _DX0, place another diskette in that drive, and strike [RETRN]. This prompt appears on your terminal screen:

```
Initialize this volume (Y or N)? >
```

Strike the Y character key if the diskette is not initialized under the WMCS. Strike the N character key to indicate that the diskette is already initialized.

If you strike N, the following kind of prompt appears on the screen:

```
Label = archive
```

You must now determine whether the volume you inserted, which is labeled "archive," is the diskette on which you wish to have the archive file continued.

If the volume is incorrect, strike the N character key and this prompt appears:

```
Insert volume 2
Press [RETRN] when ready
```

If you type Y, the archive file is continued on that volume.

Lines 1-6 begin with the filename within the archive file. This is followed by "extracted as" and the full file designation of the extracted file. Following these six lines are six more which list the filenames and whether or not the file is verified.

```
> arch archive.arc *.pas new.newl.new2.new3/* :builddir :edit=arch
    :newarch :extract
```

This command generates the following display:

```
     ARCH.PAS.4           extracted as _DS0/NEW.NEW1.NEW2.NEW3/NEWARCH.PAS
```

If the directories in the path /NEW.NEW1.NEW2.NEW3/ do not already exist, the :builddir switch creates them.

The :edit=arch:newarch switch changes the name of ARCH.PAS.4 to NEWARCH.PAS when ARCH.PAS.4 is extracted.

```
> arch archive.arc *.pas :list :full
```

This command displays all of the attributes of each file in the specified archive file. The format of the display is as follows:

```
_DS0/UTILITY.WORK.SUPPORT.ARCH/ARCH.PAS.4
Logical :       56.0                          Created : 23-Dec-1985 10:50:10
Physical:       56.0                          Modified: 23-Dec-1985 10:50:13
Start   :     454741   File type:     Data    Archived: 23-Dec-1985 11:04:26
Record  :          1   File id  :        0    Owner    : [0003,0002]
Protection    : S:  RE,P:     ,G:DWRE,O:DWRE   Username: JIM
Archive file : _DS0/UTILITY.WORK.SUPPORT.ARCH.TST/ARCHIVE.ARC.7

                1 files.
```

Line 1 contains the complete file designation of the file within the archive file.

Line 2 contains the file's logical size. The logical size is expressed in kilobytes and is rounded to the nearest tenth. This value represents the actual number of bytes in the file. (Contrast this with the physical size described below.)

The Created Field shows the date and time the file was created.

assign

Functional Description

Use this command to assign, reassign, or delete logical name assignments.

Command Line Syntax

Mnemonic	:=	Assign to the current process
	:==	Assign to the user process
	:===	Assign to the system
Required parameter	Name	
Optional parameter	Equivalence	

Parameters

Name	Function	Required. Use this parameter to specify the name to be assigned (or equated) to the value given for the equivalence.
	Default	None.
	Syntax	Type a string containing any characters. Put double quotation marks at the beginning and end of the string if it contains any spaces. Use the asterisk to tell the system what abbreviations of the name to accept.
Equivalence	Function	Optional. Use this parameter to specify the value to be equated to the name.
	Default	Delete the equivalence.

```
_DS0/UTILITY.WORK.SUPPORT.ARCH/ARCH.PAS.4
Logical :        56.0                          Created : 23-Dec-1985 10:50:10
Physical:        56.0                          Modified: 23-Dec-1985 10:50:13
Start   :      454741   File type:    Data     Archived: 23-Dec-1985 11:04:26
Record  :           1   File id  :       0     Owner   : [0003,0002]
Protection    : S:  RE,P:     ,G:DWRE,O:DWRE   Username: JIM
Archive file : _DS0/UTILITY.WORK.SUPPORT.ARCH.TST/ARCHIVE.ARC.7

          1 files.
```

Line 5 begins with the Record Field. This value represents the length of each record in the file.

The File ID Field is a user-defined field that can contain any value between 0 and 65535.

The Owner Field contains the user identification code (UIC) of the file's owner. The first number (the number preceding the comma) in the UIC is the user identification number, and the second number is the group identification number.

The Protection Field on line 6 shows the protection mask for the file.

The Username Field contains a name of the UIC that represents the file owner. The same UIC can have more than one name. If the system cannot find a name for the UIC, "Unknown" is displayed in this field.

Line 7 contains the file designation for the archive file that contains the original file.

The last line shows the number of files on display.

Using Prompts

```
> arch
Archive file spec> archive.arc
File specs> * :extract :verify
Destination> *
```

This performs the same function as the third example above.

If, in response to the "File specs>" prompt, you type a file designation and the :extract Function, you are prompted for the Destination parameter.

```
> "daily tab" :== "cd _ds0/users.grace.accounts.payable/"
```

This logical name moves you to the directory / USERS.GRACE.ACCOUNTS.PAYABLE/ on device _DS0 when you type 'daily tab' and strike [RETRN], no matter where you are on the system.

Note that if you want spaces in either the Name or Equivalence parameters you must surround them with double quotation marks. Also, if the logical name has spaces in it, you must enclose the logical name in single quotation marks when you type the logical name on the CIP command line.

```
> d*ir := "dir :size :create :head"
```

The asterisk in the foregoing name indicates that if you type D, DI, or DIR, you want the system to translate what you type into the specified equivalence.

```
> a :=
```

This deletes any equivalence that may have been assigned to A. Use the same kind of command to delete equivalences from either the user or the system logical name tables.

Using Prompts

None.

Notes on Usage

A logical name assignment string can contain up to 93 characters.

Any string of printable ASCII characters (except the question mark, ?, the colon, :, and the equal sign, =) can be used in a logical name assignment. If you need to use a question mark, colon, or equal sign, you must type a backslash, \, in front of the symbol as an accept character.

Using := or :== does not affect other users.

If the volume you insert needs to be initialized, and you type a Y in response to the prompt that asks whether the volume needs to be initialized, ARCH initializes the volume for you. If you specified a value for the :init= switch, that value is used as the command line for the initialization process. If you did not use the :init= switch, this prompt appears on your terminal screen:

```
     Device          >
```

Type the devicename of the device containing the volume you want to initialize. This prompt appears when you strike [RETRN]:

```
     Label           >
```

The label is for your convenience in identifying a volume (i.e., the WMCS does not use the label). It can consist of up to 16 characters and spaces. Any character is allowed.

A disk must be formatted the first time it is initialized. Formatting is optional thereafter.

Therefore, unless the volume is already initialized, it is necessary to specify the :format switch in the initialization process. This is best done when responding to the label prompt. For example:

```
     Label           > archivel :format
```

When the value for this option is specified and you strike [RETRN], the volume is initialized and this prompt again appears on the screen:

```
     Label = archivel Is this the correct volume (Y or N)? >
```

Type a Y if the specified volume is the one on which you want the archive file continued. Type N if the volume is incorrect.

Related CIP Commands

backup	Create backup copies of files
dinit	Initialize media
mnt	Mount a device
restore	Restore backup files

backup

Functional Description

Use this command to create a backup copy of files.

Command Line Syntax

Mnemonic	backup			
Required parameter	Directory			
Required parameter	File List			
Switches				
Files to be included in the function	:before= :confirm	:mod :since=	:exclude=	:uic=
Content of the display	:log			
Interaction with backup media	:mount= :space=	:init= :prefix=	:preinit :volume=	:rewind
Verification of files	:retries=	:verify		
Other	:logfile=	:openfiles=		

Syntax Type a string containing any characters. Put double quotation marks at the beginning and end of the string if it contains any spaces.

Switches

None.

Examples

```
> d := dir
```

This command assigns to the logical name D the value of DIR in the current process logical name table. Whenever D is typed, it is translated to DIR. As soon as the current process terminates, this logical name disappears.

If, for example, you use the CP Function in VEW to create a CIP, make the foregoing logical name assignment, and then log off of that CIP, the logical name assignment disappears along with that copy of the CIP.

```
> d :== dir
```

This command makes the same type of assignment as the previous example, but assigns the logical name to the user process (usually the table of logical name assignment belonging to the copy of the CIP made when you logged on--your user process).

If, for example, you use the CP Function in VEW to create a CIP then make the foregoing logical name assignment, when you log off of that CIP the logical name assignment still exists.

```
> d :=== dir
```

This command makes the same assignment as the previous examples, but places the assignment in the table of logical names belonging to the system. Therefore, this logical name becomes instantly available to every process on the system. Note that operator privilege is needed to assign logical names to the system.

one of which may contain wildcard characters.

:init= Function Use this switch to specify the parameters that are used when media are initialized. (This differs from :preinit, which is used to initialize the first volume, not to specify parameters.) In other words, when you specify :init=, you are not prompted for the parameters under which a diskette or tape is initialized.

This switch does not mean that media are automatically initialized. It allows you to specify (when BACKUP or ARCH is executed) what parameters are to be used when a diskette or tape is initialized. Thus, for each volume, you are prompted as to whether or not to initialize it. If you decide to initialize it (and you specified :init=) you are not prompted for initialization parameters.

For example, the following value for this switch provides for the automatic initialization of all volumes required for the execution of a particular version of BACKUP or ARCH:

```
:init="dinit dx0 backup_1"
```

Read the DINIT command description for details on what to specify.

Use two number signs as part of the volume label portion of the :init= character string in order to allow the incremental assignment of volume numbers to each diskette. For example, the following character string produces the labels shown for each volume:

```
:init="dinit dx0 2/feb/83_##"
```

These are the kind of labels produced by the foregoing character string:

```
2/feb/83 _1, 2/feb/83 _2, ...
```

Default Null. DINIT will prompt for the needed parameters if a volume is initialized.

Related CIP Commands

None.

:prefix=	Function	Use this switch to specify a prefix to be used in naming files created by the BACKUP utility. The string specified is concatenated to the front of the four numbers which indicate the sequence of the backup file.
	Default	Null. The prefix will be a concatentation of the day of the month and the month, for example, :prefix=10JAN0001.BAK
	Syntax	Type **:prefix=** followed by any string up to five characters long.
:preinit	Function	Use this switch to automatically initialize the first backup or archive volume.
	Default	:nopreinit
	Syntax	Type **:preinit**
:retries=	Function	Use this switch to specify the number of times to backup and verify a file if verification fails.
	Default	:retries=3
	Syntax	Type **:retries=** followed by a number.
:rewind	Function	Use this switch to rewind the tape device.
	Default	:norewind
	Syntax	Type **:rewind**
:since=	Function	Use this switch to select only those files that match the File List parameter and were created/ modified since the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:since=** followed by a date and/or time in the standard syntax.
:space=	Function	Use this switch to reserve space within the directory portion of a backup file for future insertion of a specified number of files.
	Default	:space=0
	Syntax	Type **:space=** followed by a positive integer.
:uic=	Function	Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.

Parameters

Directory

Function: Required. Use this parameter to specify the directory that will contain the backup copies of the files specified in the next parameter. The specified directory need not exist when you execute BACKUP.

Default: None.

Syntax: Type a standard file designation. Wildcard symbols are not allowed.

File List

Function: Required. Use this parameter to specify the files to be copied to the backup file specified in the Directory parameter.

Default: None.

Syntax: Standard file designation. File lists and wildcard symbols are allowed.

Switches

:before=

Function: Use this switch to select only those files that match the File List parameter and were created/ modified before the specified date and time.

Default: All files that match the File List parameter are selected.

Syntax: Type **:before=** followed by a date and/or time in the standard date and time syntax.

:confirm

Function: Use this switch to confirm or deny the desired operation on each file or device.

Default: :noconfirm

Syntax: Type **:confirm**.

:exclude=

Function: Use this switch to select only those files or devices that match the File List parameter and do NOT match any of the files specified as the value of the :exclude= switch.

Default: All files or devices that match the File List parameter are selected.

Syntax: Type **:exclude=** followed by a list of file or device designations, separated by commas, any

When you strike [RETRN] (having typed the foregoing command-line character strong), the following report appears, one line at a time, on your terminal screen:

```
_DC0/USERS.AL/
    1  CALENDAR.000.1              entered
    2  PRINTLOG.000.1              entered
    3  USERUP.COM.2                entered
_DC0/USERS.AL.AGENDAS/
    1  APR.000.2                   entered
    2  BOILERPL.ATE.1              entered
    3  FEB.000.2                   entered
    4  JAN.000.3                   entered
    5  JUL.000.2                   entered
    6  JUN.000.3                   entered
    7  MAR.000.3                   entered
    8  MAY.000.2                   entered
BACKUP.LOG file is written on VOL # 1
```

The first line in this report tells you files in _DC0/USERS.AL/ are being copied to a backup file. The files from that directory are listed in the order in which they are backed up, and you are informed when each is entered in the backup file.

Then the files in _DC0/USERS.AL.AGENDAS/ are backed up and the same kind of report appears.

Finally, you are told BACKUP.LOG is being copied to the directory in which the backup file itself is located, i.e., _DX0/ROOTDIR/.

	Syntax	Following the equal sign type the command to DINIT the volume. This string must be enclosed in quotes.
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:logfile=	Function	Use this switch to create a file that contains log messages from BACKUP.
	Default	A backup log file is created in the destination directory, but not in your default directory.
	Syntax	Type **:logfile=** followed by a standard file designation. No wildcard characters are allowed.
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**
:mount=	Function	Use this switch to mount the backup media.
	Default	None.
	Syntax	Type **:mount=** followed by a MNT command string enclosed in double quotation marks.
:openfiles=	Function	Use this switch to specify the number of files BACKUP will keep open.
	Default	:openfiles=100
	Syntax	Type **:openfiles=** followed by a number from 1 to 200.
:override	Function	Use this switch to override the requirement that the ARCH and BACKUP utilities work only on archive type files.
	Default	:nooverride
	Syntax	Type :override

```
     > backup  _dx0/rootdir users.al.*/*
```

The following report then appears on your terminal screen:

```
     _DC0/USERS.AL/
         1  CALENDAR.000.1          entered
         2  PRINTLOG.000.1          entered
         3  USERUP.COM.2            entered
     _DC0/USERS.AL.AGENDAS/

               Insert Volume 2
               Press RETURN when ready
```

This tells you the volume in drive _DX0 is full, and you should insert another volume in the drive and strike [RETRN] when you are ready for BACKUP to resume.

This kind of report appears in BACKUP.LOG for this execution of BACKUP:

```
_DC0/USERS.AL/
_DX0/ROOTDIR/25JAN0001.BAK  VOL # 1
    1  CALENDAR.000.1          entered
    2  PRINTLOG.000.1          entered
    3  USERUP.COM.2            entered
_DC0/USERS.AL.AGENDAS/
_DX0/ROOTDIR/25JAN0002.BAK  VOL # 2
    1  APR.000.2               entered
    2  BOILERPL.ATE.1          entered
    3  FEB.000.2               entered
    4  JAN.000.3               entered
    5  JUL.000.2               entered
    6  JUN.000.3               entered
    7  MAR.000.3               entered
    8  MAY.000.2               entered
Copying log file
```

In other words, the files copies from _DC0A0/USERS.AL/are located on the first volume to which backup copies were written, and the rest of the files are in /ROOTDIR/25JAN0002.BAK on volume 2.

When more than one volume contains backup files from a single execution of BACKUP, a copy of BACKUP.LOG is written to the last volume only.

```
>    backup    _dx0/rootdir    users.al.*/*    :volume=6    :prefix=al
:logfile=mylog.bak
```

This command causes reports similar to those in the previous example to appear on your screen. In addition, it creates a file, MYLOG.BAK.1, in your default directory. The contents of this file are:

Default — All files that match the specified list are selected.

Syntax — Type **:uic=** followed by a list of UICs or usernames.

:verify

Function — Use this switch to report whether the command line character string has been correctly executed.

For example, the following character string indicates that you want to include REPORT.DOC in ARCHIVE.000, and that you want a verification that the archived copy matches the original:

```
> backup _dc0/mydir/ report.doc :insert :verify
```

This kind of report appears on the screen when you strike [RETRN]:

```
1  REPORT.DOC.4     entered.
     REPORT.DOC.4     verified.
```

Default — :noverify

Syntax — Type **:verify**

:volume=

Function — Use this switch to specify the volume number of the first volume used by ARCH or BACKUP.

Default — :volume=1

Syntax — Type **:volume=** followed by a numeral.

Examples

```
> backup _dx0/rootdir users.al.*/*
```

The foregoing command backs up the most recent version of every file in every subdirectory of /USERS.AL/.

Suppose, for example, that this is the directory listing for /USERS.AL/:

```
Directory listing of _DC0A0/USERS.AL/
AGENDAS.DIR.1       CALENDAR.000.1       PRINTLOG.000.1       USERUP.COM.2
```

Notes on Usage

If the backup files are to be placed on a disk or tape other than the disk or tape in the default device, either use the :preinit switch or put the diskette or tape in the drive, initialize the volume (if necessary), mount it, and then execute BACKUP.

An execution of BACKUP can fill more than one volume. Suppose you insert a diskette into drive _DX0 and mount it. You then type the following onto the command line and strike [RETRN]:

```
> backup _dx0/rootdir *.*.*
```

When the diskette in drive _DX0 is full, the following prompt appears on your terminal screen:

```
Insert volume 2
Press RETURN when ready
```

Remove the diskette in _DX0, place another diskette in the drive, and strike [RETRN]. This prompt appears on your terminal screen:

```
Initialize this volume (Y or N)? >
```

Strike the Y character key if the diskette is not initialized under the WMCS. Strike the N to indicate the diskette is already initialized.

If you strike N, the following prompt appears on the screen:

```
Label = backup2
Is this the correct volume (Y or N)? >
```

In other words, read the label assigned to the volume to determine whether the volume you inserted is the one on which you wish to have the backup file continue.

If the volume is incorrect, strike the N and this prompt appears:

```
Insert volume 2
Press RETURN when ready
```

If you type Y to indicate the inserted volume is correct, the backup file is continued on that volume.

If the volume you insert needs to be initialized, and you type Y in response to the prompt that asks you whether the volume needs to be initialized, BACKUP initializes the media for you. If you have specified a value for the :init= switch, that value is used as the command line for

Were you to ask for a directory listing of _DX0/ROOTDIR/ following the execution of BACKUP you would find a file designated BACKUP.LOG in the listing. This file contains a record of the execution of BACKUP. In other words, were you to look at the contents of BACKUP.LOG, this is what you would see:

```
_DC0/USERS.AL/
_DX0/ROOTDIR/25JAN0001.BAK  VOL # 1
    1  CALENDAR.000.1          entered
    2  PRINTLOG.000.1          entered
    3  USERUP.COM.2            entered
_DC0/USERS.AL.AGENDAS/
_DX0/ROOTDIR/25JAN0002.BAK  VOL # 1
    1  APR.000.2               entered
    2  BOILERPL.ATE.1          entered
    3  FEB.000.2               entered
    4  JAN.000.3               entered
    5  JUL.000.2               entered
    6  JUN.000.3               entered
    7  MAR.000.3               entered
    8  MAY.000.2               entered
BACKUP.LOG file is written on VOL # 2
```

The first line of this report tells you the files in 25JAN0001.BAK were copied from _DC0/USERS.AL/.

The second line of this report tells you the backup file is designated 25JAN0001.BAK, i.e., it is the first (0001) backup file (.BAK) created on 25 January in /ROOTDIR/ on device _DX0. VOL # 1 tells you 25JAN0001.BAK is located on volume one of the volumes on which backup files, created by this execution of BACKUP, are located (this is explained below).

PRINTLOG.000, PRINTLOG.001, and USERUP.COM are the three files in _DC0/USERS.AL/ that were copied to 25JAN0001.BAK.

The second backup file, 25JAN0002.BAK, contains eight files. Note that BACKUP creates a backup file for each directory specified in the command line character string, e.g., 25JAN0001.BAK contains files from _DC0/USERS.AL/, and 25JAN0002.BAK contains files from _DC0/USERS.AL.AGENDAS/.

The BACKUP.LOG file created by BACKUP in the directory in which the backed up files are located is your guide to their location on the backup volumes.

Suppose the volume (diskette, tape, etc.) on which the backup file is to be located fills up before all the specified files have been backed up. For example, suppose you type the following character string on the command line and strike [RETRN]:

Daily Backup

DBACK.COM is the kind of command file that allows you to perform a daily backup. You might want to create a file like this in a subdirectory of your user-account default directory:

```
prompt INSERT FIRST BACKUP VOLUME IN _DX0, THEN PRESS [RETRN].
mnt _dx0                                  ! Mount drive containing backup vol
backup @_dc0/users.al.ecomcon/dailyback    ! Execute BACKUP parameter file
dmnt _dx0                                 ! Dismount _dx0
send _TT0 DAILY BACKUP OF _DC0 IS COMPLETE.
```

The following logical name assignment in USERUP.COM in your user-account default directory would allow you to execute this command file simply by typing DBACK on the CIP command line and striking [RETRN]:

```
dback :== @_dc0/users.al.ecomcon/dback
```

By typing the complete file designation as the equivalence for DBACK, you can execute DBACK.COM no matter what your default device and directory are.

The first line in DBACK.COM tells the CIP you want this prompt to appear on the screen when you execute DBACK.COM:

```
INSERT FIRST BACKUP VOLUME IN _dx0, THEN PRESS [RETRN].
```

In other words, you want to be reminded to insert a diskette, or volume, in drive _DX0. When you strike [RETRN], the CIP executes the next line in the command file, i.e., drive _DX0 is mounted.

Line 3 tells the CIP to execute BACKUP with its parameters coming from DAILYBACK.PRM, the BACKUP parameter file in _DC0/USERS.AL.ECOMCON/. By typing the complete path designation for DAILYBACK.PRM, the backup procedure can be executed no matter what your default device and directory are.

```
_DC0/USERS.AL/
_DX0/ROOTDIR/AL0001.BAK  VOL # 6
    1  CALENDAR.000.1            entered
    2  PRINTLOG.000.1            entered
    3  USERUP.COM.2              entered
_DC0/USERS.AL.AGENDAS/
_DX0/ROOTDIR/AL0002.BAK  VOL # 7
    1  APR.000.2                 entered
    2  BOILERPL.ATE.1            entered
    3  FEB.000.2                 entered
    4  JAN.000.3                 entered
    5  JUL.000.2                 entered
    6  JUN.000.3                 entered
    7  MAR.000.3                 entered
    8  MAY.000.2                 entered
BACKUP.LOG file is written on VOL   7
```

This report tells you which volume contains the backup copies of each directory and what files each directory contains. _DC0/USERS.AL/ is on volume 6, and _DC0/USERS.AL.AGENDAS/ is on volume 7.

This log file looks like the one in the preceding example with the following differences:

The :volume= switch started the volume numbering with 6.

The :prefix= switch named the new backup files AL0001.BAK and AL0002.BAK.

The :logfile= switch created log file MYLOG.BAK in your default directory. This log file is an exact copy of the log file created in the directory where the backup files were created.

Using Prompts

```
> backup
> Directory    >_dx0/rootdir
> File list    >users.al.*/*
```

This performs the same function as the first example.

Weekly Backup

WBACK.COM is the kind of command file that allows you to perform a weekly backup. You might want to create a file like this in a subdirectory of your user-account default directory:

```
prompt INSERT FIRST BACKUP VOLUME IN _dx0, THEN PRESS [RETRN].
mnt _dx0                                ! Mount drive containing backup vol
backup @_dc0/users.al.ecomcon/weekback  ! Execute BACKUP parameter file
dmnt _dx0                               ! Dismount _dx0
send _TT0 WEEKLY BACKUP OF _DC0 IS COMPLETE.
```

The foregoing command file is similar to DBACK.COM, discussed earlier in Daily Backup.

The contents of WEEKBACK.PRM, the BACKUP parameter file for a weekly backup, might look like this:

```
_dx0/rootdir/
_dc0/*/*.*
:mod
:since=28/01/83_17:00
:verify
:logfile=_dc0/users.al.backup.weekly/83~01~28.log
```

Note that the system manager must type the date in the foregoing parameter file before performing the weekly backup.

Monthly Backup

The files for the monthly backup are virtually identical to those for daily and weekly backups, except that you should back up everything on your system, i.e., no :since= or :mod switches should be included.

Related CIP Commands

arch	Create or modify archive files
dinit	Initialize a disk or a tape
mnt	Mount a device
restore	Restore backup files

the initialization process. If you have not used the :INIT= Switch, this prompt appears on your terminal screen:

```
     Device          >
```

Enter the devicename you want initialized. At this time, the prompt for the label appears:

```
     Label           >
```

The label is for your convenience in identifying a volume. It can consist of up to 16 characters and spaces. Any character is allowed.

A disk must be formatted the first time it is initialized. Formatting is optional thereafter.

Therefore, unless the volume is already initialized, it is necessary to specify the :format switch in the initialization process. This is best done when responding to the label prompt. For example:

```
     Label           > backup2 :format
```

When the value for this option is specified and you strike [RETRN], the volume is initialized and this prompt again appears on the screen:

```
     Label = backup2
     Is this the correct volume (Y or N)? >
```

Type Y if the specified volume is the one on which you want the backup file continued. Type N if the volume is incorrect.

The options available under BACKUP allow you to perform a selective backup, to verify the backed up copies match the originals, and to assign one of the copies of BACKUP.LOG to a directory other than your default. Study the examples below for information on the use of options.

Here are some sample command files that can be used for daily, weekly, and monthly backup. **The files shown in the following examples must be modified to meet your particular needs.**

bkup

Functional Description

Use this command to create a backup copy of a software product that can be used to reload the product onto your system disk.

Command Line Syntax

Mnemonic	bkup

Parameters

None.

Switches

None.

Examples

Follow this procedure to backup a software product.

Step 1 | Type the following on the command line and strike [RETRN]:

```
bkup
```

The BACKUP parameter file _DC0/USERS.AL.ECOMCON/DAILYBACK.PRM contains:

```
_dx0/rootdir/
_dc0/*/*.*
:mod
:since=02/02/83_17:00
:verify
:logfile=_dc0/users.al.backup/83~02~02.log
```

The first line of this parameter file tells the BACKUP Command the backup file (the file that will contain the backup copies) is to be located in /ROOTDIR/ on _DX0. Note that DBACK.COM instructed the CIP to mount _DX0 before executing the BACKUP parameter file.

The second line of the parameter file tells BACKUP what files are to be backed up, i.e., the most recent version of each file on _DC0.

The next two lines in the parameter file tell BACKUP you want to backup all files in _DC0 that match the pattern specified in line two, but including only those files modified since 5 p.m. on 2 February 1985.

Line 5 tells BACKUP you want BACKUP to verify the integrity of the copy of each file, and the last line indicates you want one copy of BACKUP.LOG to be named 83~02~02.LOG, and to be assigned to _DC0/USERS.AL.BACKUP/.

After BACKUP has completed, the CIP goes to the fourth line in DBACK.COM and dismounts _dx0.

Finally, the CIP sends this message to the system manager's terminal (which is usually _TT0):

```
(CIP_AL) DAILY BACKUP OF _DC0 IS COMPLETE.
```

Note that the system manager must daily type the date and time on the fourth line of DAILYBACK.PRM. (The system manager can also change the name of the log file every day, if desired.) Or the system manager could type the following on the fourth line, which eliminates the need to type the date and time each day:

```
:since=yesterday_17:00
```

Step 7 | Type Y. BKUP is now copying the product onto the tape or diskette.

If the backup fills more than one diskette or tape, your terminal will beep and display a message like the following each time you need to insert another diskette or tape:

```
Insert volume 2
Press RETURN when ready
```

Step 8 | Remove the diskette or tape that is in the drive and put the next volume into the drive and strike [RETRN].

Follow the instructions on your screen to finish the backup and display the backup results.

When the product is completely backed up to the media, the media will be dismounted so that you may remove it and store it in a safe place for future reference.

Using Prompts

None.

Notes on Usage

BKUP should only be used by the system manager to ensure that all of the files that compose the release are included in the backup and are accessible to the users when they are loaded back onto the system.

BKUP looks for a command file in /SYSEXE.USERS/ called xxxBKUP.COM, where xxx is the product mnemonic. A generic form of the command file, called GENBKUP.COM, is located in /SYSLIB/ so that users can set up their own products to work the same way as WICAT products.

btup

Functional Description

Use this command to display disk boot-block or tape-label attributes, or to assign disk boot-block attributes.

Command Line Syntax

Mnemonic	btup			
Required parameter	Devicename			
Switches				
Devices to be included	:class=	:exclude=	:confirm	:uic=
Content of the display	:log	:pause		
Attributes to be set	:alloc=	:ialloc=	:autoflush	:label=
	:cache=	:owner=	:devname=	:protection=
	:drivetype	:readahead	:drivetype=	:usercache=
	:forcedwrite			

This display appears on your screen:

```
****************** SOFTWARE PRODUCT BACKUP UTILITY ******************

Note: This backup mechanism initializes both tape and diskette media.

What device would you like to use?

      ==================================
      DEVICE |        DESCRIPTION
      ----------------------------------
        dx0  | 5.25 inch Diskette
        ct0  | Cartridge Tape
        mt0  | Cipher Tape
      other  | (none of the above devices)
      ==================================

Which DEVICE do you wish to backup the product to (press [SPACE]
            to change the value or [RETRN] to accept the value): dx0
```

Step 2 Strike the spacebar until the letters next to the colon (in the display above) change to the name of the drive for your tape or diskette.

Step 3 Strike [RETRN]. The following line appears on your screen:

```
Please enter the product identifier(s):
```

Step 4 Type the three-character product identifier (described in the installation instructions for the specific product you are making a backup copy of).

Step 5 Strike [RETRN]. If you are copying the product onto a tape, the following line appears on your screen:

```
Load tape -- Press [RETRN] when ready
```

If you are copying the product onto a diskette, this line appears on your screen:

```
Insert diskette -- Press [RETRN] when ready
```

Step 6 Load volume one of the tape or diskette into the appropriate drive and strike [RETRN]. After a moment, a prompt like the following appears on your screen:

```
_DX0 Mounted. Label is "wsa_5.1.0_voll:.
Is this the correct volume (Y or N) ? >
```

Default :class=*

Syntax Type **:class=** followed by a list of device classes separated by commas. Wildcard symbols are allowed.

:confirm

Function Use this switch to confirm or deny the desired operation on each device.

Default :noconfirm

Syntax Type **:confirm**

:devname=

Function Use this switch to set the boot-disk name. This is the name by which the disk (not the device in which it is mounted or the label by which you can identify the disk or volume) is known if that disk is used to boot the system.

Default Display the current value. If any other attribute switches are specified, nothing is displayed.

Syntax Type **:devname=** followed by a name up to 8 characters long.

:drivetype=

Function Use this switch to write the drive table for this drive type into the boot block. (This is done so the driver can be properly configured.)

Default Display the current value. If any other attribute switches are specified, nothing is displayed.

Syntax Type **:drivetype=** followed by one of the following valid drive types:

What you type	Description
FLOP09A	5 1/4 inch floppy
FLOP09B	5 1/4 inch floppy (5 sector)
FLOP015	8 inch floppy
WIN12	CMI 10 Mbyte 5 1/4-inch Winchester
WIN19	CMI 15 Mbyte 5 1/4-inch Winchester
WIN30	28 Mbyte 5 1/4-inch Winchester
WIN43	39 Mbyte 5 1/4-inch Winchester
SMD84B	76 Mbyte SMD disk
SMD168B	152 Mbyte SMD disk
SMD474B	421 Mbyte SMD disk
IMI20	20 Mbyte IMI disk
IMI40	40 Mbyte IMI disk

Related CIP Commands

None.

	Syntax	Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC that they represent.
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when the user presses any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:protection=	Function	Use this switch to modify the protection mask associated with the device.
	Default	The current value is displayed. If any other attribute switches are specified, nothing is displayed.
	Syntax	Type **:protection=** followed by a protection mask. Note that only those fields of the protection mask that are specified are altered. Fields not specified keep their original value. For example, the switch :PROTECTION=S:,O:RWE allows the system no privilege, while the owner is granted read, write, and execute privilege. The public and group privileges are unchanged.
:readahead	Function	Use this switch to issue an asynchronous read-request for the next logical sector in a file.
	Default	The current value is displayed. If any other attribute switches are specified, nothing is displayed.
	Syntax	Type **:readahead** or **:noreadahead** to override the default.
:uic=	Function	Use this switch to select only those devices that match the Device List parameter and are owned by a specified user or list of users.
	Default	The current value is displayed. If any other attribute switches are specified, nothing is displayed.
	Syntax	Type **:uic=** followed by a list of UICs in the standard UIC syntax, or the usernames of authorized system users. For example: :uic=[7,2],[*,3],system,backup

Parameters

Devicename	Function	Required. Use this parameter to specify the devices to be included.
	Default	The boot-block attributes of SYS$DISK are displayed or assigned depending on which, if any, switches are specified.
	Syntax	Type a list of devicenames, separated by commas.

Switches

:alloc=	Function	Use this switch to specify the number of sectors that are added to a file each time it is extended. The sectors allocated are not necessarily contiguous.
	Default	Display the current value. If any other attributes switches are specified, nothing is displayed.
	Syntax	Type **:alloc=** followed by an integer greater than zero.
:autoflush	Function	Use this switch to enable automatic flushing of modified sectors in the disk cache and modified FCB's each time the disk is updated.
	Default	Does not change the autoflush flag from its current value. (If you are not running LOGFLUSH, you should enable autoflush.)
	Syntax	Type **:autoflush** or **:noautoflush** to override the default.
:cache=	Function	Use this switch to set the number of sectors of disk cache to be maintained in memory.
	Default	Display the current value. If any other attribute switches are specified, nothing is displayed.
	Syntax	Type **:cache=** followed by a positive integer.
:class=	Function	Use this switch to specify the class(es) of device(s) you wish to include.

```
Label       volume label of disk _DS0
Volumeid    (this field is not currently used)
Devname     name of the boot disk
Cache       total number of sectors kept in the computer's main memory
            for faster access to memory
Sectorsz    number of bytes per sector
Usercache   maximum number of cache sectors a user can use at one time
Ialloc      initial number of sectors allocated to a file
Numbsect    number of sectors on the disk
Drivetype   type of drive the volume is in
Created     date the disk was initialized
Owner       UIC of the user who owns the disk
Protection  disk's protection mask
Readahead   indicates whether readahead is enabled or disabled
Autoflush   indicates whether flushing on each write is enabled or disabled
Shiftcnt    how far left to shift a 1 to get the sector size
Alloc       number of sectors added when a file is extended
Fcbsector   sector number of the start of the FCB.SYS file
Username    username which corresponds to the volume's UIC
```

```
> btup * :class=tape
```

This command lists the boot-block attributes for all devices that are of the class TAPE.

```
_CT0
Devname   : CT              Protection: S:  RE,P:     ,G:DWRE,O:DWRE
Label     : mcs_5.0         Volumeid  : 0     Created   : 19-Dec-1985 12:12:15
Owner     : [0002,0002]     Cache     : 8     Readahead
Blocksize: 1024             Username  : OPERATOR

_MT0
Devname    : _MT            Protection: S:  RE,P:     ,G:DWRE,O:DWRE
Label      : test_tape      Volumeid  : 0     Created   : 11-Oct-1985 11:22:11
Owner      : [0002,0002]    Cache     : 8     Readahead
Blocksize : 1024            Username  : FORTRAN
```

What each field means:

:exclude=

Function Use this switch to exclude devices from the Device List parameter.

Default All devices that match the Device List parameter are selected.

Syntax Type **:exclude=** followed by a list of devicenames separated by commas. Wildcard symbols are allowed.

:forcedwrite

Function Use this switch to cause write requests to wait until the data is actually written successfully to the drive before returning.

Default The current value in the boot block is displayed. If any other attributes switches are specified, nothing is displayed.

Syntax Type **:forcedwrite** or **:noforcedwrite**

:ialloc=

Function Use this switch to specify the number of sectors (not necessarily contiguous) that are assigned to a file the first time data are written to it.

Default The current value is displayed. If any other attribute switches are specified, nothing is displayed.

Syntax Type **:ialloc=** followed by a positive integer.

:label=

Function Use this switch to assign an identifying user label to the device.

Default The current value is displayed. If any other attribute switches are specified, nothing is displayed.

Syntax Type **:label=** followed by a string of up to 16 ASCII characters and/or spaces (use double quotation marks with spaces).

:log

Function Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

Default The value specified by the OPTION command.

Syntax Type **:nolog** or **:log** to override the default.

:owner=

Function Use this switch to change the owner (UIC) of the device.

Default The current value is displayed. If any other attribute switches are specified, nothing is displayed.

Related CIP Commands

dinit	Initialize a disk or a tape
mnt	Mount a device

:usercache= Function Use this switch to set the maximum number of sectors of disk cache that any one process can consume at one time.

Default The current value is displayed. If any other attribute switches are specified, nothing is displayed.

Syntax Type **:usercache=** followed by an integer between 1 and 16 inclusive.

Examples

> btup _ds0

This command generates the following display of disk boot-block attributes:

```
_DS0
Devname  : DS             Protection: S: WRE,P: WRE,G: WRE,O:DWRE
Label    : MCS_source     Volumeid  : 0     Created  : 01-Jan-1985 01:29:19
Owner    : [0001,0001]    Cache     : 150   Readahead
Username : SYSTEM         Usercache : 8     NoAutoflush
Numbsect : 421000         Sectorsz  : 1024  NoForcedWrite
Fcbsector: 206803         Shiftcnt  : 10    Drivetype : SMD474b
Ialloc   : 50             Alloc     : 50
```

What each field means:

Functional Description

Use this command to change the default directory.

Command Line Syntax

Mnemonic	cd	
Optional parameter	Directory	
Switches	:log	:perm

Parameters

Directory	Function	Optional. Use this parameter to specify the default device and/or directory to which you wish to move.
	Default	The directory you are in (i.e., the directory that is already the default).
	Syntax	Type the standard syntax for directory paths. Wildcard symbols are not allowed.

Label	volume label of the tape
Volumeid	(this field is not currently used)
Devname	name of the boot tape
Cache	total number of sectors kept in the computer's main memory for faster access to memory
Blocksize	number of bytes per block
Created	date the tape was initialized
Owner	UIC of the tape's owner
Protection	tape's protection mask
Readahead	indicates whether readahead is enabled or disabled
Username	username which corresponds to the tape's UIC

```
> btup _df* :label=backup_disk
```

If the devices df0 and df1 are mounted, then this command assigns the label BACKUP DISK to devices _DF0 and _DF1 and displays a report like this:

```
    _DF0 Updated.
    _DF1 Updated.
```

Using Prompts

```
> btup
Devicename     > _ds0
```

This command performs the same function as the first example.

Notes on Usage

BTUP works only on mounted disks and tapes (mounted regularly or special).

Tape-label attributes cannot be updated; they can only be displayed.

When a disk boot block is updated, the disk must be dismounted and remounted again if the changes are to take effect within the system (such as when the cache size is changed).

There is only one boot block on a disk and it is at sector 0.

```
> cd -
```

This command moves you to the parent directory of the directory you are in. For example, if you are in _DC0/USERS.MARCH/, this moves you to _DC0/USERS/. The following display appears when you strike [RETRN]:

```
    _DC0/USERS/
```

```
> cd _dc0/users.march/ :perm :nolog
```

The :perm switch has meaning only inside of command files. When you execute a command file that contains instances of the CD Command, you usually find yourself (upon completion of that command file's execution) in the same directory you were in before you executed the command file. However, if the foregoing command appears in such a command file, you would find yourself in _DC0/USERS.MARCH/ upon completion of the command file's execution. The :nolog switch means that when the command is executed in the command file, no log message appears on the terminal screen to indicate that the default directory has been changed.

Using Prompts

None.

Notes on Usage

If you type CD on the command line and strike [RETRN], the file designation for your default directory (the directory you are already in) appears on the screen.

Related CIP Commands

def	Display the name of the default device and the default directory
crd	Create a directory

checksum

Functional Description

Use this command to sum the values of the characters in a file. This sum (a unique number) can be used as a check to see if two files are the same.

Command Line Syntax

Mnemonic	checksum			
Required parameter	File List			
Switches				
File selection	:before= :mod	:exclude=	:since=	:uic=
Function control	:bytesum			
Output control	:pause			
Other	:rewind			

Parameters

File List	Function	Required. Use this parameter to specify the files to be processed by CHECKSUM.
	Default	None.
	Syntax	Type a list of filenames separated by commas. Wildcard symbols are allowed.

Switches

:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:perm	Function	Use this switch to make the change permanent or to make it persist even after the current CIP ends execution (unless the current CIP is your user process).
	Default	:noperm
	Syntax	Type **:perm**

Examples

```
> cd _dc0/users
```

This command changes the default directory to /USERS/, on disk _DC0. The following display appears on your screen when you strike [RETRN]:

```
    _DC0/USERS/
```

```
> cd .march
```

This command changes the default directory to /MARCH/. The dot, ., is necessary because /MARCH/ is a subdirectory of /USERS/. Note that this command does not change the default device. The following display appears when you strike [RETRN]:

```
    _DC0/USERS.MARCH/
```

:rewind	Function	Use this switch to rewind the tape device before searching for file(s) to CHECKSUM.
	Default	:norewind
	Syntax	Type **:rewind**
:since=	Function	Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:since=** followed by a date or time in the standard syntax.
:uic=	Function	Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.
	Default	All files that match the specified list are selected.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> checksum *
```

This command prepares a checksum for each file in the default directory.

CHECKSUM gives you a sum of the values assigned to each byte in the specified file(s), and tells you how many bytes a file contains. These "sums" are used by the VERIFY Command to check (hence the term "checksum") or verify that a copy of a file is identical to its original.

Type >**checksum** * on the command line and the following report appears:

```
                               Checksum Summary
                       File Name                          Date        Size    Sum
 ---------------------------------------------------  -----------  --------  -----
 _DS0/TMP/A.EXE.3                                     09-Jan-1985     14336  47019
 _DS0/TMP/A.OUT.1                                     09-Jan-1985     13990  28527
 _DS0/TMP/I00204A..1                                  11-Jan-1985       982  32090
 _DS0/TMP/JU00306A..1                                 11-Jan-1985      7121  26152
 _DS0/TMP/KILLER.EXE.1                                09-Jan-1985     16384  46689
 _DS0/TMP/L.MCR.2                                     09-Jan-1985     23334  31234
 _DS0/TMP/L.MRL.1                                     09-Jan-1985      1084  20855
 _DS0/TMP/L.PAS.2                                     09-Jan-1985        73  10163
 _DS0/TMP/L.PRN.2                                     09-Jan-1985       263   9777
 _DS0/TMP/T.PAS.1                                     09-Jan-1985      2048   4101
 _DS0/TMP/TRECORD.PAS.1                               09-Jan-1985      2048   4101
 _DS0/TMP/TRECORD.TDT.3                               09-Jan-1985      1305   9049
 _DS0/TMP/TT.EXE.1                                    09-Jan-1985     16384  12957
 _DS0/TMP/TT.MCR.2                                    09-Jan-1985     21654  52997
 _DS0/TMP/TT.MRL.2                                    09-Jan-1985      8751   7792
 _DS0/TMP/TT.PAS.5                                    09-Jan-1985      1585  28306
 _DS0/TMP/TT.PRN.2                                    09-Jan-1985      3065  55201
 _DS0/TMP/U.C.3                                       09-Jan-1985       295  11050
```

Using Prompts

```
> checksum
File list     > *
```

This command prepares a checksum for each file in the default directory.

Notes on Usage

For the report produced by this command to be used by the VERIFY Command, you must redirect the output of CHECKSUM to a file. You may wish to specify .CKS as the file extension for this file so you can easily identify the file containing the CHECKSUM report.

Switches

:before=	Function	Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:before=** followed by a date or time in the standard date and time syntax.
:bytesum	Function	Use this switch to select the older and less reliable form of CHECKSUM. This may be needed for compatibility when comparing checksums generated with pre-WMCS 5.0 versions of the CHECKSUM Command.
	Default	The rotated sum method of generating checksums.
	Syntax	Type **:bytesum** to select the sum-of-bytes method of generating checksums.
:exclude=	Function	Use this switch to exclude files from the File List parameter.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of filenames separated by commas. Wildcard symbols are allowed.
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when the user presses any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.

chkd

Functional Description

Use this command to check for bad sectors on a disk.

Command Line Syntax

Mnemonic	chkd	
Required parameter	Devicename	
Optional parameters	Sectors	
Switches		
Content of the display	:log	
Extended checking	:full	:numreads=
Hibernate processes	:hibernate	

For example, suppose the default directory is /TMP/ and it contains the following:

```
Directory listing of _DS0/TMP/
File name                 Logical/  Physical     Created
------------------- ------------------- -------------------
A.EXE.3                    14.0/       14.0 09-Jan-1985 05:51:58
A.OUT.1                    13.7/       14.0 09-Jan-1985 05:51:56
I00204A..1                  1.0/       50.0 11-Jan-1985 10:26:27
JU00306A..1                 7.0/       50.0 11-Jan-1985 15:21:49
KILLER.EXE.1               16.0/       16.0 09-Jan-1985 06:08:50
L.MCR.2                    22.8/       23.0 09-Jan-1985 14:12:16
L.MRL.1                     1.1/        2.0 09-Jan-1985 14:04:59
L.PAS.2                     0.1/        1.0 09-Jan-1985 14:04:50
L.PRN.2                     0.3/        1.0 09-Jan-1985 14:04:56
T.PAS.1                     2.0/        2.0 09-Jan-1985 05:37:41
TRECORD.PAS.1               2.0/        2.0 09-Jan-1985 05:35:21
TRECORD.TDT.3               1.3/        2.0 09-Jan-1985 06:29:39
TT.EXE.1                   16.0/       16.0 09-Jan-1985 06:31:11
TT.MCR.2                   21.2/       22.0 09-Jan-1985 06:30:40
TT.MRL.2                    8.6/        9.0 09-Jan-1985 06:30:21
TT.PAS.5                    1.6/        2.0 09-Jan-1985 06:30:09
TT.PRN.2                    3.0/        3.0 09-Jan-1985 06:30:17
U.C.3                       0.3/        1.0 09-Jan-1985 05:51:42
           18 Files.      132.0/      230.0  Total blocks.
```

process. This ensures exclusive access to the disk if CHKD is unable to allocate it.

Default :nohibernate
Syntax **Type :hibernate**

Examples

> chkd _df0

Execute this command as soon as possible after you get your system and after moving the system.

The device must be mounted for CHKD to operate. The underscore is optional and will be inserted if it is omitted on the command line.

This command generates the following kind of report:

```
   Checking sectors 0-1231 on _DF0
.........|.........|.........|.........|.........|
.........|.........|.........|.........|.........|

    Sectors marked good, confirmed good   =      1231
    Sectors marked good, found to be bad  =         1
    Sectors marked bad,  confirmed bad    =         0
    Sectors marked bad,  found to be good =         0
                   Total sectors checked  =      1232
```

The first line in the report shows the devicename and which sectors on the disk are being checked.

Each dot means 1 percent of the sectors have been checked, and each vertical line means 10 percent of the specified sectors have been checked. Note: While CHKD is executing only the dots appear until all the specified sectors are checked. Any bad sectors found are reported at the end of the check.

The last five lines of the report summarize the status of the sectors that were checked.

Related CIP Commands

verify	Verify that a file copy matches the original

```
> chkd _df0 0-49
```

If new bad sectors are found, the following kind of report appears:

```
  Checking sectors 0-49 on _DF0
.........|.........|.........|.........|.........|

There are 2 bad sectors allocated to the following files:
** Sector 10 found in _DF0/ROOTDIR/ALLOC.EXE.1
** Sector 42 found in _DF0/ROOTDIR/ARCH.EXE.1

There are 1 other sectors that are bad:
** Sector        26

    Sectors marked good, confirmed good   =        47
    Sectors marked good, found to be bad  =         3
    Sectors marked bad,  confirmed bad    =         0
    Sectors marked bad,  found to be good =         0
                   Total sectors checked  =        50
```

DO NOT BE ALARMED BY BAD SECTORS. You can expect at least 1 percent of the sectors on any disk to be bad, but unless 5 to 8 percent of the sectors on a disk are marked bad, you should not worry about them.

Furthermore, the WMCS tries to retrieve data from bad sectors whenever the WMCS is asked to read a bad sector, but the WMCS does not write to bad sectors. If there is a file on a bad sector, it is reported by CHKD as in the example above ALLOC.EXE and ARCH.EXE). If the sector was not previously allocated to a file, only the sector number is reported (sector 26).

Using Prompts

```
> chkd
Devicename     > _DF0
```

This command performs the same function as the first example.

Parameters

Devicename	Function	Required. Use this parameter to specify the disk to be checked. The disk must be mounted before you execute CHKD (see the MNT Command).
	Default	None.
	Syntax	Type a devicename. Wildcard symbols are not allowed.
Sectors	Function	Optional. Use this parameter to specify the sector(s) to check.
	Default	All sectors, from sector 0 to the last sector on the disk, are checked.
	Syntax	Type a range of numerals, such as 100-200.

Switches

:full	Function	Use this switch to do a more extensive test on unallocated sectors. Note that an extensive test may take up to five times as long to execute.
	Default	:nofull
	Syntax	Type :full
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type :nolog or :log to override the default.
:numreads=	Function	Use this switch to specify the number of reads to attempt in each sector. This number of reads must be performed successfully on each sector before it is marked or confirmed good.
	Default	:numreads=1
	Syntax	Type :numreads= followed by a positive integer.
:hibernate	Function	Use this switch to hibernate all processes on the system for the duration of the CHKD

Functional Description

Use this command to execute another copy of CIP.

Command Line Syntax

Mnemonic	cip
Optional parameter	Command
Switches	None

Parameters

Command

Function: Optional. Use this parameter to execute commands in other files or directories without leaving the file or directory you are now in. Your current working environment is preserved.

Default: None.

Syntax: Type the standard syntax for CIP commands.

Switches

None.

Status of the sector before you run CHKD	CHKD's attempt to read the sector	What CHKD does to the sector's status
good	successful	Leaves the sector's status unchanged.
good	unsuccessful	If the sector is allocated, CHKD marks it bad. If the sector is unallocated, CHKD runs an extensive test and marks the sector bad if the test fails; good if the sector passes the test.
bad	successful	If the sector is allocated, it remains bad. If the sector is not allocated, it must pass the extensive test before CHKD marks it good.
bad	unsuccessful	CHKD leaves it bad.

CHKD identifies a sector as **bad** when it cannot read the sector. A sector is marginally bad when CHKD is able to read the sector only after repeated attempts to do so, i.e., after retries.

If a particular sector is successfully read, i.e., if no errors are found, that sector is marked **good** in the disk's bitmap. Thus, sectors previously marked bad can sometimes be recovered. CHKD does not fix bad sectors, but revises the report on the condition of the sectors so the WMCS knows which sectors are good, etc.

If the sector is not successfully read and is allocated to a file, the sector is marked **bad** and no attempt is made to write over the sector.

```
> chkd _dx0 100-200 :full

     Checking sectors 100-200 on _DX0
     .........|.........|.........|.........|.........|
     .........|.........|.........|.........|.........|

     Sectors marked good, confirmed good   =       100
     Sectors marked good, found to be bad  =         1
     Sectors marked bad,  confirmed bad    =         0
     Sectors marked bad,  found to be good =         0
     Total sectors checked  =        101
```

When the :full switch is specified or if the sector is not successfully read and is not allocated to a file, CHKD attempts to write four different bit patterns to the sector. If it succeeds in writing all four patterns, the sector is marked good. If CHKD fails to write any of the four patterns, the sector is marked bad.

Use the LOG Command to terminate the new CIP, i.e., when you type log onto the CIP command line and strike [RETRN], User logged off" appears on your screen and the cursor reappears next to a right angle bracket on the original CIP command line.

Related CIP Commands

log	Terminate the CIP in which you are working
stat	Display a list of current processes

Notes on Usage

CHKD does not disturb the data on the disk.

> CAUTION: Do not execute this command on a disk on which other programs are in operation unless you use the :hibernate switch. Otherwise, an inaccurate bitmap for that disk may result, inasmuch as both programs would be simultaneously updating BITMAP.SYS.
>
> Furthermore, do not execute CHKD if the environment in which your system is located has experienced a sudden change in temperature, e.g., if the system's hardware has been very hot or very cold.

Related CIP Commands

recover	Rebuild the file system on a disk

cmdst

Functional Description

Use this command to display a list of previous CIP commands executed at your terminal.

Command Line Syntax

Mnemonic	cmdst	
Switches	:full	:pause

Parameters

None.

Switches

:full

Function — Use this switch to display a list of all commands entered since the user logged on to the system.

Default — :nofull

Syntax — Type :full

Examples

> cip

This brings up a new version of CIP. When you LOG off, you return to the previous CIP. This command is useful when you want to temporarily go to another directory and work with those files.

For example, if you are working in /USERS.TOM/ and want to briefly look at a file in /USERS.DAVE/, type **cip** on the command line and strike [RETRN]. A new command line prompt, CIP>, appears. Now type **cd -.dave**, strike [RETRN], and this report appears on your screen:

```
_DC0/USERS.DAVE/
```

When you are through working in /USERS.DAVE/, simply type **log** on the command line, strike [RETRN], and you are back in /USERS.TOM/ again.

> cip stat;time

This executes the STAT and TIME commands, then brings up a command line prompt, CIP>.

Using Prompts

None.

Notes on Usage

You can execute any CIP command under the new CIP that you can execute in the original CIP.

When you type **cip** onto a command line to create a new CIP, you can, instead of striking [RETRN], type a CIP command line character string and then strike [RETRN]. For example, **> cip cd -.dave** performs the same function in one step as the two step command described in the first example.

```
> cmdst :full :pause
```

This command displays all of the information in the previous example plus all commands executed before the 20th command. The :pause switch causes the display to pause after each 20 commands.

Using Prompts

None.

Notes on Usage

CIP has an internal buffer 1.5Kb in size, which allows it to save between 80 and 140 of the previously executed commands. When the buffer is full, the commands begin "scrolling" through the buffer, and the oldest commands are lost.

Related CIP Commands

None.

config

Functional Description

Use this command to assign a device driver to the boot device, select some of the files that will constitute the WMCS when your system is booted, and assign the number of TTY ports to poll.

Command Line Syntax

Mnemonic	config	
Optional parameter	Device	
Switches		
Display of file	:list	:log

Parameters

Device	Function	Optional. Use this parameter to specify the device on which the configuration is done.
	Default	SYS$DISK
	Syntax	Type the name of the device where the files used by CONFIG will be read and written. Wildcard symbols are not allowed.

:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when the user presses any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.

Examples

> cmdst

This command shows either all of the commands you have executed or the last 20 commands, whichever is smaller. If this is the first command typed after you log on, the following display appears:

```
     > cmdst
```

If you have been working in CIP for several minutes, your display may look like this:

```
     > mnt _dxl
     > copy *.dat _dxl/rootdir/*
     > cd .work
     > dir
     > copy *.exe _dxl/*/* :build
     > sp _dxl
     > dmnt _dxl :auto
     > def
     > pu
     > dm
     > stat
     > vew test.pas
     > link test
     > test
     > dstat
     > vew
     > pas test.pas
     > link test
     > test
     > cmdst
```

Note that only the last 20 commands are shown.

CONFIG: the WMCS configuration program

Use CONFIG to assign device drivers to the boot devices on your system, and to select some of the files that will constitute the WMCS when your system is booted.

COMMANDS

[SHIFT] ?	Display information on the item in the CONFIG display next to which the cursor is located.
spacebar	Toggle the value assigned to the item next to which the cursor is located.
[RETRN]	Accept the value appearing in the display.
back space or [CTRL] h	Move the cursor to the previous item in the display.
Type	Enter in appropriate new values by typing the new number.
[CTRL] c	Abort the CONFIG program.

```
SYSTEM MODEL: 100

Boot device driver: SMD$100.DSR

Number of tty ports:  1
Include (kernel,kernelbug): Kernelbug
Include tty: yes
Include disk: yes
Include math1010: yes
Include ksam: yes
Include tape: yes
Include queue: yes

Date : 1985
Update (Y or N)? > n

Configuration completed
```

CONFIG allows you to toggle through valid options in each field as you go. The fields are described below:

SYSTEM MODEL:

This is the model of the system you want to configure. To see other possible values, press the spacebar until the desired model appears.

Include tape:

The tape class handler supports the various kinds of tape drives that can be used with the WMCS. Include this module if your system has a tape drive.

Include queue:

The queue class handler is used to manage print and batch jobs.

Date:

The date should be set to the current year. Modify this value by typing in the new year or by toggling with the spacebar.

Update (Y or N)? >

This allows you to make permanent the changes you have made in CONFIG. The system configuration file (SYSCONFIG) is updated only if you type Y. Characters other than Y or N are ignored.

```
> config :list :nolog
```

This command string lists the values but does not stop and allow you to modify the fields displayed.

```
SYSTEM MODEL: 100

Boot device driver: SMD$100.DSR

Number of tty ports:  1
Include (kernel,kernelbug): Kernelbug
Include tty: yes
Include disk: yes
Include math1010: yes
Include ksam: yes
Include tape: yes
Include queue: yes
Date : 1985
```

Using Prompts

None.

Switches

:list

Function: Use this switch to list all of the values in the SYSCONFIG file without allowing the user to modify its contents.

Default: :nolist

Syntax: Type **:list**

:log

Function: Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

Default: The value specified by the OPTION command.

Syntax: Type **:nolog** or **:log** to override the default.

Examples

```
> config
```

This command causes the following kind of display to appear on your screen:

copy

Functional Description

Use this command to create copies of files or to concatenate several files.

Command Line Syntax

Mnemonic	copy
Required parameters	Source Destination

Switches

File selection switches	:before= :exclude=	:mod :uic=	:confirm	:since=
Destination specification switches	:builddir :edit=	:append	:concat	:replace
Destination file attribute switches	:owner=	:preserve=	:protection=	
Others	:log	:closeflush	:multi	:rewind

Boot device driver:

The boot device driver is the program that controls, or drives, the boot devices on the system, e.g., the system disk. The boot device or system disk is the device from which the system is booted whenever you turn on the power to your system. On most WICAT systems, this is the Winchester (or hard) disk.

Number of tty ports: 7

Use this field to specify the number of TTY-class serial ports to be supported. These are the default values: System 150 - 7 and System 100/200 - 8. To modify this value, type in the number of ports or toggle the values until the desired number is displayed.

NOTE: If you have two I/O boards specify 14 for this value on the System 150.

The remaining fields are used to select some of the files that will constitute the WICAT Multi-user Control System (WMCS) when your system is booted.

Include (kernel,kernelbug):

Choose between kernel and kernelbug, the core of the WMCS. Kernelbug has built-in debugging facilities experienced systems programmers can use in low-level development work. Choose kernel for normal system operation.

Include tty:

The TTY class handler supports the terminals on your system and should almost always be included.

Include disk:

The disk class handler supports the disks on your system and should almost always be included.

Include math1010:

The math1010 class handler supports the old floating point math routines used in some old Pascal programs. This normally does not need to be included as a WMCS module.

Include ksam:

The KSAM class handler supports the keyed sequential access method of file access. Include this module if you are developing or using programs that employ KSAM.

Syntax Type **:builddir**

:closeflush

Function Use this switch to force write all sectors of each file at close time and report any sector errors.

Default :nocloseflush

Syntax Type **:closeflush**

:concat

Function Use this switch to cause all of the source files to be concatenated into a single destination file.

Default :noconcat

Syntax Type **:concat**

:confirm

Function Use this switch to confirm or deny the desired operation on each file or device.

Default :noconfirm

Syntax Type **:confirm**

:edit=

Function Use this switch to specify a series of edits to be performed on the destination file designation. The edits are specified as a series of "find and switch" strings. The destination file designation is searched for the first occurrence of the "find" string. If it is found, it is replaced by the "switch" string.

Default No edits are performed.

Syntax Type **:edit=** followed by a list of "find and switch" strings separated by commas. The find string is separated from the switch string by a colon. Either one or both of the strings may be empty.

:exclude=

Function Use this switch to exclude files from the File List parameter.

Default All files that match the File List parameter are selected.

Syntax Type **:exclude=** followed by a list of file designations separated by commas. Wildcard symbols are allowed.

:log

Function Use this switch to specify whether log messages are displayed. (Log messages are informational

Notes on Usage

CONFIG should only be used by the system manager. Equivalent functions can be performed by the utility SYSPROF. This utility is needed only if no terminal setup file is available on your system.

Related CIP Commands

sysprof	Set up a system profile

Syntax — Type **:protection=** followed by a protection mask.

NOTE: Only those fields of the protection mask that are specified are altered. Fields not specified keep their original value.

For example, :protection=s:,o:rwe allows the system no privilege, while the owner is granted read, write, and execute privilege. The public and group privileges are unchanged.

:replace

Function — Use this switch to create a new copy of a file and delete the old one automatically.
Default — :noreplace
Syntax — Type **:replace**

:rewind

Function — Use this switch to rewind the tape device.
Default — :norewind
Syntax — Type **:rewind**

:since=

Function — Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.
Default — All files that match the File List parameter are selected.
Syntax — Type **:since=** followed by a date or time in the standard syntax.

:uic=

Function — Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.
Default — All files that match the specified list are selected.
Syntax — Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> copy /budget/march83.dat *
```

This command copies the latest version of the file named MARCH83.DAT from the directory named /BUDGET/ to the default directory. The file created in the default directory is also called MARCH83.DAT.

Parameters

Source

Function: Required. Use this parameter to specify the names of the files to be copied.

Default: None.

Syntax: Type a list of filenames separated by commas. Wildcard symbols are allowed.

Destination

Function: Required. Use this parameter to specify the names of the files to which the source files are copied. Field substitution is allowed. That is, everywhere an asterisk, *, appears in the value typed for this parameter, the corresponding field from the corresponding source file is substituted.

Default: None.

Syntax: Type a list of file designations separated by commas. Wildcard symbols are allowed.

Switches

:append

Function: Use this switch to add one file to the end of another file without creating a new destination file (if it already exists).

Default: :noappend

Syntax: Type :append

:before=

Function: Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.

Default: All files that match the File List parameter are selected.

Syntax: Type :before= followed by a date or time in the standard date and time syntax.

:builddir

Function: Use this switch to create destination directories automatically if they do not exist.

Default: :nobuilddir

```
     will not fit on this volume...
     Insert next volume and press return
```

At this point you should remove the diskette in the drive, insert the second diskette (this diskette must already be initialized), and press [RETRN]. COPY continues copying files onto this new diskette. Remember, if any one of the files is too large to fit on a single diskette, COPY will run forever, trying over and over to copy the one file that is too large onto diskette after diskette. Note that even with the :multi switch, files are not split across multiple volumes. Any given file exists entirely on one diskette. If you have files that are too large to copy, use the BACKUP command to make a backup copy of the file on multiple volumes.

```
> copy flinch.exe _dc0/sysexe.users/ :owner=system :protection=p:re
```

This command line copies the latest version of the file named FLINCH.EXE in the default directory to the directory named /SYSEXE.USERS/ on _DC0.

The :owner= switch assigns the owner of the new file to the UIC for the username SYSTEM. (Typically the UIC for SYSTEM is [1,1].)

The protection mask assigned to the file is the same as the user's default protection mask, except that the public field of the protection mask grants read and execute privilege to the file. For instance, the user's default protection mask may be:

```
     s:rwe,p:rwe,g:rwe,o:rwed
```

If so, then the protection mask assigned to the new file is:

```
     s:rwe,p:re,g:rwe,o:rwed
```

Using Prompts

```
> copy :protection=p:re
Source          > flinch.exe
Destination     > _dc0/sysexe.users/ :owner=system
```

This performs the same function as the previous example.

displays that indicate what the utility is doing.)

Default The value specified by the OPTION command.

Syntax Type **:nolog** or **:log** to override the default.

:mod

Function Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.

Default :nomod

Syntax Type **:mod**

:multi

Function Use this switch to specify that when the destination volume becomes full the utility should dismount the destination volume and prompt the operator to insert an additional volume. Do not use this switch if any one of the source files is too large to fit on a single volume. Note that all subsequent volumes must have been initialized so they can be mounted correctly.

Default :nomulti

Syntax Type **:multi**

:owner=

Function Use this switch to designate the owner of the destination file(s) being created.

Default The UIC of the user.

Syntax Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC they represent.

:preserve=

Function Use this switch to preserve certain attributes of the original file when a new copy of it is created. The preservable aspects are creation date, owner, and protection.

Default The original file's attributes are not preserved in the new file.

Syntax Type **:preserve=** followed by any combination of the following keywords separated by commas: **created, owner, protection,** or **all.**

:protection=

Function Use this switch to specify the protection mask assigned to the file or files being created.

Default The user's default protection mask.

count

Functional Description

Use this command to display a report of the number of words, lines, and characters found in a text file.

Command Line Syntax

Mnemonic	count			
Required parameter	File List			
Switches				
Selection of files to be counted	:before= :mod	:since=	:exclude=	:uic=
Others	:log	:rewind	:pause	

Parameters

File List	Function	Required. Use this parameter to specify the names of the files to be counted.
	Default	None.
	Syntax	Type a list of file designations separated by commas. Wildcard symbols are allowed.

```
> copy reports/* * :preserve=created,protection
```

This command copies the latest version of all files from the directory named /REPORTS/ to the default directory. Each file has the same name in the default directory as it had in the original directory. The :preserve switch causes the new files to have the same creation date and the same protection mask as the original files. The owner of the original files is not preserved, i.e. the new files belong to the user who performs the COPY Command.

```
> copy users.john.*/ _dx0/*/* :builddir
```

This command copies the latest version of all the files found in directory /USERS.JOHN/ and all of its subdirectories to the device _DX0. The files are placed in directories with the same name as the original directory, e.g. /USERS.JOHN/, /USERS.JOHN.PROJECTS/, etc. The :builddir switch creates any destination directories that do not already exist. For example, you could initialize a diskette, mount it, and then execute the command shown above. All directories needed on the diskette will be created automatically.

```
> copy april.dat,may.dat,june.dat quarter.dat :concat
```

This command line copies the latest version of the files named APRIL.DAT, MAY.DAT and JUNE.DAT in the default directory into a single file named QUARTER.DAT. The three source files (APRIL.DAT, MAY.DAT, JUNE.DAT) are concatenated into a single destination file (QUARTER.DAT).

```
> copy users.elizabeth.*/ _dx0/*/ :edit=users.elizabeth:release :multi
:builddir
```

The preceding command line copies the latest version of all files found in the directory /USERS.ELIZABETH/ and all of its subdirectories to the device named _DX0. The names of the directories on _DX0 into which the files are copied are the same as the names of the directories on the default device, except that the starting directory is /RELEASE/ instead of /USERS.ELIZABETH/. For instance, suppose there are two subdirectories of /USERS.ELIZABETH/ named /TIMECARDS/ and /VOUCHERS/. All files found in /USERS.ELIZABETH/ on the default device will be copied to _DX0/ RELEASE/. Files found in /USERS.ELIZABETH.TIMECARDS/ will be copied to _DX0/RELEASE.TIMECARDS/, and files found in /USERS.ELIZABETH.VOUCHERS/ will be copied to _DX0/RELEASE.VOUCHERS/. The :builddir switch creates the destination directories automatically.

The :multi switch is used to span multiple volumes. For instance, suppose all of the files found in /USERS.ELIZABETH/ do not fit on a single diskette. The command above copies files to the diskette until the diskette is filled up. Then this prompt appears:

:since=	Function	Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:since=** followed by a date and/or time in the standard syntax.

:uic=	Function	Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.
	Default	Selects all files that match the specified list.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.

Examples

> count action.doc

This command reports the number of lines, words, and characters found in the latest version of the file named ACTION.DOC in the default directory. It generates a report like this:

```
                                                                File Name
-------------------------------------------  ---------  --------  --------
_DS0/USERS.TERRY/ACTION.DOC.3                      2958       293        37
```

Note that a word is delimited by any number of the following characters: null ($00), tab ($09), space ($20), or any line terminator.

A line is delimited by line feed ($0A), form feed ($0C), vertical tab ($0B), or carriage return ($0D).

> count *.txt :uic=hudson

This command counts the characters, words and lines in the latest version of all files in the default directory having an extension of .TXT and belonging to the owner whose username is HUDSON. It generates a report like this:

Notes on Usage

When using COPY with tape drives, you must include the :rewind switch on the command line. This is especially important when you want to copy a file which precedes the COPY Command on the tape.

When you want to move a file from one directory to another on a disk, and then remove the file from the original directory, use the REN (rename a file) Command instead of COPY. This helps to eliminate redundant copies of a file scattered over the disk.

Related CIP Commands

backup	Incremental system backup
fstat	Inspect and change file status
ren	Rename a file
tcopy	Copy a file to or from a foreign tape

Functional Description

Use this command to create a directory.

Command Line Syntax

Mnemonic	crd	
Required parameter	Directory	
Switches		
Status of directory	:owner=	:protection=
Information displayed	:log	
Change default directory	:perm	

Parameters

Directory	Function	Required. Use this parameter to specify the directory to be created.
	Default	None.
	Syntax	Type the directory name. Do not specify the extension or version. (The version is always 1 and the extension is always DIR.)

Switches

:before=

Function Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.

Default All files that match the File List parameter are selected.

Syntax Type :before= followed by a date or time in the standard date and time syntax.

:exclude=

Function Use this switch to exclude files from the File List parameter.

Default All files that match the File List parameter are selected.

Syntax Type :exclude= followed by a list of file designations separated by commas. Wildcard symbols are allowed.

:log

Function Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

Default The value specified by the OPTION Command.

Syntax Type :nolog or :log to override the default.

:mod

Function Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.

Default :nomod

Syntax Type :mod

:pause

Function Use this switch to stop the display after each screen of information. The display resumes when the user presses any key.

Default The value specified by the OPTION Command.

Syntax Type :pause or :nopause to override the default.

:rewind

Function Use this switch to rewind the tape device.

Default :norewind

Syntax Type :rewind

Examples

```
> crd test/
```

This report appears on your screen:

```
     Directory _DS0/TEST/ created.
```

This command creates the directory /TEST/ as a first-level subdirectory of /ROOTDIR/ on the system disk no matter where you are in your system's directory structure.

The protection and owner of this directory are the defaults of the user who typed the CRD command.

Note that the default directory after this command is the same as before the command.

```
> crd test/ :perm

     Directory _DS0/TEST/ created.
```

The directory is created as in the previous example, except the default directory after this command is changed to _DS0/TEST/. In other words, you are moved to the newly created directory.

```
> crd users.invoices/ :protection=g:rwed,o:rwed,p:,s:r :owner=gary
```

This creates /INVOICES/ and places it in /USERS/ as a subdirectory. The protection on the directory /INVOICES/ is G:RWED,O:RWED,P:,S:R as specified by the :protection switch. The owner of the newly created directory is GARY.

Using Prompts

```
> crd
Directory    > test
Directory _DS0/TEST/ created.
```

This performs the same function as the first example.

```
                     File Name                         Chars     Words     Lines
-----------------------------------------------      -------   -------   -------
_DS0/USERS.HUDSON/SOFTWARE.TXT.3                        27704      3056       649
_DS0/USERS.HUDSON/ACTION.TXT.14                         13728      1673       426
_DS0/USERS.HUDSON/REQUEST.TXT.1                         18353      2004       565
  Total     3 files.                                    59785      6733      1640
```

Using Prompts

```
> count :uic=hudson
File list     > *.txt
```

This performs the same function as the previous example.

Notes on Usage

It is sometimes useful to count the lines of output generated by another command. For instance, you could pipe the output of PSTAT to COUNT. The number of lines reported is the number of processes active on the system.

```
     > pstat | count sys$input
```

This command generates the following kind of report, indicating there are 14 processes on the system. Note that _C03EB$02 is the name of the pipe file set up by CIP:

```
                     File Name                         Chars     Words     Lines
-----------------------------------------------      -------   -------   -------
_C03EB$02                                                 378        28        14
```

Related CIP Commands

checksum	Sum the bytes in a file
scan	Search file(s) for a pattern

create

Functional Description

Use this command to create a text file and enter its contents.

Command Line Syntax

Mnemonic	create
Required parameter	File Desig.
Switches	
Format control	:rawinput :formatout

Parameters

File Desig.	Function	Required. Use this parameter to specify the name of the file to be created.
	Default	None.
	Syntax	Type a file designation. Wildcard symbols are not allowed.

Switches

:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:perm	Function	Use this switch to move to the newly created directory.
	Default	:noperm
	Syntax	Type **:perm**
:owner=	Function	Use this switch to designate the owner (UIC) of the destination directory or directories being created.
	Default	The user's UIC.
	Syntax	Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC they represent.
:protection=	Function	Use this switch to specify the protection mask to be assigned to the new directory or directories.
	Default	The user's default protection mask.
	Syntax	Type **:protection=** followed by a protection mask.

Note that only those fields of the protection mask that are specified are altered. Fields not specified keep their original value. Suppose you typed this switch:

```
:protection=s:,o:rwe
```

This switch value allows the system no privilege, while the owner is granted read, write and execute privilege. The public and group privileges are unchanged.

Using Prompts

```
> create
File desig.   > assign.tmp
```

This performs the same function as the first example.

Notes on Usage

CREATE can also be used as a tool for "catching" ASCII text coming in a serial communications line. For instance, type:

```
     > create input.dat < _tt17
```

All data received on _TT17 is placed in the file named INPUT.DAT in the default directory. CREATE finishes when a [CTRL] z (which signals end-of-file) is entered on _TT17.

Related CIP Commands

vew	Edit a text file

Notes on Usage

The dot, ., preceding a directory name creates a subdirectory in the default directory. You can create subdirectories to any level, i.e., you can have as many levels of subdirectories as you wish.

If a subdirectory for a directory does not already exist, it is created. Suppose you typed this command:

```
     > crd .test.test2.test3
```

A report like this would appear:

```
     Directory _DS0/USERS.TEST.TEST2.TEST3/ created.
```

If TEST and TEST2 did not previously exist, then TEST, TEST2 and TEST3 are all created.

You cannot specify an extension or version number for the new directory.

You cannot create a directory on a tape. However, you can include a directory name (or pathname) as part of a file's designation when you write a file to a tape.

Related CIP Commands

def	Display the name of the default device and the default directory
cd	Change the default directory and device

crypt

Functional Description

Use this command to encrypt or decrypt a file or set of files.

Command Line Syntax

Mnemonic	crypt
Required Parameters	Source Destination Key (except when encrypting using :public switch)

Switches

Function specification switches	:fast :encrypt	:des :user=	:private	:public
File selection switches	:before= :confirm	:mod :since=	:exclude=	:uic=
Destination specification switches	:builddir	:edit=		
Destination attribute switches	:owner=	:preserve=	:protection=	
Display switches	:log			

Switches

:rawinput

Function: Use this switch to specify that the input to the command is to be read in "raw" mode, that is that characters are not echoed to the screen, and all characters pressed are inserted directly into the file.

Default: :norawinput

Syntax: Type **:rawinput**

:formatout

Function: Use this switch to specify that characters written to the file are to be "formatted." That means that on TTY class devices, line feeds are transformed to carriage return-line feeds.

Default: :noformatout

Syntax: Type **:formatout**

Examples

```
> create assign.tmp
```

This command creates a data file named ASSIGN.TMP in the default directory. The cursor will appear at the front of the next line. At this point you may type any data you want to enter into the file. Type as many lines as you like. You can use [DEL] to erase characters entered on this line only. You cannot use [DEL] to erase characters on a previous line, i.e. once you press [RETRN], you cannot make any changes on that line.

When you finish typing, press [CTRL] z.

```
> create clean.doc :rawinput
```

In this example, the :RAWINPUT Switch enters the data in "raw" mode to the file CLEAN.DOC. That is, as characters are typed, they are not displayed on the screen. Also, [DEL] may not be used to erase characters.

Switches

:before=

Function Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.

Default All files that match the File List parameter are selected.

Syntax Type **:before=** followed by a date or time in the standard date and time syntax.

:builddir

Function Use this switch to create destination directories automatically if they do not exist.

Default :nobuilddir

Syntax Type **:builddir**

:confirm

Function Use this switch to confirm or deny the desired operation on each file or device.

Default :noconfirm

Syntax Type **:confirm**

:des

Function Use this switch to specify that the source file is to be encrypted using the U.S. National Bureau of Standards Data Encryption Standard. It may be used in conjunction with :private or :public but not with :fast. This switch has no effect if the source file is being decrypted.

Default :nodes

Syntax Type **:des**

:edit=

Function Use this switch to specify a series of edits to be performed on the destination file designation. The edits are specified as a series of "find and switch" strings. The destination file designation is searched for the first occurrence of the "find" string. If it is found, it is replaced by the "switch" string.

Default No edits are performed.

Syntax Type **:edit=** followed by a list of "find and switch" strings separated by commas. The find string is separated from the switch string by a colon. Either one or both of the strings may be empty.

:preserve=

Function — Use this switch to preserve certain attributes of the original file when a new copy of it is created. The preservable aspects are creation date, owner, protection.

Default — The original file's attributes are not preserved in the new file.

Syntax — Type **:preserve=** followed by any combination of the following keywords separated by commas: CREATED, OWNER, PROTECTION, or ALL.

:private

Function — Use this switch to specify that the source file is to be encrypted using the RSA private key belonging to the current user. If the :user= switch is specified, the private RSA key belonging to the user specified is used instead. If the :des switch is specified in conjunction with this switch, the actual file is encrypted using DES. Otherwise, the file is encrypted using the polyalphabetic method. Using the :private switch requires a key to be supplied which is the same key used when the RSA keys were generated using the KEYGEN utility. This switch has no effect if the source file is being decrypted.

Default — :noprivate

Syntax — Type **:private**

:protection=

Function — Use this switch to modify the protection mask associated with the file.

Default — The owner's protection mask.

Syntax — Type **:protection=** followed by a protection mask. Note that only those fields of the protection mask that are specified are altered. Fields not specified keep their original value. For example, the switch :PROTECTION=S:,O:RWE allows the system no privilege, while the owner is granted read, write and execute privilege. The public and group privileges are unchanged.

:public

Function — Use this switch to specify that the source file is to be encrypted using the RSA public key belonging to the current user. If the :user= switch is specified, the public RSA key belonging to the user specified is used instead. If the :des switch is specified in

Parameters

Source	Function	Required. Use this parameter to specify the names of the files to be encrypted or decrypted.
	Default	None.
	Syntax	Type a list of filenames separated by commas. Wildcard symbols are allowed.
Destination	Function	Required. Use this parameter to specify the names of the files to which the source files are encrypted or decrypted. Field substitution is allowed. That is, everywhere an asterisk, *, appears in the value typed for this parameter, the corresponding field from the corresponding source file is substituted.
	Default	None.
	Syntax	Type a list of filenames separated by commas. Wildcard symbols are allowed.
Key	Function	Required. Use this parameter to specify the password phrase used to decode a private RSA key, or to encrypt or decrypt using either the polyalphabetic or DES algorithms. The only time this parameter is not required is when encrypting or decrypting using a public RSA key. The minimum length of the key is 16 bytes. If you specify a shorter key, you will be prompted for a key that is long enough. NOTE: The key is used for encryption/decryption on ALL the files in the source file list. If the source files were encrypted with various keys and/or algorithms, then decryption of all of them with the same key is certain to fail. This also applies to a collection of source files, some of which are currently encrypted, others of which are currently non-encrypted. The same key will be used for both encryption and decryption duties.
	Default	None.
	Syntax	Type a word or phrase.

```
> crypt memo.txt *.enc :public :user=john :des :encrypt
```

This command will encrypt the file called "memo.txt" to a file called "memo.enc" using the public key belonging to user "john" on the current node. The actual data in the file are encrypted using the DES algorithm.

```
> crypt memo.enc memo.txt "This is my public/private key" :user=john
```

If used on the file "memo.enc" encrypted with the previous command, this invocation of CRYPT will decrypt "memo.enc" back to "memo.txt". The knowledge that the file was previously encrypted with DES and John's public key is contained in the file "memo.enc". At the time John's public/private keys were created using the KEYGEN utility, the password phrase "This is my public/private key" was supplied to KEYGEN and was used to encode John's private key.

Using Prompts

```
> crypt :encrypt
Source      > myfile.dat,yourfile.txt
Destination > *.enc  :des
Key         > my key is abracadabra
```

This uses the DES method to encrypt the files "myfile.dat" and "yourfile.txt" to files called "myfile.enc" and "yourfile.enc" using the key "my key is abracadabra". Note that the key is not echoed if it is entered in response to a prompt.

Notes on Usage

CRYPT allows a user to encrypt or decrypt files using a fast polyalphabetic algorithm, the widely accepted U.S. National Bureau of Standards Data Encryption Standard (DES), or a version of the Rivest, Shamir, and Adleman (RSA) public/private key algorithm. Strengths and weaknesses of each are briefly outlined below. Detailed discussions of each encryption method are available in publicly accessible literature.

The intent of this utility is to provide the user with a means of rendering sensitive files unusable to those not in possession of the key used to encrypt the file. In addition, the RSA public/private key algorithm provides a means to send a message with a "digital signature"

:encrypt	Function	Use this switch to specify that encryption is always to take place, even if the source file is already an encrypted file.
	Default	Encrypted source files are decrypted and non-encrypted source files are encrypted.
	Syntax	Type **:encrypt**
:exclude=	Function	Use this switch to exclude files from the File List parameter.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of file designations separated by commas. Wildcard symbols are allowed.
:fast	Function	Use this switch to specify that the source file is to be encrypted using the polyalphabetic method. It may be used in conjunction with :private or :public but not with :des. This switch has no effect if the source file is being decrypted.
	Default	:fast
	Syntax	Type **:nofast**
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**
:owner=	Function	Use this switch to change the owner (UIC) of the file.
	Default	The UIC assigned when the file was created or last updated.
	Syntax	Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC that they represent.

method is helpful when sending secure messages between non-cooperating parties. It also has the property of being able to include a digital signature. The method is based in modular arithmetic theory and the manipulation of very large prime numbers which compose the keys. The desirable feature of the method is that one key is used to encrypt a message and another key is used to decrypt it. Although the keys are obviously related, it is very difficult to determine one key from the other. One key is designated a public key and is available to anyone who wishes to use it. The other is designated as a private key. Although both the public and the private keys are kept in the same file which is readable by all, the private key is kept private by means of encrypting it using DES and a password phrase supplied by the owner of the keys at the time of their creation (See the command description for KEYGEN).

In order to use the public/private algorithm to send a secure message without exchanging keys with the destination party, encrypt the file using the :public switch and specify the intended receiver with the :user= switch. Once the file is encrypted, the intended receiver (and only the ıntended receiver) can decrypt the file. Not even the sender can recover the file.

In order to send a message with a digital signature, encrypt the file using the :private switch. Note that because only a person possessing the password phrase to the private key can successfully use the private key, any message which can be successfully decrypted using the corresponding public key had to have been encrypted by one knowing the password phrase. If one keeps the password phrase private, the successful decryption of such a file implies a digital signature by the encryptor. Note that file encrypted to this point is still decryptable by anyone having access to the public key.

In order to now make the file readable only to the intended receiver, it is necessary to superencrypt the file using the intended receiver's public key. This is done using the :public, :encrypt, and :user= switches. The intended receiver will be able to decrypt the file using first his own private key and then the sender's public key. The username of the owner of the keys used to encrypt a file using public/private keys is stored with the file and will be displayed at decryption time when a public key is used for decryption.

CRYPT will prompt for the password phrase used during private key encryption/decryption if it has not already been included as the key on the command line. Because the RSA method uses extreme amounts of CPU time, only certain file header information is encrypted using the public/ private keys. The actual file is encrypted using either DES or the polyalphabetic method as specified by the user.

		conjunction with this switch, the actual file is encrypted using DES. Otherwise, the file is encrypted using the polyalphabetic method. Using the :public switch is the only time that a key is not required for encryption. This switch has no effect if the source file is being decrypted.
	Default	:nopublic
	Syntax	Type **:public**
:since=	Function	Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:since=** followed by a date or time in the standard syntax.
:uic=	Function	Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.
	Default	All files that match the specified list are selected.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.
:user=	Function	Use this switch in conjunction with the :private or :public switch to specify which user's RSA keys to use.
	Default	:user=SYS$USERNAME
	Syntax	Type **:user=** followed by a valid username, optionally prepended by a valid nodename.

Examples

```
> crypt *.mem * "This is a crypt test"
```

This command will encrypt/decrypt all files with an extension of "mem" using the polyalphabetic method and using as its key "This is a crypt test".

dealloc

Functional Description

Use this command to deallocate a device which has been allocated or reserved.

Command Line Syntax

Mnemonic	dealloc		
Required parameter	Devicename		
Switches			
Prompting control	:auto	:confirm	
Device selection	:class=	:exclude=	:uic

Parameters

Devicename	Function	Required. Use this parameter to specify the devices which are to be deallocated.
	Default	None.
	Syntax	Type a list of devicenames.

or send a secure message using a publicly available key. (Digital signatures are discussed later.)

CRYPT encrypts or decrypts depending upon the file type of the source file. If the source file is of any type other than ENCRYPTED, the utility will encrypt the file. If the source file is already of file type ENCRYPTED, the utility will decrypt the file using the appropriate inverse of the method used to encrypt the file. A file may be superencrypted by specifying the :encrypt switch in conjunction with a source file that is already of type ENCRYPTED. This feature is useful in sending a secure file with a digital signature attached.

When a file is encrypted, a file checksum is included with the file. When a file is decrypted, the checksum is compared with the decrypted data. Should a mismatch occur, the user is notified. The reason for a mismatch is typically that the same key was not used for both encryption and decryption.

Polyalphabetic

The **polyalphabetic** algorithm uses the supplied key to encrypt the first 16 bytes of the source file. Thereafter, the previous 16 bytes in the source file are used as the key to encrypt each successive byte in the file. Thus, the key used to encrypt the file is as long as the file itself. This technique is often referred to as "autokeying." Because long repetitive strings of characters (especially zeros) have a tendency to reveal the key, CRYPT also automatically scans the source file for repeated characters and compresses them before encryption. For this reason, it is not uncommon for an encrypted file to be shorter than the source file from which it was derived. For this same reason, it is probable that a decrypted file will be larger than the encrypted file from which it was derived. Certain character sequences can also cause the encrypted file to be larger than the source file.

DES

The **DES** algorithm uses the supplied key to create a 56-bit key used as described in the Data Encryption Standard published by the U.S. National Bureau of Standards. It is regarded as being sufficiently secure as to be widely used in the commercial banking and finance industry. Although it is roughly one-fifth as fast as the polyalphabetic scheme, it is included for users who feel the additional security is worth the overhead.

RSA

The **RSA public/private key** algorithm as first described by Rivest, Shamir, and Aldeman is included because of its unique properties. This

Examples

```
> dealloc _TT0
```

This command deallocates device _TT0.

```
> dealloc * :confirm :class=tty* :uic=system
```

This command deallocates all allocated devices which are either class TTY or TTYSPECIAL devices and which are owned by the username SYSTEM. It lists each device and prompts for a confirmation before deallocating it. The user must have world privileges in order to deallocate devices not owned by the deallocating process.

Using Prompts

```
> dealloc
device name    > _TT0
```

This performs the same function as the first example.

Notes on Usage

DEALLOC is normally used to return an unneeded device to a pool of public resources. It is not absolutely necessary to deallocate devices allocated to a CIP before logging off. This is because WMCS automatically deallocates any devices allocated to a process during final exit. Deallocating devices after their need is completed gives other users an opportunity to use them.

Related CIP Commands

keygen	Generate a set of RSA public/private keys

def

Functional Description

Use this command to display the name of the default device and the default directory.

Command Line Syntax

Mnemonic def

Parameters

None.

Switches

None.

Switches

:auto	Function	Use this switch to perform the desired operation without any confirmation.
	Default	:noauto
	Syntax	Type **:auto**
:confirm	Function	Use this switch to confirm or deny the desired operation on each device.
	Default	:noconfirm
	Syntax	Type **:confirm**
:class=	Function	Use this switch to select only those devices that match the Devicename parameter and belong to the specified class of devices.
	Default	All devices that match the Devicename parameter are selected.
	Syntax	Type **:class=** followed by a list of class types in standard device class syntax. For example, :class=DISK,TTY*,PIPESPC.
:exclude=	Function	Use this switch to exclude devices from the Devicename parameter.
	Default	All devices that match the Devicename parameter are selected.
	Syntax	Type **:exclude=** followed by a list of device designations separated by commas. Wildcard symbols are allowed.
:uic=	Function	Use this switch to select only those devices that match the Devicename parameter and are owned by the specified user or list of users.
	Default	All devices that match the Devicename parameter are selected.
	Syntax	Type **:uic=** followed by a list of UICs in the standard UIC syntax, or by a list of usernames.

Functional Description

Use this command to remove files from a disk.

Command Line Syntax

Mnemonic		del			
Required parameter		File List			
Switches					
	File selection switches	:before= :exclude=	:since= :mod	:confirm	:uic=
	Other switches	:auto	:log	:override	:zerofiles

Parameters

File List	Function	Required. Use this parameter to specify the files to be deleted.
	Default	None.
	Syntax	Type a list of file designations separated by commas. Wildcard symbols are allowed.

Related CIP Commands

alloc	Find and reserve an available device
rotor	Add, delete or list devices in a rotor list

:override	Function	Use this switch to delete the directory even though it still contains files. Note that this is a sensitive operation because when you delete a directory that contains files, the files in that directory are not deleted but they become inaccessible. The space consumed by those files cannot be reused until the files themselves are deleted.
	Default	:nooverride
	Syntax	Type **:override**
:since=	Function	Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:since=** followed by a date or time in the standard syntax.
:uic=	Function	Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.
:zerofiles	Function	Use this switch to instruct the disk class handler to zero a file (erase all data) before deleting it.
	Default	:nozerofiles
	Syntax	Type **:zerofiles**

Examples

> del aghast.prn

This command removes the latest version of the file named AGHAST.PRN from the default directory. Then this prompt appears:

```
    _DC0/USERS.JILL/AGHAST.PRN.3
    Delete (Y or N)? >
```

Examples

```
> def
```

This command generates the following display if your default directory is in /USERS.AL/ on the device _DS0:

```
_DS0/USERS.AL/
```

Using Prompts

None.

Related CIP Commands

dir	List the contents of a directory
cd	Change the default directory and device
crd	Create a directory

```
     Delete _DI0/USERS.BUNKY/SUMMARY.TMP.7 (Y or N)? >
```

As in the first example, you should type Y to remove the file, or N to preserve it. Any other characters are ignored. You do not need to press [RETRN] after your selection.

> del /users.mary.*/*.*.*

Use this command to delete all versions of all files found in /USERS.MARY/ and all of its subdirectories. Since the :auto switch was not specified, operator confirmation will be requested periodically for files found in each directory or each group of 22 files, whichever occurs first.

For instance, suppose /USERS.MARY/ has two subdirectories named /USERS.MARY.MEMOS/ and /USERS.MARY.SYSTEMS/, and that the MEMOS subdirectory has 25 files. DEL would begin by displaying the names of the first 22 files in /USERS.MARY.MEMOS/ and request operator confirmation of those files. If the operator presses 'Y', those 22 files are deleted. Next, DEL displays the names of the remaining 3 files in the /USERS.MARY.MEMOS/ directory, and requests confirmation of them. Then DEL moves on to the /USERS.MARY.SYSTEMS/ subdirectory, and displays the names of the files in that directory for confirmation. Finally, DEL displays the names of the files in the directory /USERS.MARY/ and requests confirmation.

> del summary.dir

This command deletes the directory file named SUMMARY.DIR. If /SUMMARY/ contains files, an error is displayed indicating that you cannot delete a non-empty directory file. In general, you should never delete a directory file that contains files since that makes those files inaccessible, and does not free up the space used by those files. If you want to delete a directory, you should first delete (or rename) all of the files in that directory.

If for some reason you do need to delete a non-empty directory file, you must use the :override switch. The following command allows you to delete the directory even though it may not be empty:

```
     > del summary.dir :override
```

Switches

:auto	Function	Use this switch to perform the desired operation without any confirmation.
	Default	:noauto
	Syntax	Type **:auto**
:before=	Function	Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:before=** followed by a date or time in the standard date and time syntax.
:confirm	Function	Use this switch to confirm or deny the desired operation on each file.
	Default	:noconfirm
	Syntax	Type **:confirm**
:exclude=	Function	Use this switch to exclude files from the File List parameter.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of file designations separated by commas. Wildcard symbols are allowed.
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**

dev

Functional Description

Use this command to display a list of mounted devices.

Command Line Syntax

Mnemonic	dev		
Optional parameter	Device List		
Switches			
Device selection switches	:class=	:exclude=	:uic=
Display format	:list :owner	:pause :username	:header

Parameters

Device List	Function	Optional. Use this parameter to specify which devices to display.
	Default	All mounted devices are displayed.
	Syntax	Type a list of devicenames separated by commas. Wildcard symbols are allowed.

At this point you should type either **Y** (to delete the file) or **N** (to preserve it). Any characters other than Y or N are ignored. Pressing [RETRN] after typing your selection is not required. If you press Y this message appears:

```
_DC0/USERS.JILL/AGHAST.PRN.3 Deleted.
1 files deleted.
```

> del *.tmp

This command deletes the latest version of all files in the default directory with an extension of .TMP. The following kind of report appears:

```
_DC0/USERS.JILL/CROSS.TMP.2
_DC0/USERS.JILL/ENGINE.TMP.14
_DC0/USERS.JILL/USERS.TMP.1
Delete (Y or N)? >
```

At this point you should type either Y (to delete the files) or **N** (to preserve them). Any characters other than Y or N are ignored. Pressing [RETRN] after typing your selection is not required. If you press Y the following messages appear:

```
_DC0/USERS.JILL/CROSS.TMP.2 Deleted.
_DC0/USERS.JILL/ENGINE.TMP.14 Deleted.
_DC0/USERS.JILL/USERS.TMP.1 Deleted.
```

> del *.tmp.* :auto

This command deletes all versions of all files in the default directory with an extension of .TMP. No operator confirmation is solicited. The following messages are displayed:

```
_DC0/USERS.JILL/CROSS.TMP.1 Deleted.
_DC0/USERS.JILL/CROSS.TMP.2 Deleted.
_DC0/USERS.JILL/ENGINE.TMP.11 Deleted.
_DC0/USERS.JILL/ENGINE.TMP.12 Deleted.
_DC0/USERS.JILL/ENGINE.TMP.13 Deleted.
_DC0/USERS.JILL/ENGINE.TMP.14 Deleted.
_DC0/USERS.JILL/USERS.TMP.1 Deleted.
```

> del *.tmp.* :confirm

This command deletes all versions of all files in the default directory with an extension of .TMP. In this case, however, the operator is requested to confirm the deletion of each file. For each candidate file the following kind of report appears:

Default — All devices that match the Device List parameter are selected.

Syntax — Type **:uic=** followed by a list of UICs in the standard UIC syntax, or the usernames of authorized system users. For example:

```
:uic=[7,2],[*,3],system,backup
```

:username

Function — Use this switch to display the username associated with the owner of each file.

Default — :nousername

Syntax — Type **:username**

Examples

```
> dev
```

This command generates a display of mounted devices similar to this:

```
Device listing of __SYSTEM
TTYSpecial      : _TT19         _TT21
TTY             : _TT0          _TT13         _TT17         _TT23
                  _TT10         _TT14         _TT18         _TT8
                  _TT11         _TT15         _TT20         _TT9
                  _TT12         _TT16         _TT22
Disk            : _DS0
Queue           : _PQ0          _PQ1
NonDev          : _NULL
```

The device class is listed along the left margin. Following the colon is a list of the names of all mounted devices in that class. If there are no mounted devices in a particular device class, the name of the device class is not listed.

```
> dev _tt(20-23) :class=tty*
```

This command generates a list of all mounted devices whose names range from _TT20 through _TT23 which belong to TTY or TTYSpecial classes.

```
Device listing of __SYSTEM
TTYSpecial      : _TT21
TTY             : _TT20         _TT22         _TT23
```

Using Prompts

```
> del
File list     > summary.dir :override
```

This performs the same function as the previous example.

Notes on Usage

You can only delete files from disk devices. You cannot delete files from tape devices.

If you delete a non-empty directory, the files that were in that directory may appear in directory /LOST/ the next time you run the RECOVER utility.

Related CIP Commands

dmnt	Dismount (remove) a device
pu	Purge extra versions of files

dinit

Functional Description

Use this command to initialize a disk or a tape.

Command Line Syntax

Mnemonic	dinit
Required parameters	Devicename Volume Label

Switches

Device selection	:driver=	:density=	:driveid=	:drivetype=
Attributes of new file system	:owner=	:ialloc=	:protection=	:alloc=
	:autoflush	:cache=	:usercache=	:devname=
	:fcbsize=	:fcbsector=	:readahead	:blocksize=
Type of initialization	:format	:filesys		

Parameters

Devicename	Function	Required. Use this parameter to specify which device you wish to initialize.
	Default	None.
	Syntax	Type a standard devicename.
Volume Label	Function	Required. Use this parameter to label the disk or tape.

Switches

:class=	Function	Use this switch to specify the class(es) of device(s) you wish to include.
	Default	:class=*
	Syntax	Type **:class=** followed by a list of device classes separated by commas. Wildcard symbols are allowed.
:exclude=	Function	Use this switch to exclude devices from the Device List parameter.
	Default	All devices that match the Device List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of devicenames, separated by commas. Wildcard symbols are allowed.
:header	Function	Use this switch to display column labels at the top of each column.
	Default	:noheader
	Syntax	Type **:header**
:list	Function	Use this switch to list in a column the names of the devices that match all the criteria.
	Default	:nolist
	Syntax	Type **:list**
:owner	Function	Use this switch to display the UIC of the owner of each file.
	Default	:noowner
	Syntax	Type **:owner**
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when the user presses any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:uic=	Function	Use this switch to select only those devices that match the Device List parameter and are owned by the specified user or list of users.

system (not the device in which it is mounted or the volume label).

Default — The name from the DEVCONFIG.xxx file for the corresponding drive number.

Syntax — Type **:devname=** followed by a name no longer than 8 characters.

:driveid=

Function — Use this switch to specify the drive ID for the device.

Default — The drive ID listed for this device in the SYS$DISK/ROOTDIR/DEVCONFIG.NNN file, where NNN is the system model (100, 156, 160, etc.).

Syntax — Type **:driveid=** followed by the drive number, the driver configuration, and the board number. For example, :driveid=0A0. The following is a list of driver configurations:

<u>Disks</u>	
SMD	"A" is the only choice
FLNT	"A" is 8-inch floppy
	"B" is 5.25-inch floppy
	"C" is 3.25-inch floppy
WD3	"A" is the hard disk
	"B" is the floppy
IMI	"A" is the only choice
<u>Tapes</u>	
ADEI	"A" is fixed blocks (1022 bytes/blk)
	"B" is variable blocks
CIPH	"A" is low density
	"B" is high density

:driver=

Function — Use this switch to specify the device driver for this device.

Default — The device driver listed in the SYS$DISK/ROOTDIR/DEVCONFIG.NNN file (where NNN = 100, 156, or 160).

Syntax — Type **:driver=** followed by the name of the device driver file. No wildcard symbols are allowed. The names of available drivers on your system are found in SYS$DISK/SYSDSR/.

:drivetype=

Function — Use this switch to set the drive type for this device.

Default — The drive type that is listed for this device in the SYS$DISK/ROOTDIR/DEVCONFIG.NNN file.

```
> dev _tt(20-23) :class=tty* :list
```

This command is the same as the example above except the names of the devices are listed in a column. This is especially useful when generating a list of devicenames for use in a command file or for use by a program. Following is a sample output. Note that only the devicenames are listed with no device class.

```
    _TT20
    _TT21
    _TT22
    _TT23
```

Using Prompts

None.

Notes on Usage

DEV is useful for checking if a particular device is mounted. For instance, you want to use the tape drive, but someone else's tape is loaded in the drive. You need to remove this tape to put yours in. Before doing so, type **dev** to see if the tape is mounted by another user.

Another example: Suppose you attach a terminal to the port named _TT3 and then try to log on. The computer doesn't respond. You can use the DEV Command to see if _TT3 is mounted as a TTY or as a TTYSpecial. Remember, you cannot log on to a terminal which is mounted TTYSpecial.

Related CIP Commands

dstat	Device status

Default :noformat
Syntax Type **:format**

:ialloc=

Function Use this switch to specify the number of sectors assigned to a file the first time data are written to the file. The sectors allocated are not necessarily contiguous.

Default :ialloc=20 (for devices with 1/2 Kbyte sector size) :ialloc=10 (for devices with 1 Kbyte sector size)

Syntax Type **:ialloc=** followed by an integer greater than zero.

:owner=

Function Use this switch to specify the UIC or username for the owner of the device.

Default The UIC of the user.

Syntax Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC that they represent.

:protection=

Function Use this switch to specify the protection mask for the device.

Default The user's protection mask.

Syntax Type **:protection=** followed by a protection mask.

NOTE: Only those fields of the protection mask that are specified are altered. Fields not specified keep their original value.

For example, :protection=s:,o:rwe allows the system no privilege, while the owner is granted read, write, and execute privilege. The public and group privileges are unchanged.

:readahead=

Function Use this switch to enable READAHEAD. (Readahead is the issuing of an asynchronous read-request for the next logical sector in a file.)

Default :noreadahead

Syntax Type **:readahead**

:usercache=

Function Use this switch to set the maximum number of sectors of disk cache that any process can consume at one time.

Default None.
Syntax Type a name up to 16 characters long.

Switches

:alloc=
Function Use this switch to specify the number of sectors added to a file each time it is extended. The sectors allocated are not necessarily contiguous.
Default :alloc=20 (for devices with 1/2 Kbyte sector size) :alloc=10 (for devices with 1 Kbyte sector size)
Syntax Type **:alloc=** followed by a positive integer.

:autoflush
Function Use this switch to enable automatic flushing of modified sectors in the disk cache and modified FCBs each time the disk is updated.
Default :noautoflush (If you are not running LOGFLUSH, you should enable :autoflush.)
Syntax Type **:autoflush**

:blocksize=
Function Use this switch to specify the number of bytes per block on a tape.
Default :blocksize=1024
Syntax Type **:blocksize=** followed by a positive integer less than or equal to 4096.

:cache=
Function Use this switch to set the number of sectors (disk cache) to be maintained in main memory to allow faster access to memory.
Default :cache=30 (disks) :cache=8 (tapes)
Syntax Type **:cache=** followed by a positive integer.

:density=
Function Use this switch to specify the density of a cipher tape.
Default :density=1600
Syntax Type **:density=** followed by one of the following: 1600 or 3200 (high density).

:devname=
Function Use this switch to set the boot-disk name. This is the name for the disk used to boot the

This command initializes tape _MT0, labels it WMCS_6.0, sets the blocksize at 4096 bytes per block, and adds public read and execute access to the owner's default protection mask.

Using Prompts

```
> dinit
device       > _dfl
Volume label  > wmcs_6.0
```

This performs the same function as the first example.

Notes on Usage

Devices to be initialized must not be mounted.

There is only one boot block on a disk and it is at sector 0.

Use the :nofilesys switch to format a disk without putting any other information on the disk.

Related CIP Commands

mnt	Mount a device
btup	Display or set disk boot blocks or tape labels

Syntax Type **:drivetype=** followed by one of the following valid drive types:

What you type	Description
FLOP09A	5.25-inch floppy
FLOP09B	5.25-inch floppy (5 sector)
FLOP015	8-inch floppy
WIN12	CMI 10 Mbyte 5.25-inch Winchester
WIN19	CMI 15 Mbyte 5.25-inch Winchester
WIN30	28 Mbyte 5.25-inch Winchester
WIN43	39 Mbyte 5.25-inch Winchester
SMD84B	76 Mbyte SMD disk
SMD168B	152 Mbyte SMD disk
SMD474B	421 Mbyte SMD disk
IMI20	20 Mbyte IMI disk
IMI40	40 Mbyte IMI disk

:fcbsector= Function Use this switch to specify the starting sector number for the FCB.SYS file (i.e., position the file system on the disk).

Default The file is positioned around the middle of the disk.

Syntax Type **:fcbsector=** followed by a positive integer.

:fcbsize= Function Use this switch to specify the size of the FCB file (i.e., specify the number of FCBs in the FCB.SYS file).

Default If the disk size is greater than 80 megabytes, then the number of FCBs equals the number of Kbytes on the disk divided by 20. If the disk size is less than or equal to 80 Mb, then the number of FCBs equals the number of Kbytes on the disk divided by 10. The minimum number of FCBs is 16.

Syntax Type **:fcbsize=** followed by a positive integer greater than or equal to 16.

:filesys Function Use this switch to put a file system as well as a boot block on the disk.

Default :filesys

Syntax Type **:nofilesys**

:format Function Use this switch to format the disk. (All disks must be formatted the first time they are initialized so the sectors can be used.)

Functional Description

Use this command to list the files on a disk or tape and to display information about those files.

Command Line Syntax

Mnemonic	dir			
Optional parameter	File List			
Switches				
Files to be included in the display	:before= :since=	:uic=	:exclude=	:mod
Format of the display	:full :list	:showall :suppress=	:header :path	:sort= :total
Contents of the display	:bytes :modified :username :protection	:filetype :size :fcb	:recordlen :extents :parent	:created :owner :fileid
Others	:pause	:rewind		

Default 25 percent of the total disk cache sectors, or 8 sectors, whichever is smaller.

Syntax Type **:usercache=** followed by an integer from 1 to 16.

Examples

```
> dinit _dfl wmcs_6.0
```

This command initializes the volume in drive _DF1 and labels it WMCS_6.0. If the volume is a brand new disk you may get the error message, "The specified sector was not found on the current track," because the disk isn't formatted. The disk must be formatted the first time it is initialized (as shown in the next example). If the devicename _DF1 is not defined in the ROOTDIR/DEVCONFIG.NNN (nnn = 100 or 150 or 156) file, you will get a device class error message. You must specify the driver and driveid (and if a disk, also the drivetype) as shown in the third and fourth examples.

```
> dinit _dfl wmcs_6.0 :format
```

This command formats and initializes the disk volume in drive _DF1 and labels it WMCS_6.0. Formatting is required the first time a disk is initialized. It consists of writing the sector marks on the disk so the sectors can be used. Formatting after the first initialization is optional.

```
> dinit _ct0 mcs_6.0 :driver=adei$100 :driveid=0b0
```

This command initializes a tape referred to by devicename _CT0 which requires driver SYS$DISK/SYSDSR/ADEI$100.DSR and is in the drive designated by driveid 0b0 (drive 0 on adei controller, configuration b, on board 0).

```
> dinit _dfl mcs_6.0 :noreadahead :cache=20 :owner=system
```

This command initializes disk _DF1, disables READAHEAD, sets the cache size at 20 sectors, and specifies SYSTEM as the owner of the disk.

```
> dinit _mt0 wmcs_6.0 :blocksize=4096 :protection=p:re
```

	Syntax	Type **:extents**
:fcb	Function	Use this switch to display the FCB number and the sequence number.
	Default	:nofcb
	Syntax	Type **:fcb**
:fileid	Function	Use this switch to display the file ID number.
	Default	:nofileid
	Syntax	Type **:fileid**
:filetype	Function	Use this switch to display the file type.
	Default	:nofiletype
	Syntax	Type **:filetype**
:full	Function	Use this switch to display all information about the file.
	Default	:nofull
	Syntax	Type **:full**
:header	Function	Use this switch to display column labels at the top of each column.
	Default	:noheader
	Syntax	Type **:header**
:list	Function	Use this switch to display each filename in the list in a single column, justified against the left margin of the screen. This switch is ignored when :full, :total, or any attribute switch is specified.
	Default	:nolist
	Syntax	Type **:list**
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**
:modified	Function	Use this switch to display the date and time at which the file was last modified.

contained no files that matched all of the criteria.
Default :noshowall
Syntax Type **:showall**

:since=
Function Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.
Default All files that match the File List parameter are selected.
Syntax Type **:since=** followed by a date or time in the standard syntax.

:size
Function Use this switch to display the logical and physical size of each file. Unless the :bytes switch is specified, the sizes are specified in Kbytes. If the :bytes switch appears, the size will be in bytes.
Default :nosize
Syntax Type **:size**

:sort=
Function Use this switch to specify the order in which the files will be accessed; files may be accessed in alphabetical order by filename, by file extension, in UIC order, or by creation date (oldest file first).
Default Files are accessed alphabetically by filename.
Syntax Type **:sort=** followed by one of the following keywords: **extension**, **uic**, or **date**.

:suppress=
Function Use this switch to display filenames without the version number, or without both the file extension and version number.
Default Filenames are displayed with extensions and version numbers.
Syntax Type **:suppress=** followed by one of the following: **version** or **extension**.

:total
Function Use this switch to display only the total number of files in the directory, and the physical and logical space they occupy.
Default :nototal
Syntax Type **:total**

Parameters

File List	Function	Optional. Use this parameter to specify the names of the files to be included in the display.
	Default	The most recent version of each file in the default directory is included in the display.
	Syntax	Type a list of filenames.

Switches

:before=	Function	Use tnis switch to select only those files that match the File List parameter and were created or modified before the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type :before= followed by a date or time in the standard date and time syntax.
:bytes	Function	Use this switch to display file sizes in bytes.
	Default	:nobytes
	Syntax	Type :bytes
:created	Function	Use this switch to display the date and time the file was created.
	Default	:nocreated
	Syntax	Type :created
:exclude=	Function	Use this switch to exclude files from the File List parameter.
	Default	All files that match the File List parameter are selected.
	Syntax	Type :exclude= followed by a list of file designations separated by commas. Wildcard symbols are allowed.
:extents	Function	Use this switch to display the number of extents that comprise the file.
	Default	:noextents

```
Directory listing of _DS0/USERS.ROBERT/
File name                Logical/  Physical
------------------    -------------------
ACCOUNTS.DIR.1              2.0/       2.0
CALENDAR.TXT.4             23.4/      24.0
CHECKLIST.DIR.1             1.0/       1.0
DOC.DIR.1                   4.0/       4.0
INVENTORY.DIR.1             3.0/       3.0
MEMOS.DIR.1                 3.0/       3.0
PERSONAL.DIR.1              1.0/       1.0
PRIMARY.DAT.24            192.8/     193.0
PROJECTS.DIR.1              2.0/       2.0
USERUP.COM.5                0.1/       1.0
WORK.DIR.1                  1.0/       1.0
            11 Files.     233.3/     235.0  Total blocks.
```

The name of the most recently created version of each file in the directory appears along the left margin of the screen. Any information about a file appears to the right of the filename.

Use :noheader when you do not want column headings. If the :noheader switch was not used in the previous command, the display would be as follows:

```
ACCOUNTS.DIR.1              2.0/       2.0
CALENDAR.TXT.4             23.4/      24.0
CHECKLIST.DIR.1             1.0/       1.0
DOC.DIR.1                   4.0/       4.0
INVENTORY.DIR.1             3.0/       3.0
MEMOS.DIR.1                 3.0/       3.0
PERSONAL.DIR.1              1.0/       1.0
PRIMARY.DAT.24            192.8/     193.0
PROJECTS.DIR.1              2.0/       2.0
USERUP.COM.5                0.1/       1.0
WORK.DIR.1                  1.0/       1.0
            11 Files.     233.3/     235.0  Total blocks.
```

The :size switch causes the logical and physical size of each file to appear next to the filename. The sizes are expressed in kilobytes. The :size switch also causes the summary line (11 Files.   233.3/   235.0 Total blocks.) to appear at the end of the display.

Many other file attributes can be displayed. These are listed next to the filenames in the order specified on the command line.

```
> dir :size :username :filetype
```

This command line generates a display similar to the one shown below:

Default :nomodified
Syntax Type **:modified**

:owner
Function Use this switch to display the UIC of the owner of each file.
Default :noowner
Syntax Type **:owner**

:parent
Function Use this switch to display the FCB number and sequence number of the directory file (parent directory) in which this file resides.
Default :noparent
Syntax Type **:parent**

:path
Function Use this switch to display the complete file designation (pathname) of each file.
Default :nopath
Syntax Type **:path**

:pause
Function Use this switch to stop the display after each screen of information. The display resumes when the user presses any key.
Default The value specified by the OPTION Command.
Syntax Type **:pause** or **:nopause** to override the default.

:protection
Function Use this switch to display the protection mask associated with each file.
Default :noprotection
Syntax Type **:protection**

:recordlen
Function Use this switch to display the record length of each file.
Default :norecordlen
Syntax Type **:recordlen**

:rewind
Function Use this switch to rewind the tape device.
Default :norewind
Syntax Type **:rewind**

:showall
Function Use this switch to display the names of directories which were inspected, but which

```
    Directory listing of _DS0/USERS.ROBERT/
    File name            Record Path
    -------------------- ------ ----------------------------------------
    ACCOUNTS.DIR.1           32 _DS0/USERS.ROBERT/ACCTS.DIR.1
    CALENDAR.TXT.4            1 _DS0/USERS.ROBERT/CALENDAT.TXT.1
    CHECKLIST.DIR.1          32 _DS0/USERS.ROBERT/CHECKLIST.DIR.1
    DOC.DIR.1                32 _DS0/USERS.ROBERT/DOC.DIR.1
    INVENTORY.DIR.1          32 _DS0/USERS.ROBERT/INVENTORY.DIR.1
    MEMOS.DIR.1              32 _DS0/USERS.ROBERT/MEMOS.DIR.1
    PERSONAL.DIR.1           32 _DS0/USERS.ROBERT/PERSONAL.DIR.1
    PRIMARY.DAT.24         1024 _DS0/USERS.ROBERT/PRIMARY.DAT.24
    PROJECTS.DIR.1           32 _DS0/USERS.ROBERT/PROJECTS.DIR.1
    USERUP.COM.5              1 _DS0/USERS.ROBERT/USERUP.COM.5
    WORK.DIR.1               32 _DS0/USERS.ROBERT/WORK.DIR.1
```

> dir :list :suppress=version

Use the :list switch to generate a simple list of filenames without attributes.

Use the :suppress= switch to remove either the file version number, or both the version number and the extension from the names displayed. The :suppress= switch may be used with any of the display formats.

```
    Directory listing of _DS0/USERS.ROBERT/
    ACCOUNTS.DIR
    CALENDAR.TXT
    CHECKLIST.DIR
    DOC.DIR
    INVENTORY.DIR
    MEMOS.DIR
    PERSONAL.DIR
    PRIMARY.DAT
    PROJECTS.DIR
    USERUP.COM
    WORK.DIR
```

> dir :full

This command displays all attributes of each file:

:uic=

Function — Use this switch to select only those files that are part of the given list and are owned by the specified user or list of users.

Default — All files that match the specified list are selected.

Syntax — Type **:uic=** followed by a list of UICs or usernames.

:username

Function — Use this switch to display the username associated with the owner of each file.

Default — :nousername

Syntax — Type **:username**

Examples

> **dir**

This command generates the following kind of report (if the default directory is _DS0/USERS.ROBERT/):

```
Directory listing of _DS0/USERS.ROBERT/
ACCOUNTS.DIR.1      DOC.DIR.1           PERSONAL.DIR.1      USERUP.COM.5
CALENDAR.TXT.4      INVENTORY.DIR.1     PRIMARY.DAT.24      WORK.DIR.1
CHECKLIST.DIR.1     MEMOS.DIR.1         PROJECTS.DIR.1
```

Note that the display's first line tells you the name of the device and the directory to which the files are assigned. The filename, file extension, and version number of the most recently created version of each file are displayed in alphabetical order.

> **dir :size**

This command generates the following kind of report:

```
_DS0/USERS.ROBERT/ACCOUNTS.DIR.1
Logical :        2.0  FCB.SEQ  :   7141.7169   Created : 09-Aug-1985 14:02:14
Physical:        2.0  Parent   :   1329.1332   Modified: 29-Dec-1985 21:00:49
Extents :          2  Filetype :   Directory   Owner   : [0007,0002]
Record  :         32  Fileid   :       0       Username: ROBERT
Protection: S:  RE,P:  R ,G:DWRE,O:DWRE
```

dispatch

Functional Description

Use this command to route a single input to multiple outputs.

Command Line Syntax

Mnemonic	dispatch			
Optional parameter	File List			
Switches				
Control flow of data	:infinite	:sysout	:blocksize=	:master

Parameters

File List	Function	Optional. Use this parameter to specify a list of files (including names of terminal ports) where you want the output to appear. Output automatically goes to DISPATCH's sys$output unless :nosysout is specified.
	Default	None.
	Syntax	Type a list of filenames separated by commas (terminal port names can be included). Wildcard symbols are not allowed.

```
Directory listing of _DS0/USERS.ROBERT/
File name                 Logical/  Physical Username   Filetype
------------------- -------------------- --------- ------------

ACCOUNTS.DIR.1               2.0/        2.0 ROBERT    Directory
CALENDAR.TXT.4              23.4/       24.0 ROBERT    Data
CHECKLIST.DIR.1              1.0/        1.0 ROBERT    Directory
DOC.DIR.1                    4.0/        4.0 ROBERT    Directory
INVENTORY.DIR.1              3.0/        3.0 ROBERT    Directory
MEMOS.DIR.1                  3.0/        3.0 ROBERT    Directory
PERSONAL.DIR.1               1.0/        1.0 ROBERT    Directory
PRIMARY.DAT.24             192.8/      193.0 ROBERT    Encrypted
PROJECTS.DIR.1               2.0/        2.0 ROBERT    Directory
USERUP.COM.5                 0.1/        1.0 ROBERT    Data
WORK.DIR.1                   1.0/        1.0 ROBERT    Directory
           11 Files.       233.3/      235.0  Total blocks.
```

The ordering of the attribute switches in this example is 1) :size, 2) :username and 3) :filetype. The order in which the attribute switches are specified on the command line determines the order in which the attributes are displayed.

```
> dir :path
```

Use this switch to obtain a list of complete file designations as shown here:

```
_DS0/USERS.ROBERT/ACCOUNTS.DIR.1
_DS0/USERS.ROBERT/CALENDAR.TXT.4
_DS0/USERS.ROBERT/CHECKLIST.DIR.1
_DS0/USERS.ROBERT/DOC.DIR.1
_DS0/USERS.ROBERT/INVENTORY.DIR.1
_DS0/USERS.ROBERT/MEMOS.DIR.1
_DS0/USERS.ROBERT/PERSONAL.DIR.1
_DS0/USERS.ROBERT/PRIMARY.DAT.24
_DS0/USERS.ROBERT/PROJECTS.DIR.1
_DS0/USERS.ROBERT/USERUP.COM.5
_DS0/USERS.ROBERT/WORK.DIR.1
```

Note that the line specifying the device and the directory (found at the top of the other displays) is missing in this display. This display is especially useful in creating a list of files to be used by programs or in command files. It contains no "extra" information.

```
> dir :recordlen :path
```

The :path switch may also be used as a file attribute. If any attributes (in this case :recordlen) are specified along with the :path switch, then the display looks like this:

Examples

```
> dir|dispatch dir.txt
```

This command takes the output of the DIR command (a list of the files in the default directory) and writes it to the file DIR.TXT (in the default directory) as well as to the terminal screen (sys$output).

```
> dir|dispatch dir.txt,dir.rpt,dir.doc :nosysout
```

This command takes the output of the DIR command and writes it to the files DIR.TXT, DIR.RPT, and DIR.DOC, but not to the terminal screen (sys$output).

```
> dir|dispatch _tt1,_tt2,_tt3 :master
```

This command takes the output of the DIR command and writes it to the terminal screen (sys$output) as well as to the other terminals _TT1, _TT2, and _TT3. Because :master is specified, if [CTRL] s is pressed (to stop the output from scrolling off the screen) on sys$output, then the output also stops on _TT1, _TT2, and _TT3. Output resumes on all the terminals when [CTRL] q is pressed on sys$output.

```
> mnt _pipe1 :driver=sys$pipe

     > &dispatch _tt1,report.doc :nosysout :infinite <_pipe1
     > dir >_pipe1
     > pstat >_pipe1
```

This command mounts a pipe, _PIPE1. The ampersand, &, tells DISPATCH to run in the background with its input coming from _PIPE1 and writing to _TT1 and REPORT.DOC (but not to sys$output). At the time DISPATCH is invoked, no one is writing to _PIPE1 so :infinite instructs DISPATCH to stay around (until explicitly killed by another process such as by PSTAT) forever, always trying to read from the pipe. When the DIR command is executed (with its output redirected to _PIPE1), DISPATCH writes the directory listing to REPORT.DOC and _TT1. When the next command, PSTAT, is executed, it is also writing to the pipe, so its display is read by DISPATCH and written to _TT1 and REPORT.DOC.

```
_DS0/USERS.ROBERT/ACCOUNTS.DIR.1
Logical :        2.0  FCB.SEQ  :     7141.7169   Created : 09-Aug-1985 14:02:14
Physical:        2.0  Parent   :     1329.1332   Modified: 29-Dec-1985 21:00:49
Extents :          2  Filetype :    Directory    Owner   : [0007,0002]
Record  :         32  Fileid   :         0       Username: ROBERT
Protection: S:  RE,P:  R ,G:DWRE,O:DWRE
```

The file's logical size is expressed in kilobytes and is rounded to the nearest tenth. This value represents the actual number of bytes in the file. (Contrast this with the physical size described below.)

The FCB.SEQ Field contains the FCB (File Control Block) number and the sequence number for this file. The FCB is the numeral on the left side of the decimal point. It represents the record number within the FCB.SYS file of the record describing this file. The sequence number is on the right side of the decimal point.

The Created Field shows the date and time the file was created.

Using Prompts

None.

Related CIP Commands

fstat	File status

Functional Description

Use this command to display a map of memory usage.

Command Line Syntax

Mnemonic	dm	
Optional parameters	Refresh Rate Memory Type	
Switches		
Memory to be displayed	:megabyte=	
Process to be displayed	:pid=	
Others	:setupin=	:setupout=

Parameters

Refresh Rate	Function	Optional. Use this parameter to specify the interval (in tenths of seconds) for updating the map.
	Default	10
	Syntax	Type a number from 1 to 10.
Memory Type	Function	Optional. Use this parameter to specify which type of memory to display. You can display two types of memory:

Switches

:blocksize= Function Use this switch to specify the number of bytes read at one time from DISPATCH's input and written to its outputs.

Default :blocksize=256

Syntax Type **:blocksize=** followed by an integer from 256 to 1024.

:infinite Function Use this switch to specify that DISPATCH live forever (until explicitly killed by another process, such as by PSTAT), always reading from its sys$input and writing to its outputs.

Default :noinfinite

Syntax Type **:infinite**

:master Function Use this switch to specify that sys$output is the "master" of all of DISPATCH's outputs. Normally, if one of the outputs sends a [CTRL] s, output to all of DISPATCH's outputs halts until the one that sent the [CTRL] s sends a [CTRL] q. With :master, data are still sent to all outputs except the one that sent [CTRL] s, and that output loses data until [CTRL] q is sent. Only sys$output can [CTRL] s without losing data.

Default :nomaster

Syntax Type **:master**

:sysout Function Use this switch to specify that DISPATCH write the input to its sys$output as well as to its specified list of outputs.

Default :sysout

Syntax Type **:nosysout**

Syntax Type **:setupout=** followed by a filename.

Examples

> dm

This kind of map appears on the screen when you execute DM:

```
                                    Memory Display of Meg 2   21-Dec-1985 13:50:19
------------------------------------------------------------------------------------
MCS  |MCS  |_DS0|_DS0|_DS0|vew_|Que_|link|vew_|_DF1|_DX1|CIP_|_DC0|CIP_|pass|REST
MCS  |MCS  |_DS0|_DS0|_DS0|vew_|Que_|link|vew_|_DF1|Dynm|dm_G|_DC0|CIP_|pass|REST
MCS  |MCS  |_DS0|_DS0|_DS0|Dynm|Que_|dm_G|vew_|_DF1|_DX1|dm_G|_DC0|CIP_|pass|REST
MCS  |MCS  |_DS0|_DS0|_DS0|CIP_|Que_|Dynm|CIP_|_DF1|_DX1|CIP_|sp_C|pass|pass|REST
-----+-----+----+----+----+----+----+----+----+----+----+----+----+----+----+----
MCS  |MCS  |_DS0|Dynm|_DS0|CIP_|Que_|_DX0|_DX0|_DF1|CIP_|Dynm|dm_G|pass|REST|REST
MCS  |MCS  |_DS0|_DS0|Dynm|CIP_|Que_|CIP_|_DX0|Dynm|CIP_|Logf|pass|pass|pass|REST
MCS  |MCS  |_DS0|_DS0|****|CIP_|Que_|CIP_|CIP_|_DF1|CIP_|Dynm|pass|@upa|REST|REST
MCS  |vew_|_DS0|_DS0|****|CIP_|link|CIP_|CIP_|_DF1|CIP_|CIP_|pass|@upa|REST|REST
-----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----
MCS  |vew_|_DS0|_DS0|dm_G|CIP_|link|CIP_|CIP_|_DF1|Dynm|CIP_|pass|CIP_|REST|REST
MCS  |vew_|_DS0|_DS0|dm_G|CIP_|link|CIP_|CIP_|_DX1|CIP_|CIP_|pass|CIP_|pass|REST
MCS  |Dynm|_DS0|_DS0|link|CIP_|link|vew_|CIP_|vew_|CIP_|CIP_|Dynm|CIP_|pass|REST
MCS  |pass|_DS0|_DS0|link|link|CIP_|vew_|Dynm|vew_|CIP_|dm_G|pass|pass|pass|REST
-----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----
MCS  |Dynm|_DS0|_DS0|link|****|CIP_|Dynm|CIP_|vew_|_DX1|dm_G|dm_G|pass|pass|REST
MCS  |_DS0|_DS0|_DS0|link|Dynm|CIP_|vew_|CIP_|vew_|_DX1|dm_G|dm_G|pass|pass|REST
MCS  |_DS0|_DS0|_DS0|vew_|Que_|link|vew_|CIP_|vew_|CIP_|dm_G|sp_C|pass|pass|REST
MCS  |Dynm|_DS0|_DS0|vew_|Que_|link|vew_|CIP_|_DX1|_DX1|_DC0|@upa|pass|pass|REST
------------------------------------------------------------------------------------
```

This map represents what is stored in each page of memory in megabyte 2. A page of memory is 4 kilobytes. Each column has 4 cells divided into four 4-kilobyte pages of memory. In other words, each column has 16 entries representing 16 pages of memory. Thus the WMCS (in the foregoing map) occupies the first 23 pages of memory in megabyte 2.

The first line indicates the megabyte number illustrated by the map, plus the date and time. The report is updated every second. Megabyte 2 represents the first megabyte of physical memory. Megabytes numbered 0 and 1 represent logical memory.

Read the map by beginning in the upper left-hand corner, reading down each column and progressing from left to right across the map.

Using Prompts

None.

Notes on Usage

You can use DISPATCH to send the output from any command or program (including entire command files) to as many files and terminal ports as you like.

Related CIP Commands

pstat	Display or set process attributes

If DM has already been executed and you want a map of a particular process, strike [ESC] then type the hexadecimal PID number or the process name. For example, if you execute DM and strike [ESC] this prompt appears at the bottom of the screen:

```
     Enter PID:
```

Type the PID or process name for the memory map you want displayed on the screen and strike [RETRN].

If you change your mind after the prompt appears, strike [RETRN] again.

```
> dm 10 0 :pid=pass1_user
```

```
Proc=pass1_USER   Pid=FFFF1A8C  Memory Display of Meg 0   21-Dec-1985 13:50:19
--------------------------------------------------------------------------------
 rps| rp |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp | rp |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp | rp |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp | rp |    |    |    |    |    |    |    |    |    |    |    |    |    |
----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
 rp |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
--------------------------------------------------------------------------------
```

The first parameter you type is interpreted as the number of tenth-of-a-second intervals in which the map on the screen is updated.

The next parameter specifies which megabytes of memory to display.

The :pid= switch specifies which process to display.

Line 1 of this report tells you the map is displaying pages in the first megabyte of memory (Meg 0) allocated to the process (PROC=) PASS1_USER with a process identification number (PID=) of FFFF1A8C. The date and time are also on this line.

This is a list of the values that may appear in a map for megabytes 0 or 1 (logical memory):

Blanks The page does not exist. (Note that most of the pages in the foregoing map are blank. This literally means that the first megabyte of memory does not exist. In other words, it is logical, or conceptual only.)
r The page has been assigned to the specified process.
o The page is owned by the specified process.
s The page is shared with another process.
p The page is write-protected.
n The page is non-resident, i.e., the page has been swapped to disk.

Several of these symbols may appear on the same page of the map. For example, the foregoing map tells you the DM process has several write-protected pages assigned to it in megabyte 0 of logical memory.

If DM has already been executed and you wish to examine the other type of memory, you can change the display by typing S to show swap file memory or M to show physical memory. If DM is displaying physical memory, the top line of the display appears like this:

```
Memory Display of Meg 1
```

If DM is displaying swap file memory, the top line of the display appears like this:

```
SwapMem Display of Meg 1
```

Using Prompts

None.

Notes on Usage

Use DM to monitor the system's use of memory, to monitor the linking of an image file, or as a tool in debugging software.

Related CIP Commands

pstat	List system processes

dmnt

Functional Description

Use this command to dismount any device mounted on your system.

Command Line Syntax

Mnemonic	dmnt		
Required parameter	Device List		
Switches			
Prompt control	:auto	:confirm	
Message control	:log		
Device selection	:class=	:exclude=	:uic=

Parameters

Device List	Function	Required. Use this parameter to specify the device(s) to be dismounted.
	Default	None.
	Syntax	Type a WMCS devicename. Wildcard symbols are allowed. (The DEVICES help file gives you the names of the devices on your system.)

Switches

:auto	Function	Use this switch to perform the desired operation without any confirmation.
	Default	:noauto
	Syntax	Type **:auto**
:confirm	Function	Use this switch to confirm or deny the desired operation on each device.
	Default	:noconfirm
	Syntax	Type **:confirm**
:class=	Function	Use this switch to specify the class(es) of device(s) you wish to include.
	Default	:class=*
	Syntax	Type **:class=** followed by a list of device classes separated by commas. Wildcard symbols are allowed.
:exclude=	Function	Use this switch to exclude devices from the Device List parameter.
	Default	All devices that match the Device List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of device designations separated by commas. Wildcard symbols are allowed.
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:uic=	Function	Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.
	Default	All devices that match the specified list are selected.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> dmnt _dc0
```

This command prompts for user confirmation before dismounting the device. A prompt like the following appears on your screen:

```
     _DC0
     Dismount  (Y or N)? >
```

The device _DC0 is dismounted when you type Y. _DC0 is unusable until it is remounted.

```
> dmnt _tt5 :auto
```

This command dismounts the device _TT5 without prompting for user confirmation.

```
> dmnt _mt0 :auto
```

This dismounts the tape so you can remove the tape from drive 0. The :auto switch dismounts the tape without prompting for confirmation.

```
> dmnt * :class=ttyspecial :auto
```

This automatically dismounts all TTY ports that were mounted with the :special switch.

Using Prompts

```
> dmnt
Device list   > _dc0
   _DC0
Dismount   (Y or N)? > y
```

This command performs the same function as the first example.

Notes on Usage

When you dismount a disk, all modified disk buffers are written to the disk, and all modified File Control Blocks (FCBs) are updated.

When you dismount a tape, it is rewound.

> NOTE: Always dismount a device before you remove it from the drive. If you execute DMNT, and a disk or tape other than the one mounted is in the drive, you may lose data.

Related CIP Commands

dev	Display a list of mounted devices
mnt	Mount a device

dstat

Functional Description

Use this command to set and display the device status.

Command Line Syntax

Mnemonic	dstat
Optional parameter	Device List

Switches

Selection	:class=	:exclude=	:uic=	:confirm
Display	:log	:pause		
Any device	:numtoretry=	:owner=	:protection=	:default
Disk drives	:rawverify	:writeprotect	:iopbcount=	
Queue-class devices	:assocdev=	:numactive=	:quetype=	:baseprior=
	:printtype=	:senddev=	:formname=	:quemgrres
	:width=	:length=		
Sync-class devices	:blockcheck=	:numsyncs=	:termtype=	:inbufsize=
	:numtrlpad=	:transtable=	:lineprotocol=	:outbufsize=
	:writetransl	:linetype=	:readtransl	
Tape drives	:errintenb	:speed=	:writeprotect	:rawverify
	:density=			
TTY devices	:autobaud	:controlz	:modemctrl	:baudrate=
	:datawidth=	:packetterm=	:broadcast	:duplex=
	:parity=	:controlc	:expandtabs	:remote
	:controlo	:hangup	:stopbits=	:controlu
	:hostsync=	:termtype=	:controlx	:mask8bit
	:xonxoff	:controldefault		

Parameters

Device List	Function	Optional. Use this parameter to specify the list of devices whose status will be displayed or set. (DSTAT displays the status if there are no set status switches specified.)
	Default	sys$output
	Syntax	Type the standard syntax for devicenames. Wildcard symbols are allowed.

Switches

:assocdev=	Function	Use this switch to specify the physical port that the print program will actually write to on a print type queue device.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:assocdev=** followed by any valid devicename. Wildcard symbols are not allowed.
:autobaud	Function	Use this switch on TTY class devices to determine the baud rate of a terminal on a port. You will have to press [RETRN] at most five times. This new baud rate will stay in effect on this port until no more processes are accessing it.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:autobaud** to enable, **:noautobaud** to disable.
:baseprior=	Function	Use this switch for queue class devices to specify the default priority that new entries will be entered into the queue.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:baseprior=** followed by an integer between 0 and 15.
:baudrate=	Function	Use this switch on TTY class devices to set the baud rate of the given device.

	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:baudrate=** followed by one of the following: **50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600,** or **19200.**
:blockcheck=	Function	Use this switch on sync class devices to specify the type of block checking to be performed on the packets received and transmitted by this device.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:blockcheck=** followed by one of the following: **CRC-16, CRC-CCITT,** or **LRC.**
:broadcast	Function	Use this switch to allow broadcasted messages to be sent to a TTY class device.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:broadcast** to enable, **:nobroadcast** to disable.
:class=	Function	Use this switch to specify the class(es) of device(s) you wish to include.
	Default	:class=*
	Syntax	Type **:class=** followed by a list of device classes separated by commas. Wildcard symbols are allowed.
:confirm	Function	Use this switch to confirm or deny the desired operation on each device.
	Default	:noconfirm
	Syntax	Type **:confirm**
:controlc	Function	Use this switch on TTY class devices to set the [CTRL] c mode.
	Default	:controlc
	Syntax	Type **:nocontrolc**
:controldefault	Function	Use this switch on TTY class devices to reset all control character states to their default states.

	Default	:nocontroldefault
	Syntax	Type :controldefault
:controlo	Function	Use this switch on TTY class devices to set the [CTRL] o mode.
	Default	:controlo
	Syntax	Type :nocontrolo
:controlu	Function	Use this switch on TTY class devices to set the [CTRL] u mode.
	Default	:controlu
	Syntax	Type :nocontrolu
:controlx	Function	Use this switch on TTY class devices to set the [CTRL] x mode.
	Default	:controlx
	Syntax	Type :nocontrolx
:controlz	Function	Use this switch on TTY class devices to set the [CTRL] z mode.
	Default	:controlz
	Syntax	Type :nocontrolz
:datawidth=	Function	Use this switch on TTY class devices to specify how many bits of data are to be sent with each character.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type :datawidth= followed by one of the following: 5, 6, 7, or 8.
:default	Function	Use this switch in conjuction with the :owner= or :protection= switches to set the default protection or UIC values.
	Default	:nodefault
	Syntax	Type :nodefault to enable
:density=	Function	Use this switch on TAPE class devices to specify the density of this device in bytes per inch.
	Default	:density=1600
	Syntax	Type :density= followed by one of the following: 800, 1600, 3200, 6250, 6400.

:duplex=	Function	Use this switch on TTY class devices to specify the type of duplex to be used on a given port.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:duplex=** followed by one of the following: **full** or **half.**
:errintenb	Function	Use this switch on TAPE class devices to enable error interrupts. If enabled, blocks with errors are not returned to users. If disabled, blocks with errors are returned to users.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:errintenb** to enable, **:noerrintenb** to disable.
:exclude=	Function	Use this switch to exclude devices from the Device List parameter.
	Default	All devices that match the Device List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of device designations separated by commas. Wildcard symbols are allowed.
:expandtabs	Function	Use this switch on TTY class devices to specify whether the system should convert tab characters to the correct number of spaces.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:expandtabs** to enable, **:noexpandtabs** to disable.
:formname=	Function	Use this switch on queue class devices to specify the type of forms on the printer device.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:formname=** followed by any valid string up to 16 characters long.
:hangup	Function	Use this switch on TTY class devices to issue a hangup request to the remote device before returning to the calling program (usually the CIP).

Default :nohangup
Syntax Type **:hangup**

:hostsync=

Function Use this switch on TTY class devices to specify the type of protocol to be used when the input buffer for the particular port is almost full.

Default Driver dependent unless changed by previous DSTAT command.

Syntax Type **:hostsync=** followed by one of the following:

bell When the buffer is almost full, a bell is sent for each character that is received.

software When the buffer is almost full, a [CTRL] s is sent for each character that is received. When the buffer is half full, a [CTRL] q is sent.

hardware When the buffer is almost full, the REQUEST TO SEND line is raised. When the buffer is half full, the REQUEST TO SEND line is lowered. For this to work, the receiver's REQUEST TO SEND line must be tied to the transmitter's CLEAR TO SEND line.

none When the buffer is almost full nothing happens.

:inbufsize=

Function Use this switch to specify the size of the input buffer for sync class devices in bytes (i.e., specify the maximum number of characters that can fit in the device's input buffer).

Default Driver dependent unless changed by previous DSTAT command.

Syntax Type **:inbufsize=** followed by a positive integer.

:iopbcount=

Function Use this switch to specify the number of input-output parameter blocks (essentially the maximum number of pending operation requests) on a disk.

Default Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:iopbcount=** followed by a positive integer.

:length=

Function — Use this switch on queue class devices to specify the number of lines on each printed page. A formfeed is automatically inserted after this many lines have been printed.

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:length=** followed by any positive integer. Type a zero if you do not want the PRINT routines to insert any formfeeds.

:lineprotocol=

Function — Use this switch on sync class devices to specify the type of line protocol to use.

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:lineprotocol=** followed by one of the following: **PntToPnt**, or **MultiPnt**.

:linetype=

Function — Use this switch on sync class devices to specify the character set being used for all protocol control characters.

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:linetype=** followed by ASCII or EBCDIC.

:log

Function — Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

Default — The value specified by the OPTION command.

Syntax — Type **:nolog** or **:log** to override the default.

:mask8bit

Function — Use this switch on TTY class devices to specify whether the high-order or eighth bit of each character is to be automatically set to zero as it is input.

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:mask8bit** to enable, **:nomask8bit** to disable.

:modemctrl	Function	Use this switch on TTY class devices to specify that if the connection to a remote computer is lost, the last process to access the device will be automatically terminated.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:modemctrl** to enable, **:nomodemctrl** to disable.
:numactive=	Function	Use this switch on queue class devices to specify how many entries in the queue may be active concurrently.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:numactive=** followed by any valid integer.
:numsyncs=	Function	Use this switch on sync class devices to specify the number of leading sync characters to be sent with each block transmitted.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:numsyncs=** followed by a valid integer greater than 2.
:numtoretry=	Function	Use this switch on all classes of devices to specify the number of times to retry an operation (such as read or write) on a device before giving up and reporting a hard error.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:numtoretry=** followed by a positive integer.
:numtrlpad=	Function	Use this switch on sync class devices to specify the number of trailing pad characters to be sent after each block transmitted.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:numtrlpad=** followed by any valid integer.
:outbufsize=	Function	Use this switch to specify the size of the output buffer for sync class devices in bytes (i.e., specify the maximum number of characters that can fit in the device's output buffer).

	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:outbufsize=** followed by a positive integer.
:owner=	Function	Use this switch to change the owner (UIC) of the device.
	Default	The UIC assigned when the device was mounted.
	Syntax	Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC that they represent.
:packetterm=	Function	Use this switch on TTY class devices to specify what control characters will force a packet termination condition.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:packetterm=** followed by a unique abbreviation of one of the following: **NoCntrlChr, AllCntrlChr, CrReturn.**
:parity=	Function	Use this switch on TTY class devices to specify the type of parity to be used on a given port.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:parity=** followed by one of the following: **disabled, odd,** or **even.**
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes after you press any character.
	Default	The value specified by the OPTION command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:printtype=	Function	Use this switch on queue class printer devices to specify the command line to use for default queueing, or to specify none if default queueing is not allowed.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:printtype=** followed by one of the following: a command-line character string, or **none.**

NONE means users cannot redirect the output to this particular queue device. If you specify a command line, it is passed on to the formatter program if you redirect output to the queue. Read the QPRINT command description in this manual for the command line format to be used for the default formatter.

:protection= Function Use this switch to modify the protection mask associated with the device.

Default Driver dependent unless changed by previous DSTAT command.

Syntax Type **:protection=** followed by a protection mask.

:quemgrres Function Use this switch on queue class devices to specify that the QUEMGR program is to remain in memory at all times and thus make accesses to the queue faster.

Default Driver dependent unless changed by previous DSTAT command.

Syntax Type **:quemgrres** or **:noquemgrres** to overide the default.

:quetype= Function Use this switch on queue class devices to specify the type of queue it is.

Default Driver dependent unless changed by previous DSTAT command.

Syntax Type **:quetype=** followed by one of the following: **print**, or **job**.

:rawverify Function Use this switch to do read-after-write verification of the data written to disk or tape class devices.

Default Driver dependent unless changed by previous DSTAT command.

Syntax Type **:rawverify** to enable, **:norawverify** to disable.

:readtransl Function Use this switch on sync class devices to specify whether data should be translated from EBCDIC to ASCII on reads.

Default Driver dependent unless changed by previous DSTAT command.

	Syntax	Type **:readtransl** to enable, **:noreadtransl** to disable.
:remote	Function	Use this switch on TTY class devices in conjunction with the :modemctrl mode to automatically terminate all processes which have the port open, plus all the child processes of any of these processes that have no parent if the connection to a remote computer is lost.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:remote** to enable, **:noremote** to disable.
:senddev=	Function	Use this switch on queue class devices to specify the device where messages will be sent when problems occur.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:senddev=** followed by any valid devicename.
:speed=	Function	Use this switch on tape class devices to specify the speed of a tape in inches per second.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:speed=** followed by one of the following: 12, 25, 30, 50, 90, 100, or 125.
:stopbits=	Function	Use this switch on TTY class devices to specify how many stop bits are to be sent with each data item on the given port.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:stopbits=** followed by one of the following: 1, 1.5, or 2.
:termtype=	Function	Use this switch on TTY and sync class devices to specify the type of terminal on this particular port.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:termtype=** followed by a valid terminal type.

There are 256 valid types that can be specified. The general form of specifying the types is the letters "FT" followed by a number between 0 and 255, e.g., ft0, ft20, ft200, etc. The following names correspond to certain FT types:

TTY device	Type
wit	ft247
hydra	ft248
vt100	ft250
vt52	ft251
t7000	ft252
mg8000	ft253
tvi912c	ft254
visual200	ft255

Sync Device	
IBM3780	ft255
IBM2780	ft254
IBM3275	ft253
IBM3276	ft252
IBM2770	ft251
IBM2968	ft250
IBM3741	ft249

:transtable=

Function — Use this switch on sync class devices to specify the translate table to use when translating between ASCII and EBCDIC.

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:transtable=** followed by one of the following: **3780**, or **3270.**

:uic=

Function — Use this switch to select only those devices that are part of the given list and are owned by the specified user or list of users.

Default — All devices that match the specified list are selected.

Syntax — Type **:uic=** followed by a list of UICs or usernames.

:width=

Function — Use this switch on queue class devices to specify the length of the lines on a printed page. If a line that is to be printed is longer than this value, it truncates the line and handles over-striking properly.

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:width=** followed by any positive integer. Type a zero if you do not want the PRINT routines to examine the line lengths.

:writeprotect

Function — Use this switch on disk and tape class devices to specify that the device cannot be written on. (This is a software check. If the hardware is write-protected, this flag is set.)

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:writeprotect** to enable, **:nowriteprotect** to disable.

:writetransl

Function — Use this switch on sync class devices to specify whether data should be translated from ASCII to EBCDIC on writes.

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:writetransl** to enable, **:nowritetransl** to disable.

:xonxoff

Function — Use this switch on TTY class devices to enable [CTRL] s and [CTRL] q functions so that no data are lost.

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:xonxoff** to enable, **:noxonxoff** to disable.

Examples

```
> dstat
```

This command displays the status of SYS$OUTPUT in this kind of report:

```
_TT12,4A1                                                 12-Jan-1985 10:53
Class        : TTY          Hard errors : 2              Driver ID    : $1001
Read oper    : 27835        Soft errors : 0              Block size   : 1
Write oper   : 829945       Num to retry: 0              Cur num dev  : 17
Owner        : [0004,0002] Username    : USER           Allocated    : No
Protection   : S:  RE,P:     ,G:DWRE,O:DWRE
Term type    : VISUAL200    In char cnt : 0              Duplex       : Full
Baud rate    : 19200        In buf size : 64             Parity       : Disabled
Data width   : 8 Bit        Out char cnt: 0              Current col  : 0
Stop bits    : 1            Out buf size: 128            Host Sync    : Bell
Packet term  : NoCntrlChr       ControlC       ControlO       ControlX
ControlU        ControlZ        NoAutobaud     Broadcast      ExpandTabs
Mask8Bit        NoModemCtrl     NoRemote       XonXoff
```

> dstat _ds0,_pq0

This displays the status of devices _DS0 and _PQ0 in this kind of report:

```
_DS0,0A0                                                  12-Jan-1985 10:52
Class        : Disk         Hard errors : 0              Driver ID    : $300B
Read oper    : 240370       Soft errors : 0              Block size   : 1024
Write oper   : 68575        Num to retry: 5              Cur num dev  : 1
Owner        : [0001,0001] Username    : SYSTEM         Allocated    : No
Protection   : S: WRE,P: WRE,G: WRE,O:DWRE
Inter factor: 3             Num cylndrs : 842            Disk density: Double
Num IOPB's   : 32           Num heads   : 20             Seek direct : Forward
Num sectors  : 421000       Sector/track: 25             Cache size  : 152
Drive type   : SMD474b                   NoRAWverify     NoWriteProtect

_PQ0,0A0                                                  12-Jan-1985 10:53
Class        : Queue        Hard errors : 0              Driver ID    : $7001
Read oper    : 0            Soft errors : 0              Block size   : 1
Write oper   : 0            Num to retry: 0              Cur num dev  : 2
Owner        : [0001,0001] Username    : SYSTEM         Allocated    : No
Protection   : S: WRE,P: WRE,G: WRE,O:DWRE
Assoc dev    : _TT19        Length      : 60             Num active   : 1
Send dev     : _tt0         Width       : 132            Cur active   : 0
Next entry   : 30           Que type    : Print          Base prior   : 7
Formname     : None         Print type  : NoNewline      QueMgrRes
```

> dstat * :class=tty* :xonxoff

This command enables the XONXOFF mode on all TTY and TTYSPECIAL class devices.

> dstat :termtype=mg8000

This command sets the user's terminal to an MG8000. Note that without a devicename it works on sys$output.

Using Prompts

None.

Notes on Usage

There is a default protection mask and UIC for all classes of devices. If a process changes the UIC and/or protection of a device, that UIC and/or protection mask remains in effect until no more processes have the device open. Then the UIC and protection of the device revert back to the specified default values.

If a switch is specified that is not appropriate for any of the devices, a warning is displayed.

It is legal to set modes on multiple classes of devices in the same command. It only uses those switches that apply to each particular class of device.

For terminal types that are not in the default WICAT-supported table (equivalent to FT247 to FT255), DSTAT just displays the number given. For example, if you set your terminal type to FT6, i.e., you type dstat :termtype=ft6, that is the string displayed by dstat. However, by using logical names, you can see the descriptive name for your terminal. Suppose you assigned the following logical names:

```
    > "@ft6" :=== termname
    > "@termname" :=== ft6
```

You could then use the special string TERMNAME in place of FT6 and the string is also correctly displayed by DSTAT.

Related CIP Commands

None.

dump

Functional Description

Use this command to display the contents of a file in hexadecimal format on the screen.

Command Line Syntax

Mnemonic	dump			
Required parameter	File List			
Switches				
Files to be dumped	:before= :mod	:uic=	:exclude=	:since=
Display of file	:address= :pages=	:bytes= :mask8bit	:pause :log	:header
Other	:rewind	:confirm		

Parameters

File List	Function	Required. Use this parameter to specify the file(s) to be dumped.
	Default	None.
	Syntax	Type a standard file list designation.

Switches

:address=	Function	Use this switch to specify the relative address or the address at which the byte count will begin.
	Default	:address=0 (or the first byte in the file)
	Syntax	Type **:address=** followed by a numeral. NOTE: This does not affect what part of the file will be displayed; it only changes the address in the display by the amount specified. For example, you would type :address=$100 to specify that the address numbering begins at hex 100.
:before=	Function	Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:before=** followed by a date or time in the standard date and time syntax.
:bytes=	Function	Use this switch to specify the bytes of the file that are to be displayed.
	Default	All bytes in the file are displayed.
	Syntax	Type **:bytes=** followed by a numeral range.
:confirm	Function	Use this switch to confirm or deny the desired operation on each file.
	Default	:noconfirm
	Syntax	Type **:confirm**
:exclude=	Function	Use this switch to exclude files from the File List parameter.
	Default	All files that match the file list parameter are selected.
	Syntax	Type **:exclude=** followed by a list of file designations separated by commas. Wildcard symbols are allowed.
:header	Function	Use this switch to suppress display of the full file specification as a header.

Default :noheader
Syntax Type **:header**

:log

Function Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
Default The value specified by the OPTION command.
Syntax Type **:nolog** or **:log** to override the default.

:mask8bit

Function Use this switch to mask the high order or eighth bit for the ASCII display of the file. The hexadecimal part of the display will not be affected.
Default :nomask8bit
Syntax Type **:mask8bit**

:mod

Function Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
Default :nomod
Syntax Type **:mod**

:pages=

Function Use this switch to specify the pages to be displayed.
Default All pages are displayed.
Syntax Type **:pages=** followed by a number range. (Page numbering begins at 1.)

:pause

Function Use this switch to stop the display after each screen of information. The display resumes when you press any key.
Default The value specified by the OPTION Command.
Syntax Type **:pause** or **:nopause** to override the default.

:rewind

Function Use this switch to rewind the tape device.
Default :norewind
Syntax Type **:rewind**

:since=

Function Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.

Default — All files that match the File List parameter are selected.

Syntax — Type :since= followed by a date or time in the standard syntax.

:uic=

Function — Use this switch to select only those files or devices that match the File List parameter and are owned by the specified user or list of users.

Default — All files that match the File List parameter are selected.

Syntax — Type :uic= followed by a list of UICs or usernames.

Examples

```
> dump cip.dat
```

This command displays the following kind of report on your terminal screen:

```
File __SYSTEM_DS0/USERS.GRACE/CIP.DAT.3
00000000  54 68 65 20  43 6F 6D 6D  61 6E 64 20  49 6E 74 65   The Command Inte
00000010  72 70 72 65  74 65 72 20  50 72 6F 67  72 61 6D 20   rpreter Program
00000020  28 43 49 50  29 20 70 72  6F 76 69 64  65 73 20 61   (CIP) provides a
00000030  6E 20 69 6E  74 65 72 66  61 63 65 0A  62 65 74 77   n interface.betw
00000040  65 65 6E 20  74 68 65 20  75 73 65 72  20 61 6E 64   een the user and
00000050  20 57 49 43  41 54 27 53  20 4D 75 6C  74 69 2D 75    WICAT'S Multi-u
00000060  73 65 72 20  43 6F 6E 74  72 6F 6C 20  53 79 73 74   ser Control Syst
00000070  65 6D 20 28  4D 43 53 29  0A                         em (WMCS).
```

This gives you a hexadecimal display of the contents of the most recent version of CIP.DAT.

The output is divided into pages of 256 bytes each. If the report is several pages long, the pages will scroll past on the screen. You can use the :pause switch or [CTRL] s to temporarily halt the scrolling, and [CTRL] q to cause it to resume.

DUMP is similar to TYPE in that DUMP allows you to look at the contents of a file, but does not allow you to emend the file. Unlike TYPE, DUMP allows you to display the contents of any file type. (TYPE only allows you to display data files.)

The first column of the display is the line number. The next 16 columns represent the hexadecimal value of the contents of the file. To the far right is the ASCII equivalent for the printable characters. If the characters are not printable, a dot, ., is used.

The header lists the file being displayed. The header is especially useful when you are using wildcard symbols to dump several files.

```
> dump cip.dat :noheader :bytes=0-22,40

00000000  54 68 65 20  43 6F 6D 6D  61 6E 64 20  49 6E 74 65   The Command Inte
00000010  72 70 72 65  74 65 72                                rpreter

00000020  xx xx xx xx  xx xx xx xx  6F                         ........o
```

The :bytes switch allows you to display selected portions from the file. The byte numbering begins at zero. If the specified byte is not evenly divisible by 16 (start of a hex line), then the front of the line is padded with x's, e.g. the line numbered 00000020 above.

Using Prompts

```
> dump
File list   > cip.dat
```

This performs the same function as the first example.

Related CIP Commands

type	Display the contents of a text file
zap	Inspect and change file contents

dumpdiff

Functional Description

Use this command to compare two files, side-by-side on the screen, in hexadecimal format.

Command Line Syntax

Mnemonic	dumpdiff	
Required Parameter	File 1	
Required Parameter	File 2	
Switches		
Format of the display	:header	:pause

Parameters

File 1	Function	Required. Use this parameter to specify the name of the first of two files to be compared.
	Default	None.
	Syntax	Type the standard syntax for a WMCS file designation. No wildcard characters are allowed.
File 2	Function	Required. Use this parameter to specify the name of the second of two files to be compared.

Default None.
Syntax Type the standard syntax for a WMCS file designation. No wildcard characters are allowed.

Switches

:header
Function Use this switch to suppress display of the full file specifications as a header.
Default :noheader
Syntax Type **:header**

:pause
Function Use this switch to stop the display after each screen of information. The display resumes when you press any key.
Default :pause
Syntax Type **:nopause**

Examples

```
> dumpdiff sample.txt example.txt
```

This command compares two files, SAMPLE.TXT and EXAMPLE.TXT. Each file contains a PSTAT display. This is what SAMPLE.TXT looks like:

```
Time:     06-Mar-1985 11:05:32.78                            Free memory: 2592K
Up time:            4 19:37:03.81
  PID        Process Name          Port            Status     Size Pri Scheduled Times
--------  ---------------  ---------------  ---------  ---- ---  -------- -----
00060001  Logflush         __MEPH_TT0       Waiting       8   2   2978330   100
00060066  Watchdog         __MEPH_TT0       Waiting      60   2      1993   100
00062DBB  NSP_answer       __MEPH_VC0       Waiting      12   2         1   100
00060002  Que_Manager      __MEPH_TT0       Waiting      44   2     50637   100
00060004  Swapper          __MEPH_TT12      Waiting      60   2       773   100
00062CE0  CIP_KENT         __MEPH_TT32      Waiting      20   4       971    60
00062A63  vew_LARSEN       __MEPH_TT17      Waiting     136   4      3319    60
00062A31  OIS_MARGENE      __MEPH_TT34      Waiting     124   4      4648    60
00062A3B  CIP_MARK         __MEPH_TT20      Waiting      24   4      2536    60
00062A75  CIP_NEAL         __MEPH_TT12      Waiting      20   4       827    60
00062A33  CIP_CRAIG        __MEPH_TT15      Waiting      24   4      1895    60
00062B8C  NSP_001400A7     __MEPH_VC2       Waiting      32   7        60    60
00062E2C  CIP_VEW          __MEPH_TT9       ChildWait    20   7        49    60
00062AB8  CIP_GB           __MEPH_TT9       ChildWait    24   7      1619    60
00062A49  CIP_LARSEN       __MEPH_TT17      ChildWait    20   7        59    60
00062A30  @WP_MARGENE      __MEPH_TT34      ChildWait    20   7        25    60
00062A28  CIP_MARGENE      __MEPH_TT34      ChildWait    64   7        12    60
```

This is what EXAMPLE.TXT looks like:

```
Time:     06-Mar-1985 11:05:37.24                            Free memory: 2528K
Up time:            4 19:37:08.27
  PID        Process Name          Port            Status     Size Pri Scheduled Times
--------  ---------------  ---------------  ---------  ---- ---  -------- -----
00060001  Logflush         __MEPH_TT0       Waiting       8   2   2978335   100
00060066  Watchdog         __MEPH_TT0       Waiting      60   2      1993   100
00062DBB  NSP_answer       __MEPH_VC0       Waiting      12   2         1   100
00060002  Que_Manager      __MEPH_TT0       Waiting      44   2     50637   100
00060004  Swapper          __MEPH_TT12      Waiting      60   2       773   100
00062CE0  CIP_KENT         __MEPH_TT32      Waiting      20   4       971    60
00062A63  vew_LARSEN       __MEPH_TT17      Waiting     136   4      3319    60
00062A31  OIS_MARGENE      __MEPH_TT34      Waiting     124   4      4648    60
00062A3B  CIP_MARK         __MEPH_TT20      Waiting      24   4      2536    60
00062A75  CIP_NEAL         __MEPH_TT12      Waiting      20   4       827    60
00062A33  CIP_CRAIG        __MEPH_TT15      Waiting      24   4      1895    60
00062B8C  NSP_001400A7     __MEPH_VC2       Waiting      32   7        60    60
00062E2C  CIP_VEW          __MEPH_TT9       ChildWait    20   7        68    60
00062AB8  CIP_GB           __MEPH_TT9       ChildWait    24   7      1619    60
00062A49  CIP_LARSEN       __MEPH_TT17      ChildWait    20   7        59    60
00062A30  @WP_MARGENE      __MEPH_TT34      ChildWait    20   7        25    60
00062A28  CIP_MARGENE      __MEPH_TT34      ChildWait    64   7        12    60
00062E06  vew_GB           __MEPH_TT9       ChildWait    64   7      1907    60
```

This is the display DUMPDIFF produces when it compares the two files:

```
Comparing _DS0/USERS.GRACE/SAMPLE.TXT.2 (#1)
     with _DS0/USERS.GRACE/EXAMPLE.TXT.2 (#2)
Address     File #1 Hexadecimal       Text      File #2 Hexadecimal       Text
--------  -----------------------   --------   -----------------------  --------
0000001C  32                        2          37                       7
0000001E  37 38                     78         32 34                    24
0000004C  39 32                     92         32 38                    28
0000006C  33                        3          38                       8
0000006E  38 31                     81         32 37                    27
00000159  30                        0          35                       5
00000518  34 39                     49         36 38                    68
00000661  xx xx xx xx xx xx xx xx   ........   30 30 30 36 32 45 30 36  00062E06
00000669  xx xx xx xx xx xx xx xx   ........   20 20 76 65 77 5F 47 42    vew_GB
00000671  xx xx xx xx xx xx xx xx   ........   20 20 20 20 20 20 20 20
00000679  xx xx xx xx xx xx xx xx   ........   20 20 20 5F 5F 4D 45 50     __MEP
00000681  xx xx xx xx xx xx xx xx   ........   48 5F 54 54 39 20 20 20  H_TT9
00000689  xx xx xx xx xx xx xx xx   ........   20 20 20 20 43 68 69 6C      Chil
00000691  xx xx xx xx xx xx xx xx   ........   64 57 61 69 74 20 20 20  dWait
00000699  xx xx xx xx xx xx xx xx   ........   20 36 34 20 20 20 37 20   64   7
000006A1  xx xx xx xx xx xx xx xx   ........   20 20 20 20 20 31 39 30       190
000006A9  xx xx xx xx xx xx xx xx   ........   37 20 20 20 20 36 30 0A  7    60.
```

Note the x character filler in the hexadecimal dump when one file is shorter than the other.

```
> dumpdiff sample.txt sample.txt
```

This command generates the following kind of display:

```
Comparing _DS0/USERS.GRACE/SAMPLE.TXT.2 (#1)
     with _DS0/USERS.GRACE/SAMPLE.TXT.2 (#2)
The files are the same.
```

Using Prompts

```
> dumpdiff
File 1          > sample.txt
File 2          > example.txt
```

This command produces the same result as the first example.

Related CIP Commands

dump	Dump file contents

err

Functional Description

Use this command to display the WMCS diagnostic message assigned to a WMCS status number.

Command Line Syntax

Mnemonic	err
Required parameter	Diagnostic No.
Switches	:pause

Parameters

Diagnostic No.	Function	Required. Use this parameter to specify the ranges of status numbers for which you want messages.
	Default	None.
	Syntax	Type a list of decimal or hexadecimal numerals separated by commas. Ranges of numerals are allowed. For instance, you can type 3-7 to represent the numbers 3, 4, 5, 6, and 7.

Switches

:pause Function Use this switch to stop the display after each screen of information. The display resumes when you press any key.

Default The value specified by the OPTION Command.

Syntax Type **:pause** or **:nopause** to override the default.

Examples

```
> err 147
```

This command generates the following report:

```
     147  The specified filename is syntactically incorrect.
```

This is the message assigned to diagnostic number 147. The messages are found in the file named sys$disk/syslib/error.txt.

```
> err 147,133,3-7
```

This command generates the following report:

```
  3  The process's buffer does not begin on a word boundary.
  4  The logical address, for the memory requested, is invalid.
  5  The process requested a logical page that was already allocated.
  6  The process tried to affect a page in memory it did not own.
  7  All available memory has been allocated.
133  The specified file could not be found.
147  The specified filename is syntactically incorrect.
```

Note that the messages appear in ascending order, regardless of the order in which they are typed on the CIP command line.

```
> err 133,$b2,-176
```

This command generates the following report:

```
-176  The process buffer is too small for the specified operation.
 133  The specified file could not be found.
 178  The FCB.SEQ number for the file does not match the specified FCB.
```

Note that hexadecimal numbers are preceded by a dollar sign, $. Note also that the decimal equivalent of the diagnostic number $b2 is shown in the report (178).

Using Prompts

```
> err
Diagnostic no. > 133,$b2,-176
```

This performs the same function as the previous example.

Notes on Usage

WMCS diagnostic messages, or status numbers, range from zero to 4095. (See appendix A in this manual for details.) The hexadecimal range is $01 through $FFF. Messages are now assigned only to status numbers 0 through 459. Numbers 460 through 4095 are reserved for use as the system is enhanced. This report appears on the screen when you specify a status number greater than 459:

```
499  No error message available.
```

If you specify a status number from 0 to 459 but no message has been assigned to the specified number, this report appears:

```
100  No error assigned.
```

If you specify a value for the parameter that is not a number, the following kind of report appears:

```
ERR      : Parsing B3F9
FAILED   : Status = 11.
MESSAGE  : No number was found during a search or scan for a number.
```

Related CIP Commands

None.

fpmgr

Functional Description

Use this command to display, install, or remove the system's floating-point packages.

Command Line Syntax

Mnemonic fpmgr

Optional parameters
Package List
Function (:list, :add, :delete, :clear)

Parameters

Package List

Function Optional. Use this parameter to specify types of floating-point hardware and software.

Default None.

Syntax Type one or more of the following (separated by commas): **LIB2**, **SKY1**, or **FFP1**.

Function

Function Optional. Use this parameter to specify the action to be performed on floating-point packages.

Default :list.

Syntax Type one of the following actions:

:list to list the installed floating-point packages in the order they were installed.

:add to install floating-point packages in the order they appear in the package list.

:delete to remove the specified packages from the system.

:clear to remove all floating-point packages from the system.

Switches

None.

Examples

```
> fpmgr :list

     LIB2
     SKY1
```

This command lists the installed floating-point packages in the order they were installed. If no packages are installed, the message "No floating-point found" is displayed.

```
> fpmgr lib2,sky1 :add
```

This command installs the floating-point software emulation package (lib2) and the package which uses the Sky hardware (sky1). In doing so, it initializes the packages and the Sky board, including downloading the Sky microcode.

```
> fpmgr sky1 :del
```

This command removes the package which utilizes the Sky board. Programs which are subsequently executed will not be able to use the Sky board to do their floating-point.

Using Prompts

None.

Notes on Usage

LIB2 is the floating-point emulation software package, SKY1 uses the Sky Computers, Inc. board, and FFP1 uses WICAT's own fast floating point board.

Programs which were not compiled to run on a specific hardware will use the first package installed.

To install a hardware package, the corresponding hardware board must be present in the system.

You must have system privilege to add or delete packages; however, anyone can list the packages.

When a package is installed by FPMGR, an initialization program specific to that package is run. These programs initialize the package and any associated hardware (this includes downloading microcode). The programs are: lib2init.exe for LIB2, sky1init.exe for SKY1, and ffp1init.exe for FFP1. They reside in /SYSEXE/. The microcode file for SKY1 resides in / SYSDSR/SKYUCODE.NNN, where NNN is 100 or 156.

Related CIP Commands

None.

fstat

Functional Description

Use this command to inspect and change file attributes.

Command Line Syntax

Mnemonic		fstat			
Optional parameter		File List			
Switches					
	File selection	:since= :uic=	:exclude= :confirm	:before=	:mod
	Set file attributes	:protection= :fileid=	:owner= :recordlen=	:modified= :filetype=	:created=
	Format of display	:pause	:bytes	:log	
	Other	:rewind			

Parameters

File List	Function	Optional. Use this parameter to specify the files you want to inspect or change.
	Default	All files in the default directory are displayed or altered.
	Syntax	Type the standard syntax for file designations.

Switches

:before=	Function	Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:before=** followed by a date or time in the standard date and time syntax.
:bytes	Function	Use this switch to display file sizes in bytes.
	Default	:nobytes
	Syntax	Type **:bytes**
:confirm	Function	Use this switch to confirm or deny the desired operation on each file.
	Default	:noconfirm
	Syntax	Type **:confirm**
:created=	Function	Use this switch to set the creation date of a file.
	Default	Does not change the creation date from its current value.
	Syntax	Type **:created=** followed by the standard syntax for dates and time.
:exclude=	Function	Use this switch to exclude files from the File List parameter.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of file designations separated by commas. Wildcard symbols are allowed.
:fileid=	Function	Use this switch to set the file ID on a file.
	Default	Does not change the file ID from its current value.
	Syntax	Type **:fileid=** followed by a number.
:filetype=	Function	Use this switch to set the file type of a file.
	Default	Does not change the file type from its current value.

	Syntax	Type **:filetype=** followed by one of the following: **Data, Directory, Image, KsamData, KsamKey, System, Archive, ArchiveCont, LLImage, Encrypted** or a number greater than or equal to 256 (user-defined type).
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**
:modified=	Function	Use this switch to set the modification date of a file.
	Default	Does not change the modification date from its current value.
	Syntax	Type **:modified=** followed by the standard syntax for dates and time.
:owner=	Function	Use this switch to change the owner (UIC) of the file.
	Default	The UIC assigned when the file was created or last updated.
	Syntax	Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC that they represent.
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when you press any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:protection=	Function	Use this switch to modify the protection mask associated with the file.

Default — The protection assigned when the file was created or last updated.

Syntax — Type **:protection=** followed by a protection mask.

:recordlen=

Function — Use this switch to set the record length on a file.

Default — Does not change the record length from its current value.

Syntax — Type **:recordlen=** followed by a positive integer.

:rewind

Function — Use this switch to rewind the tape device.

Default — :norewind, i.e. the tape device begins at the current position instead of at the beginning of the tape.

Syntax — Type **:rewind.**

:since=

Function — Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.

Default — All files that match the File List parameter are selected.

Syntax — Type **:since=** followed by a date or time in the standard syntax.

:uic=

Function — Use this switch to select only those files that are part of the given list and are owned by the specified user or list of users.

Default — All files that match the specified list are selected.

Syntax — Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> fstat
```

This command generates the following kind of display:

```
_DS0/USERS.WORKBENCH.PROJECTS.180101201.D/AAA.TXT.1
Logical :         0.1  FCB.SEQ :    7393.7465  Created : 30-Dec-1983 09:15:58
Physical:         1.0  Parent  :  12504.12506  Modified: 30-Dec-1983 09:15:58
Extents :           1  Filetype:     Data      Owner   : [0001,0002]
Record  :           1  Fileid  :       0       Username: SEAN
Protection: S:  RE,P:     ,G:DWRE,O:DWRE

_DS0/USERS.WORKBENCH.PROJECTS.180101201.D/ALLOC.TXT.3
Logical :         6.9  FCB.SEQ :  15518.15525  Created : 04-Jan-1985 13:48:00
Physical:         7.0  Parent  :  12504.12506  Modified: 04-Jan-1985 13:48:00
Extents :           3  Filetype:     Data      Owner   : [0012,0002]
Record  :           1  Fileid  :       0       Username: GRACE
Protection: S:  RE,P:     ,G:DWRE,O:DWRE

_DS0/USERS.WORKBENCH.PROJECTS.180101201.D/ARCH.TXT.23
Logical :        27.1  FCB.SEQ :    5246.5346  Created : 06-Jan-1985 12:15:03
Physical:        28.0  Parent  :  12504.12506  Modified: 06-Jan-1985 12:15:03
Extents :           5  Filetype:     Data      Owner   : [0007,0002]
Record  :           1  Fileid  :       0       Username: DEBRA
Protection: S:  RE,P:     ,G:DWRE,O:DWRE
```

What each field means:

Logical — logical file size expressed to the nearest 1/10th Kbyte.

Physical — physical file size expressed to the nearest 1/10th Kbyte.

Extents — number of extents in the FCB.

Record — record length, i.e. the number of bytes per record.

FCB.SEQ — FCB and sequence number.

Parent — FCB and sequence number of the parent FCB.

Filetype — what kind of file this is.

Possible file types are:

Data	data file
Directory	directory file
Image	executable image file
KsamData	KSAM data file
KsamKey	KSAM key file
System	system file
Archive	archive file
ArchiveCont	archive continuation file
LLImage	executable image file created by LL linker

Encrypted   encrypted file
A number   user-defined file types (256-4095)

Fileid   (currently not in use).

Created   date/time the file was created.

Modified   date/time the file was last modified.

Owner   UIC of the owner of the file.

Username   username of the owner of the file.

Protection   protection mask of the file.

> fstat arch.txt :bytes

This command displays all the attributes of the file ARCH.TXT (as in the preceding example), but expresses the logical and physical file sizes in bytes instead of Kbytes.

```
_DS0/USERS.WORKBENCH.PROJECTS.180101201.D/ARCH.TXT.23
Logical :      27704  FCB.SEQ :    5246.5346   Created : 06-Jan-1985 12:15:03
Physical:      28672  Parent  :  12504.12506   Modified: 06-Jan-1985 12:15:03
Extents :          5  Filetype:      Data      Owner   : [0007,0002]
Record  :          1  Fileid  :        0       Username: DEBRA
Protection: S:  RE,P:     ,G:DWRE,O:DWRE
```

> fstat arch.txt :filetype=data :owner=system

This command sets the filetype of ARCH.TXT to DATA and the owner to SYSTEM. Then this message appears:

```
_DS0/USERS.WORKBENCH.PROJECTS.180101201.D/ARCH.TXT.23 Altered.
```

Using Prompts

None.

Notes on Usage

Note that attributes of tape files cannot be set, they can only be displayed. Attributes of disk files can be set or displayed.

FSTAT produces the same display for files on a tape and files on a disk.

Related CIP Commands

dir	List the contents of a directory

gprint

Functional Description

Use this command to write screen graphics to a printer.

Command Line Syntax

Mnemonic gprint

Switches

Device selection	:input=	:output=	:savefile=	
Printer selection	:axiom	:prism		
Print control	:binaryupload	:bottom=	:color	:confirm
	:darkblack	:left=	:mapcolor=	:print
	:right=	:rotate	:scale=	:showmap
	:top=			

Parameters

None.

Switches

:input=

Function Use this switch to specify the devicename of the input MG8000 or WIT, or the filename of the input savefile. If a devicename is specified it must be the device name of an input MG8000 or WIT terminal. If a filename is specified, it must be the name of a file that was previously created by GPRINT using the :savefile= switch.

Default :input=sys$input

Syntax Type **:input=** followed by a devicename or a filename in the standard WMCS syntax.

:output=

Function Use this switch to specify the device or filename that is to receive the output that is formatted for the printer. If the output device is a print queue, the print data are written to a temporary queue file which is queued up for printing after GPRINT terminates. If this is the case, the queue's ":printtype=" switch must be properly initialized as described in the Notes on Usage section below. If the output device is a file, then the print data are written to the file and GPRINT terminates. If the output is the name of a port that is directly hooked up to an Axiom or Prism printer, GPRINT writes the print data directly to the printer and does not terminate until the picture is entirely printed.

Default :output=sys$print

Syntax Type **:output=** followed by a devicename or a filename in the standard WMCS syntax.

:savefile=

Function Use this switch to make a savefile which contains a ribbon-color bitmap of the image to be printed. This savefile can be used as input to a later GPRINT which actually formats the output data and prints it.

Default None.

Syntax Type **:savefile=** followed by a valid WMCS filename.

:axiom

Function Use this switch to specify that the destination printer is an Axiom model IMP2QAA1 graphics printer.

NOTE: If this switch is specified the :prism switch is not allowed. Also, if this switch is specified GPRINT does not allow color printing.

Default :prism

Syntax Type **:axiom**

:prism Function Use this switch to specify that the destination printer is an Integral Data Systems (IDS) model Prism 132 graphics printer. If this printer is selected and the input is a WIT terminial, GPRINT assumes that the printer is a color printer.

Default :prism

Syntax Type **:prism**

:binaryupload Function Use this switch--on a WIT terminal only--to upload the data using the WIT terminal's binary "rastor upload" command. Using this method to print a full screen requires uploading 153600 bytes of data from the terminal. This takes approximately 2 1/2 minutes if the terminal is running at 9600 baud. The alternative to this switch is the normal WIT run-length-encoded upload format which is usually much shorter than the binary version. Occasionally some complicated pictures can be printed faster using the :binaryupload switch.

Default Run-length-encoded uploading is done.

Syntax Type **:binaryupload**

:bottom= Function Use this switch to specify the bottom margin of the picture. This is the line number of the bottom-most line of the terminal screen to be uploaded and printed. The lines on the terminal start at 0 on the bottom of the screen, and end at 299 on an MG8000 terminal. The lines end at 479 on a WIT terminal.

Default :bottom=0

Syntax On an MG8000, type **:bottom=** followed by a number between 0 and 299. On a WIT, type **:bottom=** followed by a number between 0 and 479.

:color

Function Use this switch to specify whether or not a COLOR Prism printer is available. This switch is ignored if the :axiom switch is specified. If :nocolor is specified, the screen is printed in a black-and-white format.

Default :color (if the printer is a Prism printer and the input device is a WIT terminal). Otherwise the default is :nocolor and cannot be changed.

Syntax Type **:color** or **:nocolor** to override the default.

:confirm

Function Use this switch with a WIT terminal to cause GPRINT to pause after mapping print colors, but before uploading and printing the screen. GPRINT prompts the user for confirmation, and allows only Y or N (upper- or lowercase) as a response. This is a very useful switch when working with WIT terminals. Just before pausing, GPRINT shows the user how the picture will look on the printer with the specified color mapping. If the user does not like the mapping as is, he can respond with N and restart GPRINT with a different :mapcolor switch. This means the user does not have to go through the upload screen and print phases in order to see how the colors will map.

Default :noconfirm

Syntax Type **:confirm**

:darkblack

Function Use this switch with a WIT terminal to cause GPRINT to make multiple ribbon passes on black colors in order to make the black appear to be a real black (instead of grey). To keep the number of ribbon passes to a maximum of four per line, :darkblack combines all four ribbons —with one pass of each. If :nodarkblack is used, only one pass of the black ribbon is made.

Default :darkblack

Syntax Type **:nodarkblack**

:left=

Function Use this switch to specify the left margin of the picture. This is a column number of the left-most column of the terminal screen to be uploaded and printed. The columns on the terminal start at 0 on the left edge of the screen and end at 399 on the right edge of the

Notes on Usage

A Network Surrogate Process (NSP) is responsible for mounting and tending to the virtual circuits. NSP automatically mounts virtual circuit devices as it needs them, based on a user-specified allowable range of device numbers.

NSP must be executing in order for networking to operate. It performs the requested operations locally in behalf of a process on a remote system. The NSP process must be forked when the system is booted. After that, the process clones itself so there is always an NSP available for new remote processes to use.

The NSP image file SYS$DISK/SYSEXE/NSP.EXE must be installed at boot time with the chngsuper privilege.

The operating system and the NSP work very closely together ir order to make networking happen. When a local process attempts to do something on a remote system over the network, the operating system "calls" the remote system through an available virtual circuit. The NSP process on the remote system "answers" the call. It then performs supervisor calls on behalf of the local process for as long as the process exists or the virtual circuit is open. As soon as the local process dies or a "hangup" is performed on the virtual circuit by the process, the NSP on the remote system dies.

Related CIP Commands

nstat	Display network status
nsysprof	Manage the system network profile

screen for an MG8000 terminal. The columns end at 639 on the right edge of the screen for a WIT terminal.

Default :left=0

Syntax Type **:left=** followed by a number from 0 to 399 (or from 0 to 639 for a WIT terminal).

:mapcolor= Function Use this switch on a WIT terminal to manually map specific color numbers on the WIT to be printed as specific printer colors on the printer. When GPRINT is first started up on a WIT, it reads the current values of the 16 color registers and assigns a default color mapping for each of the 16 colors. This switch overrides the default mapping. Any colors mapped using this switch are mapped with the user-specified values. All other non-user mapped colors are mapped to the default colors.

Default Default color mapping is done.

Syntax Type **:mapcolor=** followed by a list of **color=registerlist** commands separated by commas.

The **color** can be any of the following colors typed in upper- or lowercase: WHITE, BLACK, YELLOW, MAGENTA or RED, CYAN or BLUE, ORANGE, VIOLET or PURPLE, GREEN, BROWN, and GREY.

NOTE: MAGENTA and RED are synonyms, as are CYAN and BLUE. VIOLET and PURPLE are also synonyms.

Color names do not need to be typed out in their entirety. Like other WMCS switches, any unique abbreviation of the color names is allowed.

The **registerlist** which comes after the equal sign in the syntax is a list of color register numbers from 0 to 15. This list of registers uses standard WMCS range wildcarding.

Suppose you typed this switch:

```
     :mapcolor=Green=1-3,11,Red=15,ora=7
```

WIT colors 1,2,3 and 11 will print as green, WIT color 15 will print as red, and WIT color 7 will print as orange.

:print

Function: Use this switch to indicate to GPRINT whether or not it should go through the print phase. This switch is typically used with the :savefile= switch to cause GPRINT to create the savefile and quit without printing the picture.

Default: :print

Syntax: Type :noprint

:right=

Function: Use this switch to specify the right margin of the picture. This is a column number of the right-most column of the terminal screen to be uploaded and printed. The columns on the terminal start at 0 on the left edge of the screen, and end at 399 on an MG8000 terminal. The columns end at 639 at the right edge of the screen for a WIT terminal.

Default: :right=399 (for a WIT, :right=639)

Syntax: Type **:right=** followed by a number from 0 to 399 (or from 0 to 639 for a WIT terminal).

:rotate

Function: Use this switch to rotate the picture 90 degrees before printing it on the printer. A non-rotated picture is printed with the top line of the terminal screen horizontal as the top line of the printed output. A rotated picture is printed with the top line of the terminal screen vertical as the left column of the printed output. This switch is mainly useful if a scale is used that is greater than 1, because larger pictures typically will not fit horizontally on the paper.

Default: :norotate

Syntax: Type **:rotate**

:scale=

Function: Use this switch to scale the picture to a larger size. For instance, a scale of 2 prints the picture twice as wide and twice as tall (actually four times as big). A scale of 3 prints the picture three times as wide and three times as tall, etc. If the scaled picture cannot fit on the printer page, GPRINT truncates it and prints what will fit.

Default: :scale=1

Syntax: Type **:scale=** followed by any number.

:showmap Function Use this switch to display on the user's terminal screen the current values for the WIT-to-printer color mapping. This map is printed on the terminal before the optional :confirm prompt, so the user can look at the screen and look at the printed map, and then decide whether or not to print the picture. Default :noshowmap

Syntax Type **:showmap**

:top= Function Use this switch to specify the top margin of the picture. This is a line number of the topmost line of the terminal screen to be uploaded and printed. The lines on the terminal start at 0 on the bottom of the screen, and end at 299 on an MG8000 terminal. On a WIT terminal, the lines end at 479 at the top of the screen.

Default :top=299 (for a WIT :top=479)

Syntax Type **:top=** followed by a number from 0 to 299 (or from 0 to 479 for a WIT).

Examples

```
> gprint
```

This command uploads the whole graphics screen of the user's current terminal and formats it to be printed on a Prism printer. If the current terminal is a WIT, then :color is assumed, and default color mapping is used. The formatted output is written to the device or file identified by the logical name sys$input.

```
> gprint :input=_tt17 :output=_pq1 :prism :confirm :savefile=test.sav
      :showmap :mapcolor=green=5,yellow=8,10-12 :bottom=240 :right=100
```

If _tt17 is a WIT terminal, this command automatically maps the colors on the WIT screen to the closest available printer color, and then overrides that automapping by making color 5 be green and colors 8, 10, 11 and 12 be yellow. The actual colors on the WIT display are changed so that the user can see on the screen how the mapping looks. Then the color map is displayed on the users terminal in a text format, followed by a prompt which asks for a confirmation of whether the user really wants to print the screen now that he has seen how it will map on the printer. If the user responds with a N, GPRINT maps the screen back to the original colors and exits. If the user responds with a Y, GPRINT uploads the portion of _tt17's screen with a left margin of 0, a right margin of 100, a bottom margin of 240 and a top margin of 479. GPRINT then writes a ribbon-color-bitmap of this portion of the screen to a file called test.sav in the current directory. Finally, GPRINT formats the output for an IDS Prism printer and writes it to _pq1. Then it maps the colors back to the original colors and exits.

If _tt17 is an MG8000, this command does basically the same thing, except that it maps all of the non-black colors to be WHITE, and the top margin would be 299.

```
> gprint :input=test.sav :output=_pq2
```

This command reads the ribbon-color-bitmap from the file test.sav and formats it for a Prism printer and output it to _pq2.

Using Prompts

None.

Notes on Usage

This utility only supports WICAT graphics terminals, which are the MG8000 and the WIT terminals.

Also, you must use one of the following kinds of printers (or a printer that corresponds to one of these) in order to use GPRINT:

1. Integral Data Systems (IDS) model Prism 132 graphics printer. This printer comes with a color option which is necessary if the user desires to print color pictures from a WIT terminal.

2. The Axiom printer, model IMP2QAA1, graphics printer. This printer is black and white only.

The MG8000 terminal has a separate graphics and text memory. GPRINT only prints the graphics memory. This means that the user can use the terminal to start up GPRINT, and the GPRINT commands that show up in the text memory on the terminal will not interfere with the graphics printing. Only the graphics memory is printed.

However, the WIT terminal only has a graphics memory, and all non-graphics text is mixed in with that memory. Therefore, GPRINT prints everything that shows up on the screen. Thus, the user will probably want to run the GPRINT utility on a terminal that is not the same as the one which contains the picture to be printed. This can be done by simply redirecting input as follows:

```
> gprint <_tt17
```

Or, use the :input= switch as follows:

```
> gprint :input=_tt17
```

Either way works the same. If both are specified the :input switch takes precedence.

If the savefile= switch is used, GPRINT writes an intermediate file containing a ribbon-color bit map of the screen that was uploaded. This bit-map is saved after doing the automatic and manual color mapping, it does not save any information about the original WIT colors. This file is in the following format: (Note all fields are stored as binary values)

Addr 0 for 1 word:	rows_in_map
Addr 2 for 1 word:	bytes_per_row
Addr 4 for 1 word:	bytes_per_map
Addr 6 for 1 longword:	yellow_offset
Addr 10 for 1 longword:	magenta_offset
Addr 14 for 1 longword:	cyan_offset
Addr 18 for 1 longword:	black_offset
Addr 22	start of first bitmap

The rows_in_map field is the first 16-bit word of the file. It indicates how many rows of bytes there are in each bitmap. For example, if the specified bottom and top margins were 100 and 150 then the number of rows is 51, and this is what would be saved in this first word.

The bytes_per_row field indicates how many 8-bit bytes make up each row. For example, if the specified left and right margins were 100 and 150 then there would be 51 bit-columns per row, which would be stored as the first 51 bits of 7 8-bit bytes. The last 5 bits of the last byte would be zero. The bits are arranged such that the high order bit of the first byte of each row is the left most bit that was uploaded from that row on the terminal. The low order bit of the last byte of each row is the right most bit of the row.

The bytes_per_map is simply the product of the rows_in_map value multiplied by the bytes_per_row value. This is the total number of bytes that are contained in any given ribbon-color bit map if the map exists. If all four maps exist, then the total number of bytes in the savefile would be bytes_per_map * 4 plus the size of the header for the savefile which is 22 bytes.

The yellow_offset value is a zero-relative offset into this same map file where the yellow ribbon-color bitmap resides. For example, if this field is 22 then the yellow map starts at the 22nd byte of the savefile. If this field is zero then there is NOT a yellow bitmap in this file. For WIT terminals with the color switch enabled this bitmap is always present in the savefile. Otherwise, for black and white prints this bitmap is never present.

The magenta_offset (red) value is a zero-relative offset into this same map file where the magenta ribbon-color bitmap resides. For example, if this field is 22 then the magenta map starts at the 22nd byte of the savefile. If this field is zero then there is NOT a magenta bitmap in this file. For WIT terminals with the color switch enabled this bitmap is always present in the savefile. Otherwise, for black and white prints this bitmap is never present.

The cyan_offset (blue) value is a zero-relative offset into this same map file where the cyan ribbon-color bitmap resides. For example, if this field is 22 then the cyan map starts at the 22nd byte of the savefile. If this field is zero then there is NOT a blue bitmap in this file. For WIT terminals with the :color switch enabled this bitmap is always present in the savefile. Otherwise, for black and white prints this bitmap is never present.

The black_offset value is a zero-relative offset into this same map file where the black ribbon-color bitmap resides. For example, if this field is 22 then the black map starts at the 22nd byte of the savefile. There is always a black bitmap in every savefile.

If GPRINT output is written to a printer queue, the printer queue needs to be initialized properly or the picture will not print properly. The queues are generally initialized in the command file sys$disk/syslib/deviceup.com. The system manager should edit this file and make sure the print queue is initialized in a manner SIMILAR to the following. For this example assume that _tt5 is connected to a Prism printer, and _tt1 is connected to an Axiom printer.

```
PRISM PRINTER INITIALIZATION
mnt _pq0
dstat _pq0 :assocdev=_tt5 :quemgrres :width=80 :owner=[1,1]\
   :default :printtype="qprint __bob_pq0 :raw :nohead :noprintnull\
   :killedout=\"00 03 02 1B (Q4$) 0D\""

AXIOM PRINTER INITIALIZATION
mnt _pq1
dstat _pq1 :assocdev=_tt1 :quemgrres :width=80 :owner=[1,1]\
   :default :printtype="qprint __bob_pq1 :raw :nohead :noprintnull\
   :killedout=\"%508[00] 0A 1B 34 0D\""
```

The important switches in the above examples are the :raw switch inside the :printtype= switch and the :killedout switch. The above initialization of the queues has nothing to do with the switch settings used by the PRINT utility. This initializion is only used by WMCS if a print queue is opened directly (as in GPRINT) and written to and then closed. The :raw switch indicates to QPRINT that it should allow binary escape sequences, etc., to be written directly to the printer. The :killedout= switch specifies a string of data that is written to the printer if QPRINT does not run to normal completion, i.e., if it is killed prematurely. If the printer were in graphics mode when a print job is killed, then it would still be in graphics mode when the next print job starts up, and it will not have done a formfeed and a carriage return. The :killedout= switch allows the system manager to specify a string of bytes that will reset the printer. In the above example for the Prism printer the sequence "00 03 02 1B (Q4$) 0D\" writes a hex null incase the printer was processing an internal graphics sequence. Then it writes a hex "03 02" which takes the printer out of graphics mode. Then it writes "<ESC> Q 4 $" which selects the black ribbon, and finally it writes a carriage return (hex 0d).

In the above example for the Axiom printer the sequence "%508[00] 0A 1B 34 0D\" writes 508 null bytes to the printer to flush out its graphics mode buffers. (When the Axiom printer is put into graphics mode it is told that the next n characters are to be printed in graphics mode. We must write 508 nulls because that is the maximum number of graphics characters that the printer may be waiting for. If it was only needing 10 more nulls to exit graphics mode the other 498 will not print and will have no effect on the output etc...) Next it writes a linefeed to advance the paper and an <ESC> 4 sequence to put the printer back into 6 lines per inch. Finally it writes a carriage return (hex 0d). For a complete description of the syntax of this switch see the documentation for the QPRINT utility.

Related CIP Commands

None.

incl

Functional Description

Use this command to include or merge text files.

Command Line Syntax

Mnemonic	incl	
Required parameters	Source Destination	
Switches		
Content of output	:log	:filelines

Parameters

Source	Function	Required. Use this parameter to specify the file containing the text and include strings.
	Default	None.
	Syntax	Type the standard syntax for file designations. No lists or wildcard symbols are allowed.
Destination	Function	Required. Use this parameter to name the file that will contain the merged text.
	Default	None.
	Syntax	Type the standard syntax for file designations. No lists or wildcard symbols are allowed.

Switches

:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:filelines	Function	Use this switch to generate a text record which lists the line number and filename.
	Default	:nofilelines
	Syntax	Type **:filelines**

Examples

INCL involves more than typing the mnemonic and parameters onto the CIP command line. Follow this procedure to execute this command:

Step 1 — Use the VEW Command to access the file in which you wish to include the text of another file.

Step 2 — Place the cursor on the left margin of the screen at the point in the text where you want the other file inserted.

Step 3 — Type two percent signs, %%, and the file designation for the file you want to insert at this point. This is known as the "include string." (The file you specify here need not be in your default directory. However, if it is not in your default directory, you must type the complete file designation.)

> NOTE: Neither a list of files nor wildcard symbols can be used in this file designation. Furthermore, the include string must be the first and only thing on the line (leading spaces are not allowed).

Step 4 — Type as many include strings, of the kind you typed in step 3, as you need.

Step 5 — Strike [ESC][ESC]

Step 6 — Type **ex** onto the VEW command line in order to exit the file. In a moment, the cursor reappears on the CIP command line.

Step 7 Type **incl** onto the CIP command line.

Step 8 Press the spacebar.

Step 9 Type the file designation for the file you just exited.

Step 10 Press the spacebar.

Step 11 Type the name of the destination file. This file will contain the text of the file specified in the first parameter, as well as the text called for by the include strings inserted in that file.

For example, suppose this is the contents of the file MARY.TXT:

```
1 Mary had a little lamb
3 And everywhere that Mary went
5 It followed her to school one day
7 It made the children laugh and play
```

Suppose this is the contents of the file LAMB.TXT:

```
2 Its fleece was white as snow
```

Follow this procedure to merge the files MARY.TXT and LAMB.TXT:

Step 1 Use the VEW Command to access MARY.TXT.

Step 2 Type an INCL string in MARY.TXT requiring the contents of LAMB.TXT. For example:

```
1 Mary had a little lamb
%%lamb.txt
3 And everywhere that Mary went
5 It followed her to school one day
7 It made the children laugh and play
```

Step 3 Strike [ESC][ESC]

Step 4 Type ex onto the VEW command line.

Step 5 Type the following onto the CIP command line:

```
incl mary.txt rhyme.txt
```

Step 6 Strike [RETRN]

This message appears on the screen:

```
     Including file LAMB.TXT
```

Step 7 After a moment, the cursor appears on the screen next to a right angle bracket.

Use the VEW or TYPE Commands to check the contents of RHYME.TXT. It should appear as follows:

```
     1 Mary had a little lamb
     2 Its fleece was white as snow
     3 And everywhere that Mary went
     5 It followed her to school one day
     7 It made the children laugh and play
```

Using Prompts

```
> incl
Source        > mary.txt
Destination   > rhyme.txt
Including file LAMB.TXT
```

This command performs the function described in the previous example.

Notes on Usage

INCL scans the source file for all lines beginning with "%%" and replaces that line with the contents of the file designation following the "%%" in the destination file. The included files and the source files are unchanged.

The destination file is created and contains a copy of the source file with the specified files included. INCL creates the file you specify as the destination; therefore, the file need not exist when you execute INCL. However, if the file does exist, INCL creates another version of it.

As long as you use complete file designations for the files specified in the parameters or in the include strings, you need not be concerned with what your default directory is when you execute INCL.

An INCL file may also include another INCL file. You can merge any number of files, but INCL is not recursive. In other words, you cannot put an INCL command string in file X that requires file Y if there is a command string in file Y that requires file X.

Related CIP Commands

vew	Edit a text file

install

Functional Description

Use this command to list, cancel, or grant privileges on privileged files.

Command Line Syntax

Mnemonic	install			
Required parameter	File List (optional if Function is :list)			
Optional parameter	Function (:insert, :delete, :list)			
Switches				
Files to be used	:before= :exclude=	:uic= :siteid=	:privilege= :since=	:confirm :mod
Display	:log	:pause		
Function modifiers	:auto			

Parameters

File List	Function	Required. (Optional if Function is :list.) Use this parameter to specify the files you want to install.
	Default	None. (All files if Function is :list.)
	Syntax	Type a standard file designation syntax. Wildcard symbols are allowed.

Function

Function Optional. Use this parameter to specify whether you want to list, cancel or grant privileges on the file.
Default :insert
Syntax: Type one of the following:

:list to list the installed files

:delete to cancel the installed privileges

:insert to grant certain privileges

Switches

:auto

Function Use this switch to perform the desired operation without any confirmation.
Default :noauto
Syntax Type **:auto**

:before=

Function Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.
Default All files that match the File List parameter are selected.
Syntax Type **:before=** followed by a date or time in the standard date and time syntax.

:confirm

Function Use this switch to confirm or deny the desired operation on each file.
Default :noconfirm
Syntax Type **:confirm**

:exclude=

Function Use this switch to exclude files from the File List parameter.
Default All files that match the File List parameter are selected.
Syntax Type **:exclude=** followed by a list of file designations separated by commas. Wildcard characters are allowed.

:log

Function Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

Default The value specified by the OPTION command.

Syntax Type **:nolog** or **:log** to override the default.

:mod

Function Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.

Default :nomod

Syntax Type **:mod**

:pause

Function Use this switch to stop the display after each screen of information. The display resumes when you press any key.

Default The value specified by the OPTION Command.

Syntax Type **:pause** or **:nopause** to override the default.

:privilege=

Function Use this switch to set the privileges for a process.

Default :privilege=none

Syntax Type **:privilege=** followed by **all**, **none** or by any of the following, separated by commas:

SETPRIV Set Privilege - The user can set any other privilege for his own or any other process.

SYSTEM System Access - Grant system-access to files, i.e., system user is one of four classes of users.

READPHYS Read Physical - The user can read physical blocks on the disk, or other media, and copy system tables into his own process memory.

WRITEPHYS Write Physical - This allows the user to write physical blocks on the disk, or other media, format disks, and update the system's internal tables.

SETPRIOR Set Priority - The user can change his process priority, timeslice, or become a realtime process.

CHNGSUPER Supervisor - Allows the user to change his run mode to supervisor mode. Once the user is in supervisor mode, he has unrestricted access to the system.

BYPASS Protection Bypass - The user can inspect, modify, and delete any files on the system.

OPERATOR Operator - Allows the process to set the system clock, format disks, and set the scheduling priority ratio, flush cache, install files, mount "special" disks, execute _MNTMNT SVC, mount devices with non-installed drivers, set process names, set network window sizes, and define rotor lists.

ALTUIC Alternate UIC - The process can access a file as though the process has the same UIC as the owner of the file.

WORLD World Access - This allows the user to affect processes outside his group.

GROUP Group Access - The process can affect any process whose group ID matches the group ID of his own process.

NETWORK Network - The process is allowed to perform remote operations.

SETATTR Set Attributes - The process can change its attributes.

:since= Function Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.

Default All files that match the File List parameter are selected.

Syntax Type **:since=** followed by a date or time in the standard syntax.

:siteid= Function Use this switch to specify the system on which the action is to take place.

Default The system on which the calling process is executing.

Syntax Type **:siteid=** followed by a numeral.

:uic=	Function	Use this switch to select only those files that are part of the given list and are owned by the specified user or list of users.
	Default	All files that match the specified list are selected.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.

Examples

> install sysexe/dm.exe :priv=chngsuper

This command installs the DM image file with CHNGSUPER privilege. The following message appears on the screen:

```
     _DS0/SYSEXE/DM.EXE.11 installed
```

Files are installed with certain privileges so a user can execute "safe" commands without having the required privileges himself. This allows protection to the system manager yet allows the users access to useful image files that require privileges.

For example, the command DM requires the privilege CHNGSUPER. DM is useful to most users and is considered safe because it displays memory but does not modify it. If the DM image file is installed with CHNGSUPER privilege, users without this privilege can still execute DM.

> install d* :list

This command lists the installed files and their corresponding privileges for all filenames beginning with the letter D. The following display appears on the screen:

```
_DS0/SYSEXE/DINIT.EXE.5
               NOSETPRIV      NOSYSTEM         READPHYS     WRITEPHYS
               NOSETPRIOR     NOCHNGSUPER    NOBYPASS       OPERATOR
               NOALTUIC       NOWORLD        NOGROUP

_DS0/SYSEXE/DM.EXE.11
               NOSETPRIV      NOSYSTEM       NOREADPHYS   NOWRITEPHYS
               NOSETPRIOR       CHNGSUPER    NOBYPASS     NOOPERATOR
               NOALTUIC       NOWORLD        NOGROUP
```

Note that only the filename is used in the wildcard selection. Use a slash, /, to specify the device and directory fields. Suppose you typed this command:

```
     install _ds0/sysexe/d* :list
```

Then INSTALL would display all the installed files from the SYSEXE directory on device _DS0 whose filenames begin with D.

In the display all possible privileges are listed. If there is a NO preceding the privilege name, the image does not have that privilege. In the example above DM.EXE has CHNGSUPER (change to supervisor) privilege and no others.

```
> install dm.exe :del
```

This removes DM.EXE from the system list of installed images. The following messages appear on the secreen:

```
     _DS0/SYSEXE/DM.EXE.11
     Deinstall (Y or N)? > y
     _DS0/SYSEXE/DM.EXE.11 deinstalled.
```

If you type **install :list** on the CIP command line, DM.EXE.11 won't be displayed. DM.EXE in SYSEXE no longer has installed privileges, so a user must have CHNGSUPER privilege to execute DM.

Using Prompts

```
> install
File list     > sysexe/dm.exe :priv=chngsuper
_DS0/SYSEXE/DM.EXE.11 installed
```

This performs the same function as the first example.

Notes on Usage

The actual installation of files should only be done by the system manager.

In addition to image files that must be installed with special privileges, the drivers on your system should also be installed. Drivers do not need to be installed with any privileges, but they must exist in the list of installed images. If a driver is not installed, a user must have operator privilege to use a driver.

The files are installed by FCB sequence number so every file is distinct. This is important for two reasons:

1. When you install a file that is already installed, it does not replace the old version of the file. Therefore, if you want to replace an installed file with an updated version of the file, you must install the new file and delete the old file.

2. When putting out a new version of a file, the new version must be installed. Renaming a new version of an image file to have the same version number as a previously installed file does not give the new file any privileges. The new version of the file must be installed from scratch. All privileges needed by the file must be specified using the :privilege= switch.

If you delete an installed file, it does not affect the contents of the file but all privileges it may have had are canceled.

Related CIP Commands

None.

keygen

Functional Description

Use this command to generate public and private key sets for use by CRYPT and the networking portions of WMCS.

Command Line Syntax

Mnemonic	keygen		
Switches			
Parameter specification switches	:username=	:password=	:key=
Function specification switches	:system	:delete	
Display control	:inform	:log	:sysprompt=

Parameters

None.

Switches

:delete

Function — Use this switch to remove either user or system public/private keys from their appropriate files. When no public/private keys are present, the public/private features of the CRYPT utility cannot be used for the user with no key, nor can secure node validation-authentication be done between nodes.

Default — :nodelete

Syntax — Type **:delete**

:inform

Function — Use this switch to send a message to sys$prompt upon completion of key generation or deletion.

Default — :noinform

Syntax — Type **:inform**

:key=

Function — Use this switch in user mode to specify the key which will be used to encode and decode the private portion of the RSA public/private key pair which KEYGEN will generate. If the :key= switch is used on the command line, the key will appear on the screen or command file from which it is read. However, if KEYGEN is run in user mode and the :key= switch is not used, it prompts for the key and does not echo it to the screen.

NOTE: Whether entered on the command line or via a prompt, the key must be greater than or equal to 16 characters in length. Choose the key such that it will be easily remembered but difficult to guess. Numbers, punctuation, and special characters make the key more secure. The characters in the key are case sensitive.

Default — None when in user mode, not needed for system mode.

Syntax — Type **:key=** followed by a string of 16 or more characters.

:log

Function — Use this switch to send logging markers to sys$output while keys are being generated. A period, ., is written during generation of the first key, while an asterisk, *, is written during generation of the second key.

Default :log

Syntax Type **:nolog**

:password=

Function Use this switch in user mode to specify the password associated with the user specified in the :username= switch. If the :password= switch is used on the command line, the password will appear on the screen or command file from which it is read. However, if KEYGEN is run in user mode and the :password= switch is not used, it prompts for the password and does not echo it to the screen.

Default None when in user mode, not needed for system mode.

Syntax Type **:password=** followed by a valid password.

:sysprompt=

Function Use this function to direct the message from the :inform switch to a file or device other than the one assigned to sys$prompt at the time KEYGEN was started.

Default Messages from the :inform switch are sent to sys$prompt.

Syntax Type **:sysprompt=** followed by a valid filename or devicename.

:system

Function Use this switch to specify that system public/ private keys are to be generated. Absence of this switch implies that user keys are to be generated. The use of this switch requires system privileges in addition to bypass and world privileges. When this switch is present, the :username=, :password=, and :key= switches are ignored on the command line and are not prompted for.

Default :nosystem

Syntax Type **:system**

:username=

Function Use this switch in user mode to specify the user for which a set of RSA public/private keys are to be generated. The user must already have a valid account in sys$disk/syslib/ uaf.dat.

Default None when in user mode, not needed for system mode.

Syntax   Type **:username=** followed by a valid username.

Examples

```
> &keygen :system :nolog :inform
```

This command will fork KEYGEN as a background process to generate system keys. It will send a message to sys$prompt when it has completed.

```
> keygen :user=john :password=quefob :key="snazzle_razzle_frum"
```

This command generates a set of public/private keys for John if his password is "quefob". Logging markers will be sent to the screen while the keys are being generated. When the keys have been successfully generated, they are placed in sys$disk/syslib/rsakeys.dat and the private key is encoded using the phrase "snazzle_razzle_frum".

```
> keygen :user=mike :password=foobar :delete
```

This command will delete the public/private keys belonging to user Mike, provided his password is "foobar".

Using Prompts

```
> keygen
Username      > sam
Password      > raborit
Key           > MY KEY IS abracadabra
```

Note that the password and key are entered but will not be echoed to the screen.

Notes on Usage

The intent of KEYGEN is to generate public/private keys for use by other utilities and the networking portions of WMCS. This utility is capable of generating keys for individual users as well as for the system. Public/ private keys are used in the RSA (Rivest, Shamir, and Adleman) algorithm for encryption/decryption. This algorithm is particularly useful because of its capacity to produce digital signatures. For a fuller discussion of public/private keys and digital signatures, see the command description for CRYPT.

KEYGEN can be used in either of two modes. The first and most commonly used is the user mode. This mode is denoted by the absence of the :system switch. In order to prevent malicious users from altering another's keys, a valid username password sequence is required in user mode. The key entered to KEYGEN is used to encode the private portion of the public/private key pair which is generated and stored in /syslib/ rsakeys.dat. The same key supplied to KEYGEN must also be supplied to CRYPT when encrypting or decrypting using a private portion of the RSA key pair. Because /syslib/rsakeys.dat is write-protected to all processes, KEYGEN must be run with bypass and world privileges in user mode and as such, should probably be an installed process.

KEYGEN's second mode is system mode. When the :system switch is specified, KEYGEN will generate RSA keys to be used by the networking software of WMCS. These keys are used in node verification and authentication routines when two nodes first contact each other. The keys are stored in /rootdir/nodconfig.xxx and /rootdir/netconfig.xxx. In system mode KEYGEN must have system privilege in order to write its keys to the appropriate files.

KEYGEN requires extensive amounts of processor time in order to generate its keys. Ten to sixty minutes are not atypical, although many times the keys will be generated in just a short time. In order to assure the interactive user that KEYGEN is alive and well, a series of periods or asterisks are repeatedly displayed and erased on the screen as the keys are generated. The period indicates that the first key is being generated. An asterisk appears while the second keys being built. The second key usually takes quite a bit longer to generate than the first. The :log switch can be used to disable display of the logging markers. Should KEYGEN be run as a background or batch job, logging is usually disabled and the :inform switch is used to inform the user that KEYGEN has finished. The :sysprompt= switch may be used to direct the termination message to any desired file or device.

Related CIP Commands

crypt	Encrypt/decrypt files using public/private keys

link

Functional Description

Use this command to create executable image files from the object modules created by the Pascal compiler and the ASM assembler.

Command Line Syntax

Mnemonic	link			
Required parameter	File List			
Switches	:bitmaps	:caseless	:crossref	:imagename=
	:impurebase=	:library=	:protectpure	:purebase=
	:relocation	:stacksize=	:symbol=	

Parameters

File List	Function	Required. Use this parameter to specify the names of the files (object modules) to be linked.
	Default	None.
	Syntax	Type a file designation. Wildcard symbols are not allowed. If no extension is specified, .MRL is assumed. File lists are allowed. The image file produced by this command is given the name of the first file (with a .EXE file extension) in the list. The files specified for this parameter are loaded in the order in which you specify them. The first object module you specify on the command line is the module to which control is

transferred when the program, i.e. the image produced by LINK, is executed.

Switches

:bitmaps Function Use this switch to suppress the bitmaps generated at the front of the image file.
Default :bitmaps
Syntax Type **:nobitmaps**

:caseless Function Use this switch to specify that upper- and lowercase characters in symbols are not distinct, i.e., uppercase A and lowercase a are considered the same.
Default :caseless Syntax Type **:nocaseless**

:crossref Function Use this switch to generate a symbol cross reference file.
Default :crossref
Syntax Type **:nocrossref**

:imagename= Function Use this switch to specify the name of the image file.
Default The name of the first object module specified.
Syntax Type **:imagename=** followed by a file designation. The filename you type is also assigned to the cross reference file. The file extension for the image file will be .EXE if none is specified.

:impurebase= Function Use this switch to specify the address at which the impure program segment is to begin.
Default :impurebase=1048576 (in decimal), or $100000 (in hexadecimal)
Syntax Type **:impurebase=** followed by a numeral indicating the base address of the impure program segment. All impure data are linked to be loaded beginning at this address in memory.

:library= Function Use this switch to specify a library definition file that will be used in addition to the standard system libraries. Standard system libraries are defined in SYS$DISK/SYSLIB/ LINKLIB.DEF.

Default :library=sys$disk/syslib/linklib.def

Syntax Type **:library=** followed by a file designation. If a file extension is not specified, .DEF is used.

:protectpure Function Use this switch to specify that the pure code not be shared.

Default Reentrant code is marked as shareable, and is protected by the operating system.

Syntax Type **:noprotectpure** to specify non-pure code.

:purebase= Function Use this switch to specify the address at which the pure program segment is to begin.

Default :purebase=4096

Syntax Type **:purebase=** followed by a numeral indicating the base address of the pure program segment. All pure data are linked to be loaded beginning at this address in memory.

:relocation Function Use this switch to indicate that relocation data should be written to the image file. The relocation information is required only for the unmapped system 150.

Default :relocation

Syntax Type **:norelocation**

:stacksize= Function Use this switch to specify the amount of stack space (in bytes) to reserve for the program created by LINK.

Default :stacksize=4096

Syntax Type **:stacksize=** followed by a numeral.

:symbol= Function Use this switch to define a symbol. For programs written in Pascal, this is the mechanism used to define the heap. The linker defines the symbol named in this switch and reserves the specified amount of space for the symbol.

Default None.

Syntax Type **:symbol=** followed by a list of symbol names and sizes pairs. The name is a symbol name to be defined. The size is a numeral indicating the number of bytes to reserve for the symbol. Separate the size from the symbol name with a colon. For example:

```
:symbol=_heap:$1000,test:256.
```

Examples

```
> link simulate :symbol=_heap
```

This command links the object module named SIMULATE.MRL, and defines a symbol, _HEAP, and reserves zero bytes for it. The system library definition file (syslib/linklib.def) is used to reference various system libraries. This command generates two files named SIMULATE.MCR (which contains the link map and symbol cross reference), and SIMULATE.EXE which is the program image file. This is the method used to link Pascal programs.

```
> link macro,math :symbol=_heap :norelocation
```

This command links the two modules MACRO.MRL and MATH.MRL with the standard system libraries, defining the symbol _HEAP. Relocation information is not written to the image file. The image file is named MACRO.EXE.

Using Prompts

```
> link
File list      > simulate :symbol=_heap
```

This command performs the same function as the first example.

Notes on Usage

LINK reads object modules, resolves references between modules, and then creates an image file.

LINK is used to link Pascal and WICAT assembler programs. Other languages use other linkers.

Read the Linker Programmer Reference Manual for a complete description of the linker program.

If undefined references remain after you link all the specified files, and after all possible references from the specified libraries are resolved, the undefined references are written to SYS$OUTPUT. LINK then terminates, leaving these unresolved. Running a program with unresolved references will probably result in a crash dump.

A link map is generated as part of the cross reference file (the file with a .MCR file extension) that names all files that were loaded, all symbols defined in each file, how large each module is (in bytes of pure code and impure code), and where each module is linked to be loaded.

The linker uses the logical name SYS$DISK to find the device containing /SYSLIB/LINKLIB.DEF, the link definition file. The filenames contained in the library definition files can also be prefixed with SYS$DISK.

The default file extension for files to be linked is .MRL. Files with a .MRL file extension are the object files produced by compilers and the assembler. The default extension for the image file is .EXE.

Related CIP Commands

prelink	Prepare libraries for linking

load

Functional Description

Use this command to load a software product from backup media and set up the system so that the product is ready to be used.

Command Line Syntax

Mnemonic load

Parameters

None.

Switches

None.

Examples

Follow this procedure to load a software product from backup media.

Step 1 | Type the following on the command line and strike [RETRN]:

```
load
```

This display appears on your screen:

```
**************** SOFTWARE PRODUCT INSTALLATION UTILITY ****************

What device would you like to use?

          ==================================
           DEVICE |        DESCRIPTION
          ----------------------------------
             dx0  | 5.25 inch Diskette
             ct0  | Cartridge Tape
             mt0  | Cipher Tape
           other  | (none of the above devices)
          ==================================

Which DEVICE do you wish to load the product from (press [SPACE]
             to change the value or [RETRN] to accept the value): dx0
```

Step 2 Strike the spacebar until the letters next to the colon (in the display above) change to the name of the drive for your tape or diskette.

Step 3 Strike [RETRN]. The following line appears on your screen:

```
     Please enter the product identifier(s):
```

Step 4 Type the three-character product identifier (described in the installation instructions for the specific product you are loading).

Step 5 Strike [RETRN]. A message like the following appears on your screen:

```
     _DX0 Mounted. Label is "wsa_5.1.0_vol1".
     Is this the correct volume (Y or N) ? >
```

Step 6 Type Y. A message like the following appears on your screen:

```
    ************* Installing WSA from device _DX0 *************

    What disk number do you wish to load the product on? (press
      [SPACE] to change the value or [RETRN] to accept the value: 0
```

Step 7 Strike the spacebar until the number next to the colon (in the display above) changes to the number of the disk you are loading the product on.

Step 8 Strike [RETRN]. LOAD is now copying the product onto the disk you specified.

If the product fills more than one diskette or tape, your terminal will beep and display a message like the following each time you need to insert another diskette or tape:

```
Insert volume 2
Press RETURN when ready
```

Step 9 Remove the diskette or tape that is in the drive and put the next volume into the drive and strike [RETRN].

Follow the instructions on your screen to finish the load and display the verification results.

When the product is completely loaded, a display like the following appears on your screen:

```
Purging release directories...

****************** LOAD OF WSA 5.1 COMPLETED ******************
```

LOAD automatically dismounts the last volume of media so that you may remove it and store it in a safe place for future reference.

Using Prompts

None.

Notes on Usage

LOAD should only be used by the system manager to ensure that all of the files that compose the release are included in the product and are accessible to the users after the product is loaded.

The APPLICUP.COM file in the directory /SYSLIB/ will automatically be updated so that the product will be available even after the system has been restarted.

Related CIP Commands

bkup	Create a backup copy of a software product

log

Functional Description

Use this command to terminate the CIP in which you are working.

Command Line Syntax

Mnemonic	log	
Switches	:log	:perm

Parameters

None.

Switches

:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:perm	Function	Use this switch to make the change permanent or to make it persist even after the current CIP ends execution (unless the current CIP is your user process).
	Default	:noperm

Syntax Type :perm

Examples

> log

This command terminates the CIP the user is executing and displays the following message:

```
USER logged off at 15-Apr-1985 08:40:37
```

Note that the username, date, and time are displayed when the user logs off.

> log :nolog

This command terminates the CIP the user is executing but suppresses the display of the log message.

> log :perm

If this command is executed from inside a command file, it terminates all levels of the command file back to the previous interactive CIP. If LOG is executed from an interactive CIP, it terminates that CIP only. Then this message appears:

```
USER logged off at 15-Apr-1985 08:40:37
```

Using Prompts

None.

Notes on Usage

As the CIP is terminating, it will execute a command file called LOGOFF.COM located in SYS$DISK/SYSLIB. This command file in turn executes SYS$DISK/SYSLIB/LOCALOFF.COM and USEROFF.COM located in the user's home directory.

NOTE: If the LOG command is executed from within a command file, LOGOFF.COM, LOCALOFF.COM, and USEROFF.COM are not executed.

Related CIP Commands

cip	Execute a copy of CIP.EXE

logflush

Functional Description

Use this command to monitor mounted TTY class devices for attempts to log on to the system, and to periodically flush disk devices.

Command Line Syntax

Mnemonic	logflush
Optional parameter	Logon Interval
Optional parameter	Flush Interval
Switches	:timeslice=

Parameters

Logon Interval	Function	Optional. Use this parameter to specify the interval, in seconds, in which LOGFLUSH searches for terminals to log on.
	Default	5
	Syntax	Type a positive integer.
Flush Interval	Function	Optional. Use this parameter to specify the interval, in seconds, in which LOGFLUSH flushes all disks on the machine.
	Default	30
	Syntax:	Type a positive integer.

Switches

:timeslice=	Function	Use this switch to specify in milliseconds the timeslice LOGFLUSH will have.
	Default	:timeslice=100
	Syntax	Type **:timeslice=** followed by a number less than 650.

Examples

```
> logflush
```

LOGFLUSH has two modes. It can run indefinitely as a background process (specified by the ampersand, &) always searching for log ons and flushing at the specified intervals. Or it can run once, flush every disk on the machine, and then stop (as in the above example). This allows the user to force a flush at any time.

```
> &logflush 10
```

This command runs LOGFLUSH in the background mode. The log on searching interval is specified at 10 seconds. The flushing interval is the default, 30 seconds.

Using Prompts

None.

Notes on Usage

The PID of a LOGFLUSH that is forked to run in the background is always 1. If you try to fork a new LOGFLUSH while one is already running, an error is reported.

You do not have to kill LOGFLUSH to use the RECOVER command.

Related CIP Commands

logon	Log on to a system

logline

Functional Description

Use this command to append a time stamped message to the end of the specified file.

Command Line Syntax

Mnemonic	logline	
Required parameters	File Desig. Message	
Switches	:create	:rewind

Parameters

File Desig.

Function: Required. Use this parameter to specify the name of a file to which the time stamped message is to be appended. This file must already exist. That is, if the file does not exist, an error is reported. The file is not created (unless the :create switch is specified).

Default: None.

Syntax: Type a file designation. File lists and wildcard symbols are not allowed.

Message

Function: Required. Use this parameter to specify the message to be written to the file.

Default: None.

Syntax: Type a string. If the string contains spaces you must enclose it in double quotation marks.

Switches

:create	Function	Use this switch to create a new file.
	Default	:nocreate
	Syntax	Type **:create**
:rewind	Function	Use this switch to rewind the tape device.
	Default	:norewind, i.e. the tape device begins at the current position instead of at the beginning of the tape.
	Syntax	Type **:rewind.**

None.

Examples

```
> logline logfile.dat "Phone call from Robert Hatch."
```

This command appends the following message to the end of the file named LOGFILE.DAT in the default directory:

```
     Monday 15-Apr-1985 10:16:22.30   Phone call from Robert Hatch.
```

```
> logline sys$disk/syslib.users/'sys$username'.dat "Logged off 'sys$input'"
```

This command line appends a message to a file whose name is a derivative of the user's logon username. The message contains the name of the terminal port used by the current user. If sys$username is JOHN and sys$input is _TT4, the above command line will append a message to the file named SYS$DISK/SYSLIB.USERS/JOHN.DAT. The message will be:

```
     Monday 15-Apr-1985 10:16:22.30   Logged off _TT4
```

Using Prompts

```
> logline
File desig     > logfile.dat
Message        > "Phone call from Robert Hatch."
```

This performs the same function as the first example.

Notes on Usage

Use LOGLINE to log events on your system. For instance, you can log a message each time a user logs on or logs off of your system. You can log your activities during the day.

Related CIP Commands

None.

logon

Functional Description

Use this command to log on to a system.

Command Line Syntax

Mnemonic	logon
Optional parameter	Username

Parameters

Username	Function	Optional. Use this parameter to specify the name of the user.
	Default	None.
	Syntax	Type the ASCII characters that represent the username.

Switches

None.

Examples

> **> logon**

This command causes a prompt like the following to appear on the screen:

```
Welcome to SYSTEM NAME
Username:
```

LOGON is executed when a user first tries to access a port. LOGON can also be executed from the command line to log on as another user without logging off the original user.

Logging on is usually a four-step process.

1. LOGON prints the name of the system as a greeting to the user.

2. LOGON prompts for the username. The username is the name assigned by the system manager to each person who uses the system. As you enter your username it is echoed on the screen.

3. If you or the system manager has set up security for your account, (see the PASSWORD and USERPROF command descriptions) LOGON then prompts for the user's password. As you enter your password, the characters are not echoed on the screen. This helps to prevent password stealing.

   If you enter your name or password incorrectly, the greeting does not reappear, and LOGON returns to step 2 and again prompts for the username.

   LOGON allows five unsuccessful attempts, then makes the user wait for 30 seconds to try logging on again.

   If no password is set up on your account, the prompt for the password does not appear and you can go directly to step 4.

4. After checking to see if the user can pass all security checks (see USERPROF command for available security checks) for a terminal, LOGON sets the device status of the terminal, sets up process protection, and assigns the following logical names: syscip, sysusername, and sys$home. Then the command line, as specified in the User Authorization File, is forked.

For example, assume CHRIS was logged on to terminal _TT5 before LOGON was executed (as in the previous example). After GUEST successfully logged on over CHRIS, _TT5 would be owned by GUEST and the privilege, protection, home directory, username, and CIP would be the ones assigned to GUEST in the UAF file. (See the USERPROF command description for information on the UAF file.)

When GUEST logs off the system, CHRIS is once again the default user, and all values set by LOGON are restored to their previous state (before LOGON was executed).

In this instance, the LOGON Command performs the same function as logging off the system, logging on as GUEST, logging off, and logging back on as CHRIS.

> logon guest

This performs the same function as the previous example. The only difference is the username was entered on the command line rather than in response to the prompt. A different prompt appears:

```
Welcome to SYSTEM NAME
Password:
```

Using Prompts

None.

Notes on Usage

Some terminals may be set up with no security with the SYSPROF command specifying a default user. LOGON does not prompt for the username or password on these terminals but sets up the logical names and port characteristics just as it does on terminals with security.

The standard password checking form of protection may be replaced by a user supplied program. This is done by entering the name of the security checking program in the SECURITY field of the UAF file (See USERPROF command for more information). Instead of prompting for a password and verifying that the password is correct, LOGON spawns the user's security program and checks that the status returned from the program is zero. If the status is other than zero, the user is not allowed to log on.

> NOTE: The ability to log on from the command line is useful if another person needs to use your terminal for a short time and wants to have the UIC, privileges, priority, etc. of his user account.

Related CIP Commands

userprof	Set up the user profiles on a system
sysprof	Set up a system profile
password	Change the user password

makedsr

Functional Description

Use this command to create drivers and other system files. MAKEDSR can convert .EXE image files into .DSR files.

Command Line Syntax

Mnemonic	makedsr	
Required parameters	Source Destination	
Switches		
Function control	:base=	:simple

Parameters

Source	Function	Required. Use this parameter to specify the file to be changed.
	Default	None.
	Syntax	Type the standard syntax for a WMCS file designation. No wildcard symbols are allowed.
Destination	Function	Required. Use this parameter to specify the file to be created.
	Default	None.
	Syntax	Type the standard syntax for a WMCS file designation. No wildcard symbols are allowed.

Switches

:base=	Function	Use this switch to specify the base address at which the image file was linked.
	Default	:base=$1000
	Syntax	Type **:base=** followed by a hexadecimal number.
:simple	Function	Use this switch to only remove the bit maps from the front of the file. No other alterations are made to the file.
	Default	:nosimple
	Syntax	Type **:simple**

Examples

```
> makedsr file.exe file.dsr
```

This command creates a .DSR file from FILE.EXE

Using Prompts

```
> makedsr
Source      > file.exe
Destination > file.dsr
```

This performs the same function as the previous example.

Notes on Usage

The format of driver files differs from the format of image files. The purpose of MAKEDSR is to convert one binary format into the other.

An image file format consists of bit maps, code, data, and optional relocation information.

Driver files begin with an identification field, size-of-driver field, and size-of-relocation field. This is followed by the body of the driver, any relocations, and an optional driver-defined section.

The image which is to be changed into a driver must have the identification and size fields already built in. MAKEDSR removes the bit maps, copies data as specified in the first size field, copies the relocations and fills in the second size field, and then copies the remainder of the file.

The DSR file contains a binary image which is loaded into memory using the MOUNT SVC call. The first 8 bytes of this file contain size and ID information used to tell MOUNT how to load the file. These fields are defined as follows:

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|       header -- this block contains size information
|  size   name            purpose
|  ----   -----------     --------------------------------
|     2   code_size       size in bytes of the code segment
|     2   reloc_size      size in bytes of the relocations
|     4   id field        driver id number
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|       code -- this is body of the driver
|
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|       relocations -- long word addresses containing
|          offsets from header of absolute long references
|
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|       additional driver defined data -- this will be
|          copied.  may be used for download code, etc.
|
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

The above description applies to the file as it exists on the disk. After it is loaded into memory, the RELOC_SIZE field is zeroed and is used as a reference count. This is compatible with the original definition of the file which had no provision for relocation information. In the original definition, the RELOC_SIZE/REFERENCE_COUNT field was required to be zero.

The relocation information is a list of long word pointers to the addresses which need to be adjusted. The code section is assumed to start at location zero.

Following the code section and the relocation section, there may be additional code and data. The ICI driver is an example of a DSR file where the main body of the device driver is followed by a section of downloadable code. This code is sent to the intelligent device by the device driver when the device is initialized.

Related CIP Commands

install	List, cancel, or grant privileges on privileged files
link	Create executable image files
fstat	Inspect and change file status

memtest

Functional Description

Use this command to verify correct operation of the main (RAM) memory.

Command Line Syntax

Mnemonic	memtest	
Switches		
Control of memory test	:passes=	:wait=

Parameters

None.

Switches

:passes=	Function	Use this switch to specify how many times to test all of memory before the MEMTEST program terminates.
	Default	Never quits testing memory.
	Syntax	Type :passes= followed by a positive integer.
:wait=	Function	Use this switch to specify in minutes how long to wait between each pass of test memory.

Default	:wait=15
Syntax	Type **:wait=** followed by a positive integer.

Examples

> memtest

This command tests all of memory, pausing for 15 minutes between each pass. MEMTEST never terminates. It reports any error it receives to SYS$ERROR.

> &memtest >/systmp/memtest.log ^/systmp/memtest.log :wait=30

This command line character string executes MEMTEST in background mode (indicated by the ampersand, &) and assigns the program's output and error message to /SYSTMP/MEMTEST.LOG. It will pause for 30 minutes between each test.

Using Prompts

None.

Notes on Usage

MEMTEST examines unallocated pages in the system's memory (one 4-Kbyte page at a time) by performing the following tests:

1. Reading existing data in memory to check for parity errors.

2. Checking whether each bit will accept both a zero and a one.

MEMTEST reports all errors to SYS$ERROR. MEMTEST does not die when an error occurs, it reports the error and goes on to the next page. The error page is removed from the good page list so it won't be used again.

A sample error display is:

```
     Location 20936C contains 00100001 should be 00000001
```

NOTE: MEMTEST does not function properly on S2220's with cache CPU and pre-fetch memory. It works as documented on all other systems.

Related CIP Commands

dm	Display memory status
pstat	Display or set process attributes

mnt

Functional Description

Use this command to mount a device.

Command Line Syntax

Mnemonic	mnt
Required parameter	Devicename

Switches

Identify device	:driveid=	:driver=	:drivetype=	
Indicate device class	:special			
Set initial device status	:blocksize=	:writeprotect	:numtoretry=	:rawverify
	:errintenb	:speed=	:iopbcount=	:cache=
	:inbufsize=	:outbufsize=	:protection=	:print
	:job	:restart	:density=	
Miscellaneous	:log	:pipesize		

Parameters

Devicename	Function	Required. Use this parameter to specify the name of the device you wish to mount.
	Default	None.
	Syntax	Type the standard syntax for a devicenames. Wildcard symbols are not allowed.

Switches

:blocksize=

Function: Use this switch to specify the blocksize for the tape in bytes per block.

Default: The blocksize found in the tape label if the tape is mounted regular class. If the tape is mounted special class, :blocksize=1024.

Syntax: Type **:blocksize=** followed by a positive integer less than or equal to 4096.

:cache=

Function: Use this switch to specify the number of sectors of disk or tape cache to be maintained in memory.

Default: The number of cache sectors specified in the disk's or tape's boot block.

Syntax: Type **:cache=** followed by a positive integer.

:density=

Function: Use this switch to specify the density of the tape device in bytes per inch.

Default: :density=1600

Syntax: Type **:density=** followed by one of the following: 800, 1600, 3200, 6250, or 6400.

:driveid=

Function: Use this switch to specify the drive ID for the device.

Default: The drive ID listed for this device in the SYS$DISK/ROOTDIR/DEVCONFIG.NNN file, where NNN is the system model (100, 156, 160, etc.).

Syntax: Type **:driveid=** followed by the drive number, the driver configuration, and the board number. For example, :driveid=0A0. The following is a list of driver configurations:

Disks	
SMD	"A" is the only choice
FLNT	"A" is 8 inch floppy
	"B" is 5.25 inch floppy
	"C" is 3.25 inch floppy
WD3	"A" is the hard disk
	"B" is the floppy
IMI	"A" is the only choice

Tapes	
ADEI	"A" is fixed blocks (1022 bytes/blk)
	"B" is variable blocks
CIPH	"A" is low density
	"B" is high density (old method)

:driver=

Function Use this switch to specify the device driver for this device.

Default The device driver listed in the SYS$DISK/ROOTDIR/DEVCONFIG.NNN file (where NNN = 100, 156, or 160).

Syntax Type **:driver=** followed by the name of the device driver file. No wildcard symbols are allowed. The names of available drivers on your system are found in SYS$DISK/SYSDSR/.

:drivetype=

Function Use this switch to specify the drive type for the device you are mounting. (This is done so the driver can be properly configured if the drive table for this drive type is not in the boot block.)

Default The drive type listed for this device in the file SYS$DISK/ROOTDIR/DEVCONFIG.NNN.

Syntax Type **:drivetype=** followed by one of the following valid drive types:

What you type	Description
FLOP09A	5 1/4 inch floppy
FLOP09B	5 1/4 inch floppy (5 sector)
FLOP015	8 inch floppy
WIN12	CMI 10 Mbyte 5 1/4-inch Winchester
WIN19	CMI 15 Mbyte 5 1/4-inch Winchester
WIN30	28 Mbyte 5 1/4-inch Winchester
WIN43	39 Mbyte 5 1/4-inch Winchester
SMD84B	76 Mbyte SMD disk
SMD168B	152 Mbyte SMD disk
SMD474B	421 Mbyte SMD disk
IMI20	20 Mbyte IMI disk
IMI40	40 Mbyte IMI disk

:errintenb

Function Use this switch on TAPE class devices to enable error interrupts. If enbabled, blocks with errors are not returned to users. If disabled, blocks with errors are returned to users.

Default Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:noerrintenb** to disable, or **:errintenb** to enable.

:inbufsize=

Function — Use this switch to specify the size of the input buffer for SYNC class devices in bytes (i.e., specify the maximum number of characters that can fit in the device's input buffer).

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:inbufsize=** followed by a positive integer.

:iopbcount=

Function — Use this switch to specify the number of input-output parameter blocks (essentially the maximum number of pending operation requests) on a disk.

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:iopbcount=** followed by a positive integer.

:job

Function — Use this switch to specify that this queue device is a job queue (as opposed to a print queue).

Default — :print

Syntax — Type **:job**

:log

Function — Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

Default — The value specified by the OPTION command.

Syntax — Type **:nolog** or **:log** to override the default.

:numtoretry=

Function — Use this switch on all classes of devices to specify the number of times to retry an operation (such as read or write) on a device before giving up and reporting a hard error.

Default — Driver dependent unless changed by previous DSTAT command.

Syntax — Type **:numtoretry=** followed by a positive integer.

:outbufsize= Function Use this switch to specify the size of the output buffer for SYNC class devices in bytes (i.e., specify the maximum number of characters that can fit in the device's output buffer).

Default Driver dependent unless changed by previous DSTAT command.

Syntax Type **:outbufsize=** followed by a positive integer.

:pipesize= Function Use this switch to specify the size of the pipe in bytes.

Default :pipesize=256

Syntax Type **:pipesize=** followed by an integer greater than or equal to 256.

:print Function Use this switch to specify that this queue device is a print queue (as opposed to a job queue).

Default :print

Syntax Type **:print**

:protection= Function Use this switch to specify the protection mask for the device. (On disks and tapes mounted regular class, only the owner--the UIC in the disk boot block or tape label--can change the protection of the device).

Default For disks and tapes mounted regular class the default is the protection mask found in the disk's boot block or the tape's label. For all other devices the default is the protection mask of the user (process) mounting the device.

Syntax Type **:protection=** followed by a protection mask.

Note that only those fields of the protection mask that are specified are altered. Fields not specified keep their original value.

For example, if you typed :protection=s:,o:rwe the system would have no privilege, the owner would have read, write and execute privilege, and the public and group privileges would remain unchanged.

:rawverify Function Use this switch to do read-after-write verification of the data written to DISK or TAPE class devices.

	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:rawverify** to enable, **:norawverify** to disable.
:restart	Function	Use this switch to specify that this queue device is to be restarted when the system is rebooted. Any jobs left in the queue are put into a hold state so they can be individually restarted. If you specify :norestart, any jobs that are left in the queue when the system is rebooted are removed from the queue without being executed.
	Default	:restart
	Syntax	Type **:norestart**
:special	Function	Use this switch to mount the device as special class, as opposed to regular (such as class DiskSpecial as opposed to class Disk). Special class means the device is treated as a raw device. File systems on disks and tapes mounted special are not recognized by WMCS.
	Default	:nospecial
	Syntax	Type **:special**
:speed=	Function	Use this switch on TAPE class devices to specify the speed of a tape in inches per second.
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:speed=** followed by one of the following: 12, 25, 30, 50, 90, 100, or 125.
:writeprotect	Function	Use this switch on DISK and TAPE class devices to specify that the device cannot be written on. (This is a software check. If the hardware is write-protected, this flag is set.)
	Default	Driver dependent unless changed by previous DSTAT command.
	Syntax	Type **:writeprotect** to enable, **:nowriteprotect** to disable.

Examples

```
> mnt _dxl
```

This command mounts the device _DXl as regular class (not special). In this case, information about _DXl needed to mount it is found in the SYS$DISK/ROOTDIR/DEVCONFIG.NNN file because it is not explicitly specified. The following message appears:

```
_DXl Mounted. Label is "WMCS_5.0".
```

```
> mnt _dxl :special
```

This command mounts device _DXl as special class (such as DiskSpecial instead of Disk). The following message appears:

```
_DXl Mounted. Label is "_DXl".
```

```
> mnt _ct0 :driver=adei$100 :driveid=0b0
```

This command mounts a tape, devicename _ct0, as tape class using driver SYS$DISK/SYSDSR/ADEI$100.DSR. The tape volume is in the drive designated by drive ID 0b0 (drive 0 on ADEI controller, configuration b on board 0).

```
> mnt _dxl :driver=wd3$156 :driveid=lb0 :drivetype=flop09a
```

This command mounts a disk, devicename _dxl, as class Disk (not DiskSpecial). The disk is of type FLOP09a (5 1/4-inch floppy disk with 600 Kbyte unformatted capacity) which requires driver SYS$DISK/SYSDSR/WD3$156.DSR and is in the drive designated by drive ID lb0 (drive l on WD3 controller, configuration b on board 0).

```
> mnt _dxl :writeprotect :cache=50
```

This command mounts _DXl, sets it to be write-protected and sets a 50 sector cache in memory.

```
> mnt _tt23 :inbufsize=1024 :outbufsize=1024
```

This command mounts _TT23, with an input buffer size of 1024 bytes (characters) and an output buffer size of 1024 bytes (characters). (This example works only if _TT23 is a sync class device.)

Using Prompts

```
> mnt
Devicename    > _dxl
```

This command performs the same function as the first example.

Notes on Usage

If there is no SYS$DISK/ROOTDIR/DEVCONFIG.NNN file as is frequently the case when you boot a system from floppy or tape), you must specify the driver, drive ID, and (if mounting a disk which does not have a drive table in its boot block) the drive type.

If you are executing MNT from tape, the driver file must be after the MNT.EXE file on the tape.

If you are mounting a disk or tape as regular class, the label displayed by the log message is the one found in the disk boot block or tape label. If you are mounting any other type of device, either regular or special class, or if you are mounting a disk or tape as special, the label displayed is the devicename.

The :print and :job switches are mutually exclusive.

Related CIP Commands

dmnt	Dismount (remove) a device
dev	Display the list of mounted devices
dstat	Display information about a mounted device

mstat

Functional Description

Use this command to display information about the Named Shared Memory (NSM) areas on the system.

Command Line Syntax

Mnemonic	mstat	
Optional parameter	Region Name	(Required if :owner= or :protection= are used.)
Switches		
Output control	:log	:pause
Function control	:owner=	:protection=

Parameters

Region Name	Function	Optional. (Required if :owner= or :protection= switches are used.) Use this parameter to specify the name of a Named Shared Memory region (NSM).
	Default	All memory areas are displayed.
	Syntax	Type a valid memory region name.

Switches

:log

Function — Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

Default — The value specified by the OPTION command.

Syntax — Type **:nolog** or **:log** to override the default.

:owner=

Function — Use this switch to change the owner (UIC) of the memory region.

Default — The UIC assigned when the NSM was created or last updated.

Syntax — Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC that they represent.

:pause

Function — Use this switch to stop the display after each screen of information. The display resumes when you press any key.

Default — The value specified by the OPTION Command.

Syntax — Type **:pause** or **:nopause** to override the default.

:protection=

Function — Use this switch to modify the protection·mask associated with a device, file, process, or memory region.

Default — The owner's protection mask.

Syntax — Type **:protection=** followed by a protection mask. Note that only those fields of the protection mask that are specified are altered. Fields not specified keep their original value.

For example, the switch :PROTECTION=S:,O:RWE allows the system no privilege, while the owner is granted read, write and execute privilege. The public and group privileges are unchanged.

Examples

```
> mstat
```

This command displays the information about all Named Shared Memory areas.

```
MYMEMORY
    Ref count :     0   Size:        32       Owner    : [0001,0001]
    Protection: S:  RE,P:     ,G:DWRE,O:DWRE  Username: SYSTEM
    Status    : linger          linked
```

The lines in the MSTAT display are described below:

Ref count	Number of processes using the NSM
Size	Size of the NSM in kilobytes
Owner	UIC of the process that defined the NSM
Protection	Protection mask for the NSM
Username	Username for the UIC of the NSM
Status	"Linger" means the NSM will remain defined even after the ref count goes to 0. If the NSM is not defined to linger, as soon as the ref count goes to 0 the NSM will be unlinked from the list of NSM regions and its memory will be deallocated "Linked" means the NSM is linked into the system list of NSM regions

Using Prompts

```
> mstat :protection=p:rw
> Region name    > mymemory
```

Prompts occur only if the :owner= or :protection= switches are specified and no memory name is given on the command line.

This command changes the protection mask of the NSM called MYMEMORY and produces the following display:

```
MYMEMORY
    Ref count :     0   Size:        32       Owner    : [0001,0001]
    Protection: S:  RE,P: WR ,G:DWRE,O:DWRE   Username: SYSTEM
    Status    : linger          linked
```

Related CIP Commands

None.

nsp

Functional Description

Use this command to perform supervisor calls for a remote process in a network.

Command Line Syntax

Mnemonic	nsp
Optional parameter	Directory Name

Parameters

Directory Name	Function	Use this parameter to specify the name of the directory where the netuaf.dat file is located.
	Default	sys$disk/syslib/
	Syntax	Type a directory name in the standard syntax. No wildcard symbols are allowed.

Switches

None.

Examples

```
> &nsp
```

This command creates a Network Surrogate Process. The NSP automatically mounts virtual circuit devices on the network as they are needed.

The first NSP to execute sets its process name to NSP_ANSWER, its execution priority to 2 and its timeslice to 100 milliseconds. Its port (as displayed by PSTAT) is set to the name of the virtual circuit the NSP is using. When a remote process calls the system, the NSP clones itself and changes its name to NSP_CONNECT. When it receives the first data packet from the remote process, the NSP changes its process name to NSP_XXXXXXXX, where XXXXXXXX is the process ID of the remote process. The NSP also changes its execution priority, timeslice, user ID code and privileges to those specified in the network user authorization file. The newly cloned NSP sets its process name to NSP_ANSWER and repeats the above process. The following PSTAT display shows NSPs with different names:

```
Time:     12-Feb-1985 09:51:02.93                          Free memory: 1504K
Up time:            11 18:59:03.95
  PID       Process Name          Port           Status     Size Pri Scheduled Times
--------  ----------------  ----------------  ----------   ---- --- --------- -----
00060002  Que_Manager       __MEPH_TT0        Waiting        40   2     16539   100
00060001  Logflush          __MEPH_TT0        Waiting         4   2    507147   100
00060EBD  NSP_answer        __MEPH_VC2        Waiting         8   2         2   100
00060EBC  NSP_connect       __MEPH_VC1        Waiting         8   2         4   100
00060DEE  vew_GB            __MEPH_TT9        Executing      56   7      2357    60
00060E94  NSP_000300BB      __MEPH_VC0        Waiting        24   7        40    60
00060C96  CIP_GB            __MEPH_TT9        ChildWait      16   7       313    60
00060DB4  CIP_NEAL          __MEPH_TT12       ChildWait      16   7       264    60
00060CA6  CIP_BACKUP        __MEPH_TT25       Waiting        16   8        11    60
00060EBB  asm_BQ0           __MEPH_NULL       Executing     228  13      2242    60
00060EA3  make_BQ0          __MEPH_NULL       ChildWait      20  13       223    60
00060DC9  MAKE_BQ0          __MEPH_NULL       ChildWait      44  13        25    60
00060DC8  CIP_MAKE          __MEPH_NULL       ChildWait      16  13         9    60
```

Using Prompts

None.

nstat

Functional Description

Use this command to display the current status of the network.

Command Line Syntax

Mnemonic	nstat			
Optional parameter	Siteid List			
Switches				
Node selection	:rnid=	:exclude=		
Display control	:full	:log	:header	:pause
Display content	:circuits	:clockrate	:fpoint	:memory
	:model	:physlink	:processes	:readseqno
	:rnid	:siteid	:systemname	:systemstatus
	:updateseqno	:verified	:writeseqno	

Parameters

Siteid List	Function	Optional. Use this parameter to specify which network nodes should be included in the status display.
	Default	All nodes that are currently accessible in the network are included in the status display.
	Syntax	Type a list of site IDs or nodenames separated by commas. Wildcard symbols are allowed.

Switches

:circuits — Function: Use this switch to display virtual circuit information for all local virtual circuits that pertain to the specified node. If the node has no open circuits, this field will not show up in the display.
Default: :nocircuits
Syntax: Type **:circuits**

:clockrate — Function: Use this switch to display the clock rate of the specified node as defined by the logical name sys$clkrate on that node.
Default: :noclockrate
Syntax: Type **:clockrate**

:exclude= — Function: Use this switch to exclude the specified nodes from the displays.
Default: All nodes in the network are displayed.
Syntax: Type **:exclude=** followed by a list of site IDs or nodenames separated by commas. Wildcard symbols are allowed.

:fpoint — Function: Use this switch to display the kind of floating point available on the specified node as defined by the logical name sys$math on that node.
Default: :nofpoint
Syntax: Type **:fpoint**

:full — Function: Use this switch to display all information about the specified nodes.
Default: :nofull
Syntax: Type **:full**

:header — Function: Use this switch to display column headers.
Default: :header
Syntax: Type **:noheader**

:log — Function: Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:memory	Function	Use this switch to display the amount of free memory on the specified node.
	Default	:nomemory
	Syntax	Type **:memory**
:model	Function	Use this switch to display the model of the specified node as defined by the logical name wicat$model on that node.
	Default	:nomodel
	Syntax	Type **:model**
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes after the user presses any key.
	Default	The value specified by the OPTION command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:physlink	Function	Use this switch to display the name of the physical network device which connects the local node to the remote node.
	Default	:nophyslink
	Syntax	Type **:physlink**
:processes	Function	Use this switch to display the number of processes executing on the specified node.
	Default	:noprocesses
	Syntax	Type **:processes**
:readseqno	Function	Use this switch to display the network Read Sequence Number from the local node's tables pertaining to the remote node. This number is the next packet number that the local node expects to receive from the remote node.
	Default	:noreadseqno
	Syntax	Type **:readseqno**
:rnid	Function	Use this switch to display the Remote Network ID (RNID) of the node. The RNID is a 32-bit unsigned longword that can be specified by the

network manager using the NSYSPROF utility. It provides a means of logically grouping different network nodes together.

Default :nornid

Syntax Type **:rnid**

:rnid=

Function Use this switch to select only those network nodes that have the sepecified Remote Network ID (RNID).

Default All nodes are selected.

Syntax Type **:rnid=** followed by a list of hexadecimal RNID numbers separated by commas.

For example, **:rnid=1,ffff0123** will cause NSTAT to select nodes that have an RNID of either $00000001 or $FFFF0123. The RNID for a node is specified in NSYSPROF.

NOTE: You may use WMCS range wildcarding with this switch. For example, **:rnid=0-10,15** will select all RNIDS in the range of 0..10, as well as RNID 15.

:siteid

Function Use this switch to display the site ID of the node.

Default :nositeid

Syntax Type **:siteid**

:systemname

Function Use this switch to display the system name of the specified node as defined by the logical name sys$sysname on that node.

Default :nosystemname

Syntax Type **:systemname**

:systemstatus

Function Use this switch to print a "system status" line at the top of the output display. This status line contains the nodename of the local node, the total number of nodes accessible in the network, the system time, and the number of network buffers on the local node that are not in use.

Default :nosystemstatus

Syntax Type **:systemstatus**

:updateseqno	Function	Use this switch to display the network Update Sequence Number from the local node's tables pertaining to the specified remote node. This number is the number of the last update packet that was received from the specifed node.
	Default	:noupdateseqno
	Syntax	Type **:updateseqno**
:verified	Function	Use this switch to determine whether or not the local node has performed site ID verification with the specified node.
	Default	:noverified
	Syntax	Type **:verified**
:writeseqno	Function	Use this switch to display the network Write Sequence Number from the local node's tables pertaining to the specified remote node. This number is the packet number of the next packet that was the local node will write to the specified node.
	Default	:nowriteseqno
	Syntax	Type **:writeseqno**

Examples

```
> nstat
```

This command generates the following kind of display:

```
     Nodename
     --------
     __NODEA
     __NODEB
     __NODEC
```

```
> nstat :systemstatus :header :siteid :systemname :model :clockrate
       :proc :mem
```

This command generates the following kind of display:

```
    Current Node : __NODEA             Total Nodes       : 6
    Time: 19-Apr-1985 16:49:33.60      Available Buffers : 15
    Nodename  Site System Name    Model Clock Proc  Mem
    --------- ---- -------------- ----- ----- ----  -----
    __NODEA   1158 Joes_system      160   8.0    7   392
    __NODEB   0159 Your_system      156   8.0    6  1172
    __NODEC   2981 System_abc       155   8.0    8   560
    __NODED   1039 System_xyz      2220  12.5    6  1480
    __NODEF   1159 My_system        200   8.0    4    36
    __NODEG   1040 Frank           2220  12.5   20  2736
```

The :systemstatus switch prints the first two lines in the display, which contain the current nodename, the current system time, the total nodes on the network, and the total number of available network buffers.

The :header switch prints the column headers in the display.

Tne other switches cause the remaining columns to be printed in the display. These columns appear in the order that the switches are typed on the command line.

```
> nstat __noded :full
```

This command generates the following kind of display:

```
__NODED
   Siteid         : 1039               Floating Point     : SOFTWARE
   RNID           : 00000008           Physical Link      : _EN0
   System Name    : System_xyz         Verified           : Yes
   Model          : System 2220        Read Seq Number    : 01E3
   Clockrate      : 12.5 mhz           Write Seq Number   : 010B
   Free Memory    : 1345               Update Seq Number  : 1457
   Num Procs      : 27
   Virtual circuit status:
       Local   Circuit  Local     Remote Remote  Remote
       Circuit Status   Buffers   Siteid Circuit Buffers
       ------- -------- -------   ------ ------- -------
             0 Opened         6    ----    ---     ---
             1 Dialed         6    0159     1       6
             2 Dialed         6    1040     2       6
```

What each field means:

Siteid 16-bit hexadecimal site ID of __NODED

RNID 32-bit hexadecimal Remote Network ID of __NODED

System Name Translation of SYS$SYSNAME on __NODED

Model — Translation of WICAT$MODEL on __NODED

Clockrate — Translation of SYS$CLKRATE on __NODED

Free Memory — Amount of free memory on __NODED in 1-Kbyte chunks

Num Processes — Number of processes currently executing on __NODED

Floating Point — Translation of SYS$MATH on __NODED

Physical Link — Physical network device that connects local node to __NODED

Verified — Indicates whether local node has already performed site ID verification with __NODED

Read Seq Number — Next sequence number local node expects to receive from __NODED

Write Seq Number — Next sequence number local node will write to __NODED

Update Seq Number — Sequence number of last update packet received from __NODED

Virtual Circuit Status — Status of all network virtual circuits currently associated with __NODED

Local Circuit is the circuit number of the circuit on the local node.

Circuit Status is the status of the local circuit. It can be one of three values:

Opened — The circuit is open, but not currently connected to any remote circuits. This value will only show up on the display for the local node.

Dialed — The circuit is connected to a remote node, and the local node initiated the connection by doing a DIAL.

Answered — The circuit is connected to a remote node, and the local node answered the call.

Local Buffers is the number of buffers that are reserved for use by this local circuit on the local node.

Remote Siteid is the site ID of the remote node to which this local circuit is connected.

Remote Circuit is the circuit number of the circuit on the remote node to which the local circuit is connected.

Remote Buffers is the number of buffers that are reserved for use by the remote circuit on the remote node.

Using Prompts

None.

Notes on Usage

Unless overriden with the percent character, %, the site IDs specified in the Siteid List parameter will be assumed to be a hexadecimal number. Site IDs and nodenames may be used interchangeably in this parameter. For example, if this parameter is specified as __GEORGE,8 then NSTAT would display status only for node GEORGE and for site ID 8. The two leading underscores on the nodename are optional.

The nodenames in the Siteid List can use standard WMCS wildcarding. For example, n* indicates all nodes that start with the letter N.

Some of the display items cannot be displayed without having to actually execute system calls on the remote node in order to acquire the necessary information. These display items are: System Name, Model, Clockrate, Processes, Memory, and Fpoint. In order to execute these remote system calls the user must have the necessary privileges on both the local node and on the remote node. If certain information cannot be gathered from some remote nodes, those fields in the display will be replaced by a field of asterisks.

NOTE: Some of the display items take longer to display than others because it takes time to gather remote information over the network.

If site ID verification is enabled on the network, and information about remote nodes is desired which requires executing remote system calls on those nodes, then the site ID verification will take place on the first system call executed on each unverified remote node. Thus, on a large network with site ID verification enabled, the first "nstat :full" command will take approximately 10 seconds for each remote node that has not already verified.

NSTAT, like PSTAT, is a "snapshot" of the network at a given instant in time. Whatever is shown for that instant may not have been true an instant before and probably will not be true an instant later.

Related CIP Commands

nsysprof	Manage the system network profile

nsysprof

Functional Description

Use this command to set up the system network configuration. NSYSPROF is also used to maintain and distribute public keys for network site ID verification.

Command Line Syntax

Mnemonic	nsysprof		
Optional parameter	Directory		
Switches	:model=	:setupin=	:setupout=

Parameters

Directory

Function: Optional. Use this parameter to specify the directory to contain the NETCONFIG.nnn and NODCONFIG.nnn files.

Default: sys$disk/rootdir/

Syntax: Type a full directory pathname. Wildcard symbols are not allowed.

NOTE: In all examples, "nnn" is the system model number, e.g., NETCONFIG.156.

Switches

:model=

Function
: Use this switch to specify which class of computer hardware the NETCONFIG.nnn and NODCONFIG.nnn files are for.

Default
: The value of the logical name SYS$MODEL on the computer on which the files reside.

Syntax
: Type **:model=** followed by one of the following: **100, 150, 155, 156, 160, 200, 220, 2220,** or **300.**

NOTE: WICAT system models 100, 200, 220, 2220, and 300 are converted to system model number 100. WICAT models 150, 155, 156 and 160 are converted to system model number 156. If you do not specify a value for this switch, NSYSPROF assumes that the extension (100, 156, etc.) of the netconfig and nodconfig files is the same as that of the computer where the files reside. The :model= switch can override the extension with a specific model regardless of the model of the computer.

:setupin=

Function
: Use this switch to specify an alternate setup file to use for sys$input. If you specify a directory path, NSYSPROF looks in that directory for the setup file based on the terminal type. If you specify an explicit file, NSYSPROF uses that file.

Default
: The setup file in sys$disk/syslib.setup/ determined by the terminal type is used, e.g., if your terminal is a T7000, then the file used will be sys$disk/syslib.setup/setup252.sys.

Syntax
: Type **:setupin=** followed by a filename.

:setupout=	Function	Use this switch to specify an alternate setup file to use for sys$output. If you specify a directory path, NSYSPROF looks in that directory for the setup file based on the terminal type. If you specify an explicit file, it will use that file.
	Default	The setup file in sys$disk/syslib.setup/ determined by the terminal type is used, e.g., if your terminal is a T7000, then the file used will be sys$disk/syslib.setup/setup252.sys.
	Syntax	Type **:setupout=** followed by a filename.

Examples

> **nsysprof**

This command generates the following kind of display:

```
                     Local System Network Configuration
------------------------------------------------------------------------------
Local Node Name          : BOB                        No private key specified
Local Site ID            : 0003
Local Remote Network ID : 00000001

Minimum Receive Cache Buffers  : 10       Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers  : 60       System Overhead Cache Size :  5
Default Receive Window Size    : 3
                                          Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit : 0
Last SVC Call Virtual Circuit  : 19


                    Remote System Network Configuration
-----------------------------------------------------------------------------
Remote Site ID : 0001  __GEORGE                        No public key specified

Encryption     : NODESENCRYPT    FASTENCRYPT
```

The top portion of the screen deals with the local network configuration. It contains the information in the NETCONFIG.nnn.1 file. This display remains visible throughout the execution of NSYSPROF.

The bottom portion of the screen displays one record from the NODCONFIG.nnn.1 file. You can page forward and backward through the records of this file with escape-key and control-key functions.

Using Prompts

None.

Notes on Usage

NSYSPROF should only be used by the system manager.

NSYSPROF maintains two files, both in /ROOTDIR/ on the system disk. The first file is NETCONFIG.nnn.l. It contains information about the local system network configuration information such as site ID, nodename, and private key. It also contains several parameters used to fine-tune system networking performance. The second file is NODCONFIG.nnn.l. It contains the public keys of all the systems on the network with which site ID verification must be done. All computers on the network must have the public keys of all computers they need to verify with. There are also some flags specifying whether encryption of user data should be performed when transmitting to another system. These flags can be specified on a node-by-node basis.

Detailed information on configuring the network is presented under these headings:

The NSYSPROF Display
Functions Available in NSYSPROF
Field Characteristics
Cursor Movement and Control-Key Functions

The NSYSPROF Display

```
                    Local System Network Configuration
------------------------------------------------------------------------------
Local Node Name          : BOB                        No private key specified
Local Site ID            : 0003
Local Remote Network ID  : 00000001

Minimum Receive Cache Buffers  : 10    Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers  : 60    System Overhead Cache Size :  5
Default Receive Window Size    : 3
                                       Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit : 0
Last SVC Call Virtual Circuit  : 19


                   Remote System Network Configuration
------------------------------------------------------------------------------
Remote Site ID : 0001  __GEORGE                         No public key specified

Encryption     : NODESENCRYPT    FASTENCRYPT
```

The following is a description of each field in the NSYSPROF display:

Local Node Name and Private Key Fields

The Local Node Name field contains the name assigned to the local computer during the boot process. This name can be any alphanumeric (plus tilde and dollar sign) string. NSYSPROF will not let you exit the program until a valid nodename has been specified.

The Private Key field indicates whether a public/private key pair has been generated for the local computer. Public/private keys are not required. If there are no keys, no site ID verification is performed.

```
                    Local System Network Configuration
--------------------------------------------------------------------------------
Local Node Name          : BOB                          No private key specified
Local Site ID            : 0003
Local Remote Network ID  : 00000001

Minimum Receive Cache Buffers   : 10    Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers   : 60    System Overhead Cache Size :  5
Default Receive Window Size     : 3
                                        Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit  : 0
Last SVC Call Virtual Circuit   : 19


                   Remote System Network Configuration
--------------------------------------------------------------------------------
Remote Site ID : 0001   __GEORGE                         No public key specified

Encryption     : NODESENCRYPT   FASTENCRYPT
```

Local Site ID Field

This field contains the site ID assigned to the local computer during the boot process. NSYSPROF will not let you exit the program until a valid site ID has been given.

```
                    Local System Network Configuration
--------------------------------------------------------------------------------
Local Node Name          : BOB                          No private key specified
Local Site ID            : 0003
Local Remote Network ID  : 00000001

Minimum Receive Cache Buffers   : 10    Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers   : 60    System Overhead Cache Size :  5
Default Receive Window Size     : 3
                                        Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit  : 0
Last SVC Call Virtual Circuit   : 19


                   Remote System Network Configuration
--------------------------------------------------------------------------------
Remote Site ID : 0001   __GEORGE                         No public key specified

Encryption     : NODESENCRYPT   FASTENCRYPT
```

Local Remote Network ID

This field contains the remote network ID (RNID). Systems with the same RNID number form a logical group. The system manager can use RNIDs to logically separate computers in one network location from those in a different location. Should two remote networks be joined via some gateway connection, the RNID could be used to tell which computers belong to which remote network. The system manager should choose a number for the remote network ID.

```
                    Local System Network Configuration
-------------------------------------------------------------------------------
Local Node Name          : BOB                         No private key specified
Local Site ID            : 0003
Local Remote Network ID : 00000001

Minimum Receive Cache Buffers  : 10       Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers  : 60       System Overhead Cache Size :  5
Default Receive Window Size    : 3
                                          Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit : 0
Last SVC Call Virtual Circuit  : 19


                   Remote System Network Configuration
------------------------------------------------------------------------------
Remote Site ID : 0001    __GEORGE                        No public key specified

Encryption     : NODESENCRYPT    FASTENCRYPT
```

Minimum Receive Cache Buffers and Transmit Buffer Cache Size Fields

The Minimum Receive Cache Buffers field defines the number of network receive buffers which will be allocated when the first physical network device is mounted. The system manager should set this to the Default Receive Window Size times the average number of virtual circuits that he expects to use, because each active virtual circuit needs to have a window-size number of buffers. The default for this field is 10, the minimum allowed value is 3, and the maximum allowed value is 1000. The size of a buffer is 1500 bytes. With overhead, five buffers occupy two 4-Kbyte pages, or 8-Kbytes of memory.

The Transmit Buffer Cache Size field contains the number of buffers allocated for data transmission. Transmit buffers are not allocated on a per virtual circuit basis as receive buffers are. This is because data can be put into them and written out. As soon as they are written, the transmit buffers are free for some other process to use. Receive buffers, in contrast, are tied up until a process reads the data from them. The system manager can experiment with different values for this field. With high speed communication devices such as Ethernet, good throughput should be achievable with only 10 transmit buffers. The transmit buffer pool is fixed. It does not grow like the receive buffer pool can. The default for this field is 10, the minimum allowed value is 3, and the maximum allowed value is 1000.

```
                     Local System Network Configuration
---------------------------------------------------------------------------
Local Node Name           : BOB                    No private key specified
Local Site ID             : 0003
Local Remote Network ID : 00000001

Minimum Receive Cache Buffers   : 10    Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers   : 60    System Overhead Cache Size :  5
Default Receive Window Size     : 3
                                        Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit : 0
Last SVC Call Virtual Circuit  : 19


                    Remote System Network Configuration
---------------------------------------------------------------------------
Remote Site ID : 0001  __GEORGE                     No public key specified

Encryption      : NODESENCRYPT    FASTENCRYPT
```

Maximum Receive Cache Buffers and System Overhead Cache Size Fields

The Maximum Receive Cache Buffers field defines the maximum number of network receive buffers that will ever be allocated. If this number is greater than the Minimum Receive Cache Buffers field, the network software will automatically allocate the number of buffers it needs during peak usage periods, not to exceed the value specified in this field. This value should be set to something that will not allow the network buffers to use all available memory. If not enough network buffers are available to complete a call, you will get a "no network buffers" error when an attempt is made to establish a remote connection. Note that once the network allocates buffers, it will not give them back unless ALL network devices are dismounted. If the Default Receive Window Size is set to 3 and the system must support 20 active circuits, the Maximum Receive Cache Buffers should be set to at least 60 (3 X 20). The default for this field is 60. The minimum allowed value is whatever the Minimum Receive Cache Buffers is set to, and the maximum allowed value is 1000.

The System Overhead Cache Size field contains the number of buffers allocated for network system overhead. These buffers are used for handling network dynamic re-routing. Five buffers should be sufficient, although on systems where there is not much memory, one or two buffers would probably not cause any noticeable degradation in network performance. The main use for the system overhead buffers is for gateway traffic. Data will be temporarily kept in these buffers until they can be forwarded to another physical network link. For gateway computers (especially where joining a fast device such as Ethernet to a slow device such as RS232) this overhead cache size should be set to a fairly large number, depending on the amount of memory available. The default for this field is 5, the minimum allowed value is 3, and the maximum allowed value is 1000.

```
                     Local System Network Configuration
-----------------------------------------------------------------------------
Local Node Name         : BOB                        No private key specified
Local Site ID           : 0003
Local Remote Network ID : 00000001

Minimum Receive Cache Buffers  : 10     Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers  : 60     System Overhead Cache Size :  5
Default Receive Window Size    : 3
                                        Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit : 0
Last SVC Call Virtual Circuit  : 19


                     Remote System Network Configuration
-----------------------------------------------------------------------------
Remote Site ID : 0001   __GEORGE                      No public key specified

Encryption     : NODESENCRYPT    FASTENCRYPT
```

<u>Default Receive Window Size Field</u>

This field defines how many receive buffers will be used by each active circuit. A value in the range of 2 to 5 is suggested here. If this value is set to 1, each circuit can only use 1 buffer. Thus, if a data packet is written to a remote system, the remote process must read the packet before the local process can write another data packet. One window per circuit will greatly diminish throughput, but it does require less memory. If the window size is 3, the local process can write 3 packets before the remote system needs to read any packets. Then as the remote system reads packets, the local process is allowed to write more packets. Optimal throughput can be achieved by setting the window size to at least 3. A window size of 5 or greater is probably wasting memory. The default for this field is 3, the minimum allowed value is 1, and the maximum allowed value is 255.

```
                        Local System Network Configuration
--------------------------------------------------------------------------------
Local Node Name         : BOB                              No private key specified
Local Site ID           : 0003
Local Remote Network ID : 00000001

Minimum Receive Cache Buffers  : 10     Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers  : 60     System Overhead Cache Size :  5
Default Receive Window Size    : 3
                                        Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit : 0
Last SVC Call Virtual Circuit  : 19

                        Remote System Network Configuration
--------------------------------------------------------------------------------
Remote Site ID : 0001   __GEORGE                          No public key specified

Encryption      : NODESENCRYPT    FASTENCRYPT
```

Maximum Calls Waiting Field

This field controls how many unanswered "dial requests" will be queued up on a system when no process is attempting to answer them. A dial request will be immediately rejected if this is set to 0 and there is no process waiting to answer a request. If this field is set to 1, only one dial request will be queued. If there is a process answering dial requests, the one dial will be answered. If, in the meantime, another dial request arrives before the first one is answered, the first request will be rejected and the new request will be queued. The default for this field is 10, which means that the 10 most recent unanswered dial requests will be queued to be answered at a later time. It is advisable to not set this below 1 or 2, since it is possible to lose calls. If only one process is answering calls (such as NSP), there are times where more than one call can come in at the same time and the NSP would answer the first request. Before the NSP could answer the second request, it would be rejected. The default for this field is 10, the minimum allowed value is 0, and the maximum allowed value is 1000.

```
                    Local System Network Configuration
------------------------------------------------------------------------------
Local Node Name          : BOB                    No private key specified
Local Site ID            : 0003
Local Remote Network ID : 00000001

Minimum Receive Cache Buffers  : 10     Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers  : 60     System Overhead Cache Size :  5
Default Receive Window Size    : 3
                                        Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit : 0
Last SVC Call Virtual Circuit  : 19


                   Remote System Network Configuration
------------------------------------------------------------------------------
Remote Site ID : 0001    __GFORGE                    No public key specified

Encryption     : NODESENCRYPT    FASTENCRYPT
```

<u>First and Last SVC Call Virtual Circuit Fields</u>

These two fields specify a range of virtual circuit numbers that are to be automatically used by WMCS for creating supervisor call (SVC) connections. If the first field is set to 11 and the last field is set to 23, the WMCS will automatically use virtual circuits _VC11 through _VC23. It will never use any more than these 13 circuits. All other circuits are reserved for user calls. These virtual circuits are treated as mounted devices. These two fields provide a way of putting an absolute limit on the number of virtual circuits that the WMCS will use. The default for the first field is 0, the minimum allowed value is 0, and the maximum allowed value is 255. The default for the second field is 19, the minimum allowed value is whatever the First SVC Call Virtual Circuit is set to, and the maximum allowed value is 255.

```
                    Local System Network Configuration
--------------------------------------------------------------------------
Local Node Name          : BOB                          No private key specified
Local Site ID            : 0003
Local Remote Network ID  : 00000001

Minimum Receive Cache Buffers  : 10     Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers  : 60     System Overhead Cache Size :  5
Default Receive Window Size    : 3
                                        Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit : 0
Last SVC Call Virtual Circuit  : 19


                   Remote System Network Configuration
--------------------------------------------------------------------------
Remote Site ID : 0001   __GEORGE                      No public key specified

Encryption     : NODESENCRYPT    FASTENCRYPT
```

Remote Site ID and Public Key Fields

The Remote Site ID field contains the site ID of the remote computer for which this record exists in the NODCONFIG file. If this site ID is currently accessible on the network, the nodename will be displayed next to the Remote Site ID.

The Public Key field indicates whether or not there is a public key for the remote host. If the remote host has defined a public/private key pair for itself, then you need to have a public key defined for it in your NODCONFIG file. Otherwise, you will not be able to talk to this remote host over the network. To get public keys from other nodes, you must merge your NODCONFIG.nnn.1 file with that of the other computer. (See the UK and UA functions described in the next section.) If the system manager knows that another computer node has some public key information that he wants on the local computer, he can use the UK function to get the keys. The NODCONFIG.nnn.1 file can be passed around from computer to computer back-up media and updated from the media if necessary.

```
                    Local System Network Configuration
---------------------------------------------------------------------------
Local Node Name          : BOB                        No private key specified
Local Site ID            : 0003
Local Remote Network ID  : 00000001

Minimum Receive Cache Buffers  : 10    Transmit Buffer Cache Size : 10
Maximum Receive Cache Buffers  : 60    System Overhead Cache Size :  5
Default Receive Window Size    : 3
                                       Maximum Calls Waiting      : 10
First SVC Call Virtual Circuit : 0
Last SVC Call Virtual Circuit  : 19
```

```
                    Remote System Network Configuration
---------------------------------------------------------------------------
Remote Site ID : 0001   __GEORGE                     No public key specified

Encryption     : NODESENCRYPT   FASTENCRYPT
```

Encryption Field

There are two boolean toggle values in this field. These two fields specify whether or not some default data encryption is to be performed between the local host and this remote host, whenever the local calls the remote. The kinds of encryption that can be performed are DES encryption (which uses the standard DES encryption algorithm) and FAST encryption. The fast encryption uses a WICAT internal algorithm which is about 5 times as fast as DES, but is not as secure. DES encryption can do approximately 1 Kbyte per second on a WICAT System 2220 with no other jobs running, and it can do approximately 1/2 Kbytes per second on a WICAT System 150 with no other jobs running. Both encryption methods make the network noticeably slower and need only be used if there is danger of sensitive information being monitored by some unauthorized users.

Functions Available in NSYSPROF

Each of the following functions is executed by first striking [ESC][ESC], then typing the mnemonic on the NSYSPROF function line that appears at the bottom of the screen. The position of the cursor when you strike [ESC][ESC] does not affect the execution of any of these functions.

ab

Abort this session of NSYSPROF.

FUNCTIONAL DESCRIPTION

AB aborts all changes made during an editing session. The NETCONFIG and NODCONFIG files are the same as before the editing session began.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

ex	Exit NSYSPROF
sa	Save modifications made so far

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type ab on the NSYSPROF function line.

If the NETCONFIG and NODCONFIG information has not been modified, the screen is cleared, and you return to your CIP prompt. If modifications have been made, this prompt appears on the function line:

```
NETCONFIG and/or NODCONFIG files have been modified - abort?
```

Step 3 Type y or n. If you type some other character, the following message briefly appears on your screen:

```
Please type 'Y' or 'N'.
```

If you type Y, the screen is cleared and you return to your CIP prompt. If you type N, the cursor returns to the position it had on the screen before executing AB.

ad

Add a new node record.

FUNCTIONAL DESCRIPTION

AD creates a new node record, assigns it the site ID you specify, and adds it to the list of node records.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

da	Delete a node record from all on-line nodes
dl	Delete a node record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **ad** on the NSYSPROF function line. This prompt appears on the function line:

```
Add Remote Site ID>
```

Step 3 Type the site ID to be assigned to the node. The site ID must be unique to the NODCONFIG file and cannot be longer than four hexadecimal digits.

Step 4 Strike [RETRN]. The new node record appears in the bottom half of the NSYSPROF display. If the specified site ID is currently accessible on the network, the nodename for that site will be displayed to the right of the site ID.

If no site ID is specified, or if an invalid hexadecimal number is entered, the following message is displayed on the function line:

```
The site id must be a hexadecimal number.
```

cn

Change nodename.

FUNCTIONAL DESCRIPTION

CN changes the nodename for your local node in your NETCONFIG file.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

cs	Change site ID

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type cn on the NSYSPROF function line. This prompt appears on the function line:

```
Change Local Node Name to>
```

Step 3 Type the new nodename to be assigned to your node. The nodename must be unique on the network and cannot be longer than 16 characters. The name must consist of A...Z, a...z, 0...9, ~, or $. No underscores are allowed.

Step 4 Strike [RETRN]. The new nodename record appears on the third line of the NSYSPROF display.

If no nodename is specified, or if an invalid character is entered, the following message appears on the function line:

```
The local nodename may be composed only of A-Z, a-z, 0-9, $ and ~.
```

cp

Spawn a CIP while still in NSYSPROF.

FUNCTIONAL DESCRIPTION

CP allows you to execute a CIP while staying in NSYSPROF.

CORRESPONDING CONTROL-KEY FUNCTIONS

[ESC] c

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **cp** on the NSYSPROF function line. Your CIP prompt appears.

When you log off from the CIP, you are returned to NSYSPROF.

cs

Change site ID.

FUNCTIONAL DESCRIPTION

CS changes your local site ID in your NETCONFIG file.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

cn	Change nodename

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type cs on the NSYSPROF function line. This prompt appears on the function line:

```
Change Local Site ID to>
```

Step 3 Type the new site ID to be assigned to the node.

Step 4 Strike [RETRN]. The new site ID appears on the fourth line of the NSYSPROF display.

If no site ID is specified, or if an invalid hexadecimal number is entered, the following message appears on the function line:

```
The site id must be a hexadecimal number.
```

da

Delete a node record from all on-line nodes.

FUNCTIONAL DESCRIPTION

DA deletes the node record displayed at the bottom of the screen from all nodes currently accessible on the network.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

ad	Add a node record
dl	Delete a node record
ua	Update all keys
uk	Update puolic keys

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **da** on the NSYSPROF function line. This prompt appears on the function line:

```
Delete all node records of remote site id NNNN?
```

Step 3 Type y if you want to delete the node records or n if you do not want to delete them. If characters other than Y or N are typed, the following message is displayed on the function line:

```
Please type 'Y' or 'N'.
```

dk

Delete keys.

FUNCTIONAL DESCRIPTION

DK deletes the public/private key pair for the local computer.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

nk	Generate new public/private keys

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **dk** on the NSYSPROF function line. This prompt appears on the function line:

```
Delete Local System Public/Private Keys?
```

Step 3 Type y if you want to delete public/private key pair or n if you do not want to delete it. If characters other than Y or N are typed, the following message is displayed on the function line:

```
Please type 'Y' or 'N'.
```

If no public/private key pair has been generated for the local computer, the following message is displayed:

```
There is no local system private key to delete.
```

dl

Delete a node record.

FUNCTIONAL DESCRIPTION

DL deletes the node record displayed in the bottom half of the screen.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

da	Delete a node record from all on-line nodes
ad	Add a new node record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type dl on the NSYSPROF function line. This prompt appears on the function line:

```
Delete record?
```

Step 3 Type y if you want to delete the current record or n if you do not want to delete it. If characters other than Y or N are typed, the following message is displayed on the function line:

```
Please type 'Y' or 'N'.
```

ex

Exit NSYSPROF

FUNCTIONAL DESCRIPTION

EX terminates the editing session, saves any changes made during the editing session, and returns you to the CIP.

CORRESPONDING CONTROL-KEY FUNCTIONS

{-}

RELATED ESCAPE-KEY FUNCTIONS

ab	Abort NSYSPROF
sa	Save modifications made so far

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type ex on the NSYSPROF function line. The following message appears at the bottom of the screen:

```
Exiting...
```

The screen is cleared and the cursor reappears next to your CIP prompt.

he

Call to the screen a display of NSYSPROF functions and features.

FUNCTIONAL DESCRIPTION

HE displays a list of functions available in NSYSPROF.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **he** on the NSYSPROF function line. The first page (or screenful) of a three-page help display appears on your screen.

The first page contains a list of the escape-key functions available.

The second page lists functions for deletion and cursor movement.

The third page describes the different field types.

Step 3 Strike any key to cycle through the pages and return to the NSYSPROF display.

Type [CTRL] c at any time to return to the NSYSPROF display without showing the remaining help pages.

nk

Generate new public/private keys.

FUNCTIONAL DESCRIPTION

NK generates a new public/private key pair by forking the KEYGEN program. The KEYGEN program can take anywhere from 1 to 45 minutes to generate the keys. When KEYGEN finishes it stores the new public/private keys in the proper files. It will not be able to store them while the NSYSPROF program is executing. It will keep trying until it can successfully save the new keys.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

dk	Delete keys

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **nk** on the NSYSPROF function line. The following message appears temporarily on the function line:

```
Public/Private Key Generation Started...
```

pr

Create a text-file copy of NSYSPROF records.

FUNCTIONAL DESCRIPTION

PR creates two files, NETCONFIG.PRN and NODCONFIG.PRN, in the current directory.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **pr** on the NSYSPROF function line. This report appears momentarily on the screen:

```
Printing devicename/directory/NETCONFIG.nnn.1 records...
```

During the display of this message, the NETCONFIG.nnn record is written to the file NETCONFIG.PRN. This report then appears on the screen:

```
Printing devicename/directory/NODCONFIG.nnn.1 records...
```

During the display of this message, the NODCONFIG.nnn records are written to the file NODCONFIG.PRN. After the records are written, the message is erased from the screen and the cursor is returned to its original position on the screen.

rp

Repeat last function.

FUNCTIONAL DESCRIPTION

RP repeats the last function you executed.

CORRESPONDING CONTROL-KEY FUNCTIONS

[CTRL] z

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1	Strike [ESC][ESC]
Step 2	Type **rp** on the NSYSPROF function line. The last function is repeated.

rs

Refresh the screen.

FUNCTIONAL DESCRIPTION

RS restores the display on the screen. This is useful when a message or other interruption disrupts the display.

CORRESPONDING CONTROL-KEY FUNCTIONS

[CTRL] _

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1	Strike [ESC][ESC]
Step 2	Type **rs** on the NSYSPROF function line. The screen is restored to its last display.

sa

Save the modifications made so far in the NETCONFIG and NODCONFIG files.

FUNCTIONAL DESCRIPTION

SA writes out all modified records to the NETCONFIG and NODCONFIG files without exiting NSYSPROF. This is useful when you have a large number of modifications and want to protect them from system failure.

CORRESPONDING CONTROL-KEY FUNCTIONS

{,}

RELATED ESCAPE-KEY FUNCTIONS

ex	Exit NSYSPROF
ab	Abort the editing session

EXECUTION

Step 1 | Strike [ESC][ESC]

Step 2 | Type **sa** on the NSYSPROF function line. This prompt appears on the function line:

```
Saving modifications...
```

After writing all the modified records to the NETCONFIG and NODCONFIG files, the cursor returns to the position it had prior to the SA function.

sf

Show the first node record.

FUNCTIONAL DESCRIPTION

SF allows you to show the first node record in the NODCONFIG.nnn file.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

sh	Show a specific node record
sl	Show the last node record
sr	Search for a node record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **sf** on the NSYSPROF function line. This message appears momentarily on the NSYSPROF function line:

```
     Showing the first record . . .
```

The first node record is displayed on the screen.

sh

Show a specific node record.

FUNCTIONAL DESCRIPTION

SH allows you to specify which node record you want displayed on the screen.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

Related ESCAPE-KEY Functions

sf	Show the first node record
sl	Show the last node record
sr	Search for a node record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **sh** on the NSYSPROF function line. This prompt appears on the NSYSPROF function line:

```
Show Remote Site ID>
```

Step 3 Type the site ID of a node.

Step 4 Strike [RETRN]. If the specified site ID has a record, that node record is displayed on the screen. Otherwise, this message appears on the screen:

```
The site id does not exist. Please enter another site id.
```

sl

Show the last node record.

FUNCTIONAL DESCRIPTION

SL allows you to show the last node record in the NODCONFIG.nnn file.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

Related ESCAPE-KEY Functions

sf	Show the first node record
sh	Show a specific node record
sr	Search for a node record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type sl on the NSYSPROF function line. This message appears momentarily on the function line:

```
Showing the last record . . .
```

The last node record is displayed on the screen.

sr

Search for and display a node record.

FUNCTIONAL DESCRIPTION

SR allows you to give a site ID of a node and display that node record on the screen. If the given site ID does not have a record, the record of the node whose site ID follows the given site ID is displayed.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

Related ESCAPE-KEY Functions

sf	Show the first node record
sh	Show a specific node record
sl	Show the last node record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **sr** on the NSYSPROF function line. This prompt appears on the function line:

```
Search for Remote Site ID>
```

Step 3 Type the site ID of a node.

Step 4 Strike [RETRN]. A node record is displayed on the screen.

ua

Update all node records on all on-line nodes.

FUNCTIONAL DESCRIPTION

UA updates the public keys in the NODCONFIG.nnn file on every system on the network.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

uk	Update keys from another NODCONFIG.nnn file

EXECUTION

Step 1 | Strike [ESC][ESC]

Step 2 | Type **ua** on the NSYSPROF function line.

NSYSPROF then cycles through every NODCONFIG.nnn file to which it has access on the network. It goes through all nodes once in order to update the local NODCONFIG file from all other nodes, and then goes through all nodes a second time to spread the updated information back to all other accessible nodes. The following messages appear on the function line as each node is updated:

```
Reading __NODE_DEVICE/ROOTDIR/NODCONFIG.NNN

Updating public keys with __NODE_DEVICE/ROOTDIR/NODCONFIG.NNN
```

Some nodes may be inaccessible to the local NSYSPROF, in which case appropriate diagnostic messages appear momentarily, before NSYSPROF continues with the next node.

Undo changes made on a record.

FUNCTIONAL DESCRIPTION

UD allows you to undo changes made to a record. If the cursor is in a field in the upper half of the form, i.e., in the NETCONFIG record, any changes made to the NETCONFIG record are undone. If the cursor is in a field in the lower half of the form, i.e., in a NODCONFIG record, any changes made to the NODCONFIG record are undone. Changes made to the NETCONFIG record can be undone at any time. Changes made to a NODCONFIG record can be undone only while the record is displayed on the screen. Any function on the first help display page — except CP, HE, and RS — makes the changes permanent (unless NSYSPROF is aborted with the AB function).

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 | Strike [ESC][ESC]

Step 2 | Type **ud** on the NSYSPROF function line. This message appears momentarily on the function line:

    Undoing changes...

The node record is displayed on the screen as it was before the changes were made.

uk

Update public keys.

FUNCTIONAL DESCRIPTION

Once a public/private key pair has been generated and saved, UK updates a specific NODCONFIG.nnn file, which exchanges all new public keys between the two files. The UK function prompts for a directory name and a system model number. The directory name is the directory where the second NODCONFIG.nnn.1 file exists, and the model number is the model number of the computer for which the NODCONFIG.nnn.1 file was built. The function of the update commands is to merge the NODCONFIG.nnn files together, with each getting the most recently generated public keys. (A key sequence number is maintained internally, so that NSYSPROF knows which keys are the most recent.) After generating a new local key pair, the system manager can do all of the updates over the network. If this procedure is not followed, the system manager may have to copy the NODCONFIG.nnn.1 file onto a floppy disk and carry it to a different computer on the network and then use the UK command on that computer with the file on the floppy disk.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **uk** on the NSYSPROF function line. This prompt appears on the function line; along with the name of the last directory that was typed in or else the name of the current directory if this is the first update key command:

```
Update public keys with Directory> SYS$DISK/ROOTDIR
```

Type the device and directory pathname of the directory where the second NODCONFIG.nnn.1 file exists. If you type without positioning into the old directory field, the new name replaces the old name, whereas if you position into the old field, you can edit it. (This editing behaves just like the search function in the VEW.) Use [CTRL] \ at any time to insert the previous old value where the cursor is.

Step 3 When the directory name is correct, strike [RETRN]. The following prompt then appears:

```
System model number>
```

Step 4 Enter a model number that corresponds to the extension of the second NODCONFIG.nnn.1 file. This model number can be any of the same model numbers that were allowed on the :model= switch.

Step 5 Strike [RETRN]. If an incorrect directory path was specified, an appropriate diagnostic message appears. If an incorrect model number was typed, the following message appears:

```
The model number is invalid.
```

Field Characteristics

The NSYSPROF display is a form-oriented display. This means that the screen contains fields in which you may type information and the rest of the screen you cannot affect. The NSYSPROF utility has three different kinds of fields. The cursor movement and control-key functions operate differently in each type of field. The four field types are: toggle, hex, and decimal. The NSYSPROF display fields and their types are:

Local Node Name	changed only with CN command
Private key field	changed only with DK, NK
Local Site ID	changed only with CS
Local Remote Network ID	hexadecimal field
Minimum receive cache buffers	decimal
Maximum receive cache buffers	decimal
Transmit buffer cache buffers	decimal
System overhead cache buffer	decimal
Default receive window size	decimal
Maximum calls waiting	decimal
First SVC call virtual circuit	decimal
Last SVC call virtual circuit	decimal
Remote site ID	changed only with AD, DA, DL
Public key field	changed only with UA, UK
Encryption fields	toggles

Cursor Movement and Control-Key Functions

If the cursor is after the last character in a field, the right-arrow key or any forward word/field movement key moves the cursor to the beginning of the next editable field.

If the cursor is on the last character of a field, any key moves the cursor to the beginning of the next editable field.

If the cursor is in the first position in a field, the left-arrow key or any backward word/field movement key moves the cursor to the end of the previous editable field.

The up-arrow and down-arrow keys move the cursor from a field on one line to another field on another line within the form. If the cursor is in a field in the first line of the form when you strike the up-arrow key, the cursor is moved to a field in the last line of the form. If the cursor is in a field in the last line of the form when you strike the down-arrow key, the cursor is moved to a field in the first line of the form.

Arrow keys   Each arrow key moves the cursor in the direction indicated by the arrow on the keycap of the arrow key.

[CTRL] a   Moves the cursor to the beginning of the first field on a line in the form.

[CTRL] b   Deletes from the digit to the left of the cursor to the beginning of the field. For a decimal field, the digits are deleted and the leftover digits are shifted to the left. For a hexadecimal field, the digits to the right change to zeroes.

[CTRL] d   For a decimal field, deletes all digits under the cursor to the end of the field. For a hexadecimal field, changes all digits from the digits under the cursor to the end of the field to zeroes.

[CTRL] e   Moves the cursor to the end of the last field of the form.

[CTRL] f   Moves the cursor to the beginning of the next field.

[CTRL] g   Moves the cursor to the end of the last field on a line in the form.

[CTRL] h   Pages to the previous node record.

[CTRL] k   Same as [CTRL] d.

[CTRL] o   Moves the cursor to the beginning of the first field in the next line of the form.

[CTRL] p   Pages to the next node record.

[CTRL] r   Moves the cursor to the end of the previous field.

[CTRL] t   Moves the cursor to the beginning of the first field in the form.

[CTRL] u Same as [CTRL] d.

[CTRL] v Deletes the digit in the cursor.

[CTRL] w Moves the cursor to the beginning of the first field in the previous line of the form.

[CTRL] y Deletes the contents of a decimal field. For a hexadecimal field, it sets all digits to zero.

[CTRL] z Repeats the last function.

[DEL] Deletes the character to the left of the cursor in a decimal field. In a hexadecimal field, it changes the digit to the left of the cursor to zero. [DEL] cannot be used to move the cursor from one field to another.

[RETRN] Moves the cursor to the beginning of the next field in the form.

Related CIP Commands

sysprof Set up the system configuration

nuserprof

Functional Description

Use this command to set up remote user accounts for your local system.

Command Line Syntax

Mnemonic	nuserprof	
Optional parameter	Directory	
Switches	:setupin=	:setupout=

Parameters

Directory	Function	Optional. Use this parameter to specify the destination directory for the NETUAF.DAT file.
	Default	SYS$DISK/SYSLIB/
	Syntax	Type a standard directory designation.

Switches

:setupin=	Function	Use this switch to specify an alternate setup file to use for sys$input. If you specify a directory path, NUSERPROF looks in that directory for the setup file based on the terminal type. If you specify an explicit file, it uses that file.
	Default	The setup file in sys$disk/syslib.setup/ determined by the terminal type is used, e.g., if your terminal is a T7000, then the file used is sys$disk/syslib.setup/setup252.sys.
	Syntax	Type **:setupin=** followed by a filename.
:setupout=	Function	Use this switch to specify an alternate setup file to use for sys$output. If you specify a directory path, NUSERPROF looks in that directory for the setup file based on the terminal type. If you specify an explicit file, it uses that file.
	Default	The setup file in sys$disk/syslib.setup/ determined by the terminal type is used, e.g., if your terminal is a T7000, then the file used is sys$disk/syslib.setup/setup252.sys.
	Syntax	Type **:setupout=** followed by a filename.

Examples

> nuserprof

This command allows the system manager to set up remote user accounts and define the access privileges and process attributes for remote users by editing a file called NETUAF.DAT. It generates a display like the following:

```
Remote Site ID    : 0001__GEORGE
Remote UIC [U,G]  : [0011,0003]


Local UIC [U,G]   : [0013,0004]    Priority: 7     Timeslice: 60

Local Privileges  :  OSETPRIV      NOSYSTEM        NOREADPHYS      NOWRITEPHYS
                    NOSETPRIOR     NOCHNGSUPER     NOBYPASS        NOOPERATOR
                    NOALTUIC       NOWORLD           GROUP           NETWORK
                    NOSETATTR

Local Attributes  :   SWAPPABLE    NOPREZEROMEM    NOPOSTZEROMEM   NODESENCRYPT
                      FASTENCRYPT    WATCHDOG
                    NOUSER1        NOUSER2         NOUSER3         NOUSER4

     Restrictions :   CRPROC       NOEXCLUDE
```

Using Prompts

None.

Notes on Usage

The NUSERPROF command should only be used by the system manager.

The Directory parameter tells NUSERPROF to work on a NETUAF.DAT file from the specified directory rather than in the default directory. If the Directory is on a network node other than the one that NUSERPROF is running on, NUSERPROF will actually run itself on the remote computer in order to update a page of name shared memory that is shared by the NSP processes that run on that computer.

NUSERPROF is modeled after the USERPROF utility. It allows you, as the system manager, to say that user X on remote computer A is allowed to access your local computer with a certain set of access privileges. The NUSERPROF records are maintained in a file called SYS$DISK/SYSLIB/NETUAF.DAT.1.

Detailed information on editing this file is presented under these headings:

The NUSERPROF Display
Functions Available in NUSERPROF
Cursor Movement and Control-Key Functions

The NUSERPROF Display

```
Remote Site ID    : 0001 __GEORGE
Remote UIC [U,G]  : [0011,0003]


Local UIC [U,G]   : [0013,0004]     Priority: 7    Timeslice: 60

Local Privileges  :   SETPRIV       NOSYSTEM       NOREADPHYS      NOWRITEPHYS
                    NOSETPRIOR      NOCHNGSUPER    NOBYPASS        NOOPERATOR
                    NOALTUIC        NOWORLD          GROUP           NETWORK
                    NOSETATTR

Local Attributes  :   SWAPPABLE     NOPREZEROMEM   NOPOSTZEROMEM   NODESENCRYPT
                    NOFASTENCRYPT     WATCHDOG
                    NOUSER1         NOUSER2        NOUSER3         NOUSER4

    Restrictions  :   CRPROC        NOEXCLUDE
```

The following is a description of each field in the NSYSPROF display:

Remote Site ID Field

This field contains the site ID number that uniquely identifies the computer on which the remote user's user account exists. It is a hexadecimal number which corresponds to the site ID of a computer on your network. This field is modified by the AD function. If the remote computer is up and running at the time you execute NUSERPROF, the nodename corresponding to this remote site ID will appear in the display, immediately to the right of the remote site ID number.

```
Remote Site ID   : 0001  __GEORGE
Remote UIC [U,G] : [0011,0003]


Local UIC [U,G]  : [0013,0004]    Priority: 7     Timeslice: 60

Local Privileges :  SETPRIV       NOSYSTEM        NOREADPHYS      NOWRITEPHYS
                   NOSETPRIOR     NOCHNGSUPER     NOBYPASS        NOOPERATOR
                   NOALTUIC       NOWORLD           GROUP           NETWORK
                   NOSETATTR

Local Attributes :  SWAPPABLE     NOPREZEROMEM    NOPOSTZEROMEM   NODESENCRYPT
                   NOFASTENCRYPT    WATCHDOG
                   NOUSER1        NOUSER2         NOUSER3         NOUSER4

    Restrictions :  CRPROC        NOEXCLUDE
```

Remote UIC Field

This field contains the User Identification Code (UIC) of the remote user on his remote computer. This field is also modified by the AD function. The asterisk wildcard symbol, *, is allowed in both the User ID and the Group ID portions of the UIC. For example, to identify all users in group 2 on site ID 3, the system manager would type 3 for the Remote Site ID, an asterisk for the Remote UIC User ID, and 2 for the Group ID. To identify all users in all groups on Remote Site ID 3, the system manager would also type an asterisk for the Group ID.

```
Remote Site ID   : 0001 __GEORGE
Remote UIC [U,G] : [0011,0003]


Local UIC [U,G]  : [0013,0004]    Priority: 7     Timeslice: 60

Local Privileges :  SETPRIV       NOSYSTEM        NOREADPHYS      NOWRITEPHYS
                   NOSETPRIOR     NOCHNGSUPER     NOBYPASS        NOOPERATOR
                   NOALTUIC       NOWORLD           GROUP           NETWORK
                   NOSETATTR

Local Attributes :  SWAPPABLE     NOPREZEROMEM    NOPOSTZEROMEM   NODESENCRYPT
                   NOFASTENCRYPT    WATCHDOG
                   NOUSER1        NOUSER2         NOUSER3         NOUSER4

    Restrictions :  CRPROC        NOEXCLUDE
```

Local UIC, Priority, and Timeslice Fields

The Local UIC field contains a local UIC to which the remote UIC will translate when it accesses the local computer. Like the Remote UIC field, the Local UIC field can contain specific numbers or asterisks. If you type specific numbers, you are indicating that the remote user identified on the top two lines will use the new specified UIC for all files and processes that are created on your local computer. In other words, the UIC will be converted to a new one on the local computer. If you type asterisks for either part of the UIC, you are indicating that that ID number is to be the same on the local computer. This flexibility allows the system manager to easily resolve situations where different users on different computers have the same UIC codes, because the UIC codes can be changed to local UICs when going to another computer where a conflict exists.

The Priority field indicates that all processes started on the local computer (including his NSP) by this remote user will use this given priority on the local computer. An asterisk is allowed here which indicates that the user's priority should be the same on the local computer as it is on the remote computer.

The Timeslice field indicates that all processes started on the local computer (including his NSP) by this remote user will use this given timeslice on the local computer. An asterisk is allowed here which indicates that the user's timeslice should be the same on the local computer as it is on the remote computer.

```
Remote Site ID    : 0001 __GEORGE
Remote UIC [U,G]  : [0011,0003]


Local UIC [U,G]   : [0013,0004]    Priority: 7     Timeslice: 60

Local Privileges  :  SETPRIV       NOSYSTEM        NOREADPHYS     NOWRITEPHYS
                    NOSETPRIOR     NOCHNGSUPER     NOBYPASS       NOOPERATOR
                    NOALTUIC       NOWORLD          GROUP           NETWORK
                    NOSETATTR

Local Attributes  :  SWAPPABLE     NOPREZEROMEM    NOPOSTZEROMEM  NODESENCRYPT
                    NOFASTENCRYPT   WATCHDOG
                    NOUSER1        NOUSER2         NOUSER3        NOUSER4

    Restrictions  :  CRPROC        NOEXCLUDE
```

Local Privileges Field

This field contains thirteen values which indicate what privileges this remote user will have when running or accessing files on the local node. Each field can have one of three values. For example, the SETPRIV privilege can be "SETPRIV", "NOSETPRIV" or " * SETPRIV". If it is set to "SETPRIV", the remote user will have SETPRIV privilege on the local node. If it is set to "NOSETPRIV", the remote user will not have SETPRIV privilege on the local node. If it is set to "* SETPRIV," it means that if the user has SETPRIV privilege on the remote node, he will have it on the local node; but if he does not have SETPRIV privilege on the remote node, he will not have it on the local node. In other words, an asterisk indicates that for this particular privilege the user will have on the local node whatever he has on the remote node. In order to change the "SETPRIV" value, the user must position the cursor to the start of the SETPRIV field. Striking the spacebar will toggle back and forth between the values "NOSETPRIV" and "SETPRIV", and typing an asterisk will change the value to "* SETPRIV". For more information on each of these privileges, see the USERPROF command description in this manual.

```
Remote Site ID    : 0001 __GEORGE
Remote UIC [U,G]  : [0011,0003]


Local UIC [U,G]   : [0013,0004]     Priority: 7     Timeslice: 60

Local Privileges  :  SETPRIV         NOSYSTEM        NOREADPHYS      NOWRITEPHYS
                    NOSETPRIOR      NOCHNGSUPER     NOBYPASS        NOOPERATOR
                    NOALTUIC        NOWORLD           GROUP           NETWORK
                    NOSETATTR

Local Attributes  :  SWAPPABLE       NOPREZEROMEM    NOPOSTZEROMEM   NODESENCRYPT
                    NOFASTENCRYPT    WATCHDOG
                    NOUSER1         NOUSER2         NOUSER3         NOUSER4

    Restrictions  :  CRPROC          NOEXCLUDE
```

Local Attributes Field

This field contains ten values which indicate what process attributes will be assigned to processes created by the remote user on the local node. Each field can have one of three values. For example, the SWAPPABLE attribute can be "SWAPPABLE", "NOSWAPPABLE" or "* SWAPPABLE". If it is set to "SWAPPABLE", all processes created by the remote user on the local node will be SWAPPABLE. If it is set to "NOSWAPPABLE", all processes created by the remote user on the local node will not be SWAPPABLE. If it is set to "* SWAPPABLE", then if the remote user's process that creates a process on the local node is SWAPPABLE, then the process on the local node will be SWAPPABLE, too; but if the remote process was NOSWAPPABLE, the local process will be NOSWAPPABLE also. In other words, an asterisk indicates that for this particular process attribute, any processes created on the local node by a process on a remote node will take on the attribute of the creating process. In order to change the "SWAPPABLE" value, the user must position the cursor to the start of the SWAPPABLE field. Striking the spacebar will toggle back and forth between the values "NOSWAPPABLE" and "SWAPPABLE", and typing an asterisk will change the value to "* SWAPPABLE". For more information on each of these process attributes, see the USERPROF command description in this manual.

```
Remote Site ID    : 0001  __GEORGE
Remote UIC [U,G]  : [0011,0003]


Local UIC [U,G]   : [0013,0004]    Priority: 7     Timeslice: 60

Local Privileges  :  SETPRIV       NOSYSTEM        NOREADPHYS      NOWRITEPHYS
                    NOSETPRIOR     NOCHNGSUPER     NOBYPASS        NOOPERATOR
                    NOALTUIC       NOWORLD           GROUP           NETWORK
                    NOSETATTR

Local Attributes  :  SWAPPABLE     NOPREZEROMEM    NOPOSTZEROMEM   NODESENCRYPT
                    NOFASTENCRYPT   WATCHDOG
                    NOUSER1        NOUSER2         NOUSER3         NOUSER4

    Restrictions  :  CRPROC        NOEXCLUDE
```

__Restrictions Field__

This field contains two boolean toggle values with the following meanings:

CRPROC If enabled, this remote user is allowed to create processes that run on the local computer. If disabled (NOCRPROC), this remote user is not allowed to create processes that run on the local computer. The inability to create processes on the local computer does not affect the ability to access local files and other resources.

EXCLUDE If enabled, this record is not an authorization record which lets the specified user access the local computer. Instead, it is a record defining a remote user (or set of users) which will be EXCLUDED from accessing this local computer. In other words, it is a way of keeping certain users or groups of users off of your local computer. This feature is convenient to use with the asterisk wildcard symbol. If disabled (NOEXCLUDE), this record is an authorization record.

Functions Available in NUSERPROF

The editing functions for NUSERPROF are similar to USERPROF. Each of the following functions is executed by first striking [ESC][ESC], then typing the mnemonic on the function line that appears at the bottom of the screen. The position of the cursor when you strike [ESC][ESC] does not affect the execution of any of these functions.

ab

Abort this session of NUSERPROF.

FUNCTIONAL DESCRIPTION

AB exits this editing session of NUSERPROF without writing any of the changes made during the session to the NETUAF.DAT file.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

ex	Exit NUSERPROF
sa	Save modifications

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type ab on the NUSERPROF function line. If no modifications have been made, the screen is cleared, and you return to your CIP prompt. If any modifications have been made to the NETUAF.DAT file, this prompt appears on the function line:

```
NETUAF.DAT file has been modified - abort?
```

Step 3 Type y or n. If you type some other character, the following message briefly appears on your screen:

```
Please type 'Y' or 'N'.
```

If you type Y, the screen is cleared and you return to your CIP prompt. If you type N, the cursor returns to the position it had on the screen before executing AB.

ad

Add a remote authorization record.

FUNCTIONAL DESCRIPTION

AD creates a record for a remote user with default values being supplied by NUSERPROF, and orders the records in NETUAF.DAT so every user record appears in [siteid, group, user] order. The asterisks in group and user are sorted at the end of all valid nonwildcarded fields.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

dl Delete a remote user authorization record.

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **ad** on the NUSERPROF function line. This prompt appears on the function line:

```
Add Remote Site ID>
```

Step 3 Type the Site ID where the remote user's login account exists. This is a hexadecimal number up to four digits long.

Step 4 Strike [RETRN]. NUSERPROF will then write the following prompt following the Site ID on the function line:

```
Remote UIC User ID>
```

Step 5 Type the User ID portion of the remote user's UIC field if you want to authorize a specific user, or type an asterisk if you want to authorize any user ID for the "to be specified" group.

Step 6 Strike [RETRN]. NUSERPROF will then write the following prompt following the User ID on the function line:

```
Group ID>
```

Step 7 Type the group ID portion of the remote user's UIC field if you desire to authorize a specific group, or type an asterisk if you want to authorize all groups from the specified remote site ID.

Step 8 Strike [RETRN].

cp

Spawn a CIP while remaining in NUSERPROF.

FUNCTIONAL DESCRIPTION

CP spawns a CIP from within NUSERPROF.

CORRESPONDING CONTROL-KEY FUNCTIONS

[ESC] c

RELATED ESCAPE-KEY FUNCTIONS

ex	Exit NUSERPROF

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **cp** on the NUSERPROF function line. The CIP prompt appears. When you log off from the CIP, you are returned to the same record you were editing before spawning the CIP.

dl

Delete a record from NETUAF.DAT.

FUNCTIONAL DESCRIPTION

DL deletes the currently displayed record from NETUAF.DAT.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

ad	Add a remote user authorization record

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type dl on the function line. This prompt appears on the function line:

```
Delete record ?
```

Step 3 Type n if you do not want to delete the record, or type y if you wish to delete it.

ex

Exit NUSERPROF, i.e., terminate the editing session.

FUNCTIONAL DESCRIPTION

EX saves any remote user authorization record modifications, terminates the editing session, and returns you to the CIP.

CORRESPONDING CONTROL-KEY FUNCTIONS

{-}

RELATED ESCAPE-KEY FUNCTIONS

ab	Abort NUSERPROF
sa	Save modifications

EXECUTION

Step 1 | Strike[ESC][ESC].

Step 2 | Type **ex**. This report appears at the bottom of the screen:

```
Exiting ...
```

The screen is cleared and the cursor reappears next to your CIP prompt.

he

Call to the screen a display of commands available in NUSERPROF.

FUNCTIONAL DESCRIPTION

HE displays the list of commands available in NUSERPROF.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **he**. The first page of a three-screen help display appears on your screen. The first screenful contains a list of the escape-key functions available.

The second screenful lists functions for deletion of the field and how to move around the screen.

The third screenful describes the different field types on the screen display of a record.

Step 3 Press [RETRN] to cycle through the pages, then back to your current record. Press [CTRL] c at any time to return to your current record without showing the remaining help screens.

mp

Move toward the beginning or end of NETUAF.DAT.

FUNCTIONAL DESCRIPTION

MP calls to the screen the record immediately following the record on the screen. This message appears on the screen when the last record in NETUAF.DAT is already on the screen:

```
No next record ...
```

-MP calls to the screen the record immediately preceding the record on the screen. This message appears on the screen if the first record is on the screen when you execute -MP:

```
No prior record ...
```

CORRESPONDING CONTROL-KEY FUNCTIONS

[CTRL] p	(for MP)
[CTRL] h	(for -MP)

RELATED ESCAPE-KEY FUNCTIONS

sh	Show a remote user record
sf	Show first remote user record
sl	Show last remote user record
sr	Search for remote user record

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type mp or -mp. Depending on whether you type mp or -mp, one of the following reports appears momentarily on the screen:

```
Paging to next record.

Paging to prior record.
```

The next record appears on the screen, and the cursor appears over the first character of the same field that the cursor was on in the next (or previous) record.

pr

Create a text-file copy of the records in the NETUAF.DAT file.

FUNCTIONAL DESCRIPTION

PR creates a file named NETUAF.PRN in the current directory. NETUAF.PRN contains a copy of all of the records in NETUAF.DAT.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **pr** on the function line. This prompt appears on the function line:

```
Printing NETUAF.DAT records...
```

When the records are copied to the NETUAF.PRN file, the cursor returns to the position it had prior to the PR command.

rp

Repeat last function.

FUNCTIONAL DESCRIPTION

RP repeats the last function you executed.

CORRESPONDING CONTROL-KEY FUNCTIONS

[CTRL] z

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1	Strike [ESC][ESC]
Step 2	Type **rp** on the NUSERPROF function line. The last function is repeated.

rs

Refresh the screen.

FUNCTIONAL DESCRIPTION

RS restores the display on the screen when a message or another interruption disrupts the display.

CORRESPONDING CONTROL-KEY FUNCTIONS

[CTRL] _

RELATED ESCAPE-KEY FUNCTIONS

ud	Undo the modifications to a record

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **rs**. The screen is erased and redisplayed so the record reappears on the screen as it was before the interruption.

sa

Save the modifications made so far in the NETUAF.DAT file.

FUNCTIONAL DESCRIPTION

SA writes out all modified remote user authorization records to the NETUAF.DAT file without exiting the editing session. This is useful when you have a large number of modifications and want to protect them from system failure.

CORRESPONDING CONTROL-KEY FUNCTIONS

{,}

RELATED ESCAPE-KEY FUNCTIONS

ab	Abort the editing session
ex	Exit NUSERPROF

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type sa on the NUSERPROF function line. This prompt appears on the function line:

```
Saving modifications...
```

After writing all the modified records to the NETUAF.DAT file, and updating the named shared page of memory, the cursor returns to the position it had prior to the SA command.

sf

Show the first record in the NETUAF.DAT file.

FUNCTIONAL DESCRIPTION

SF displays the first record in the NETUAF.DAT file. If you then page forward using the MP command, the records will be displayed in site ID, group ID, user ID order.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

sh	Show a specific record
sl	Show the last record in the NETUAF.DAT file
sr	Search for a specific record

EXECUTION

Step 1 | Strike [ESC][ESC].

Step 2 | Type sf on the NUSERPROF function line. This prompt appears briefly on the function line:

```
Showing the first record...
```

The first record is displayed on the screen.

sh

Show a specific record.

FUNCTIONAL DESCRIPTION

SH allows you to specify the record you want to examine or modify.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

mp	Move page
sf	Show the first record in the NETUAF.DAT file
sl	Show the last record in the NETUAF.DAT file
sr	Search for a specific record

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type sh. This prompt appears at the bottom of the screen:

```
Show Remote Site ID>
```

Step 3 Type the remote site ID assigned to the record you want to examine.

Step 4 Strike [RETRN]. NUSERPROF then displays this prompt:

```
Remote UIC User ID>
```

Step 5 Type the user ID of the record you want to show.

Step 6 Strike [RETRN]. NUSERPROF then displays this prompt:

```
Group ID>
```

Step 7 Type the group ID of the record you want to show.

Step 8 Strike [RETRN]. If the record exists, it is displayed on the screen. If the specified remote record does not exist, this diagnostic message appears on the screen:

```
The SID/UIC combination does not exist. Please enter another combination.
```

sl

Show the last record in the NETUAF.DAT file.

FUNCTIONAL DESCRIPTION

SL displays the last record in the NETUAF.DAT file. If you then page backward using the -MP command, the records will be displayed in reverse order.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

sf	Show the first record in the NETUAF.DAT file
sh	Show a specific record in the NETUAF.DAT file
sr	Search for a specific record in NETUAF.DAT

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type sl on the NUSERPROF function line. This prompt appears briefly on the function line:

```
Showing the last record...
```

The last record is displayed on the screen.

sr

Search for a specific remote user authorization record.

FUNCTIONAL DESCRIPTION

SR allows you to search for a specific remote user authorization record you want to examine or modify. This differs from SH only in that if the record you specify does not exist, no message is displayed on the bottom of the screen, and the record just following where the search record would normally be is displayed, unless the search record is beyond the end of the file, in which case the last record will be displayed.

CORRESPONDING CONTROL-KEY FUNCTIONS

{7}

RELATED ESCAPE-KEY FUNCTIONS

sh	Show a specific record
sf	Show the first record
sl	Show the last record

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **sr.** This prompt appears on the function line:

```
Search for Remote Site ID>
```

Step 3 Type the Site ID assigned to the record you want to search for. Then the following prompt appears on the function line:

```
Remote UIC User ID>
```

Step 4 Type the user ID of the record you want to search for followed by [RETRN], or simply strike [RETRN] without a user ID. Then the following prompt appears on the function line:

```
Group ID>
```

Step 5

Type the group ID of the record you want to search for followed by [RETRN], or simply strike [RETRN] without a group ID. NUSERPROF will then search for the specific record. If it exists it will be displayed. If there is not an exact match, NUSERPROF finds the position in the file where the record would exist if it were in the file and then displays the record following that position. If that position is after the end of the file, NUSERPROF displays the last record in the file.

ud

Undo changes made on the remote user record.

FUNCTIONAL DESCRIPTION

UD undoes changes to cursor-editable fields of the record, i.e. fields that can be edited without using the [ESC][ESC] key.

UD can be performed at any time until a different record is displayed. At that point, changes to the record have been made permanent.

CORRESPONDING CONTROL-KEY FUNCTIONS

{PF1}

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 | Strike [ESC][ESC].

Step 2 | Type **ud.** This report appears momentarily on the screen:

```
Undoing changes...
```

The remote user authorization record reappears as it did immediately following the last executed escape-key function.

Cursor Movement and Control-Key Functions

If the cursor is after the last character in a field, the right-arrow key or any forward word/field movement key moves the cursor to the beginning of the next editable field.

If the cursor is on the last character of a field, any key moves the cursor to the beginning of the next editable field.

If the cursor is in the first position in a field, the left-arrow key or any backward word/field movement key moves the cursor to the end of the previous editable field.

The up-arrow and down-arrow keys move the cursor from a field on one line to another field on another line within the form. If the cursor is in a field in the first line of the form when you strike the up-arrow key, the cursor is moved to a field in the last line of the form. If the cursor is in a field in the last line of the form when you strike the down-arrow key, the cursor is moved to a field in the first line of the form.

Arrow keys — Each arrow key moves the cursor in the direction indicated by the arrow on the keycap of the arrow key.

[CTRL] a — Moves the cursor to the beginning of the first field on a line in the form.

[CTRL] b — Deletes from the digit to the left of the cursor to the beginning of the field. For a decimal field, the digits are deleted and the leftover digits are shifted to the left. For a hexadecimal field, the digits to the right change to zeroes.

[CTRL] d — For a decimal field, deletes all digits under the cursor to the end of the field. For a hexadecimal field, changes all digits from the digits under the cursor to the end of the field to zeroes.

[CTRL] e — Moves the cursor to the end of the last field of the form.

[CTRL] f — Moves the cursor to the beginning of the next field.

[CTRL] g — Moves the cursor to the end of the last field on a line in the form.

[CTRL] h — Pages to the previous node record.

[CTRL] k — Same as [CTRL] d.

[CTRL] o — Moves the cursor to the beginning of the first field in the next line of the form.

[CTRL] p — Pages to the next node record.

[CTRL] r — Moves the cursor to the end of the previous field.

[CTRL] t — Moves the cursor to the beginning of the first field in the form.

[CTRL] u — Same as [CTRL] d.

[CTRL] v — Deletes the digit in the cursor.

[CTRL] w — Moves the cursor to the beginning of the first field in the previous line of the form.

[CTRL] y Deletes the contents of a decimal field. For a hexadecimal field, it sets all digits to zero.

[CTRL] z Repeats the last function.

[DEL] Deletes the character to the left of the cursor in a decimal field. In a hexadecimal field, it changes the digit to the left of the cursor to zero. [DEL] cannot be used to move the cursor from one field to another.

[RETRN] Moves the cursor to the beginning of the next field in the form.

Related CIP Commands

userprof Set up user accounts

option

Functional Description

Use this command to display or specify a CIP control option.

Command Line Syntax

Mnemonic	option			
Switches				
Options for utilities	:log	:path=	:message	:pause
Options for CIP	:prompt= :protection=	:perm	:verify	:trace

Parameters

None.

Switches

:log	Function	Use this switch to specify whether log messages are displayed by all of the utilities.
	Default	:log
	Syntax	Type :nolog

:message	Function	Use this switch to specify whether all utilies display message lines when an error occurs.
	Default	:message
	Syntax	Type :nomessage
:path=	Function	Use this switch to specify what directories to look in and the order to look when executing an image file or a command file.
	Default	:path=/sysexe/,/sysexe.users/
	Syntax	Type :path= followed by a list of directories separated by commas. A null path means the default directory. If there is a leading slash on a directory string, SYS$DISK is appended to the front of it. If a leading slash is not on a directory string, it looks at the default disk. Device specifications are valid in these strings. This string may be up to 93 characters long.
:pause	Function	Use this switch to specify whether all utilities should pause after each screen of data.
	Default	:nopause
	Syntax	Type :pause NOTE: The user can override the state of this switch in each utility by specifying :pause or :nopause on the command line.
:perm	Function	Use this switch to make the change permanent or to make it persist even after the current CIP ends execution (unless the current CIP is your user process).
	Default	:noperm
	Syntax	Type :perm
:prompt=	Function	Use this switch to change or set the CIP prompt.
	Default	The prompt is not changed.
	Syntax	Type :prompt= followed by the prompt enclosed in double quotation marks.
:protection=	Function	Use this switch to change the default protection mask for CIP. When a utility creates a file, this protection mask is assigned to it.
	Default	The protection mask specified in the UAF.DAT file.

Syntax — Type **:protection=** followed by a protection mask.

Note that only those fields of the protection mask specified are altered. Fields not specified keep their original value.

:verify

Function — Use this switch to display the lines of the command file before execution exactly as they are in the file.
Default — :noverify
Syntax — Type **:verify**

:trace

Function — Use this switch to see what lines are being executed after the CIP has parsed the lines in the command file.
Default — :notrace
Syntax — Type **:trace**

Examples

> option

This command shows the state of all options in a display like this:

```
Home        : SYS$DISK/USERS.TEST/
Log         : Yes
Message     : Yes
Path        : ,/sysexe/,/sysexe.users/
Pause       : No
Prompt      : TEST>
Protection : S:  RE,P:     ,G: WRE,O:DWRE
SYS$RESULT : 0
Username    : TEST
Verify      : No
Trace       : No
```

> option :verify :nomessage

The :verify switch causes all command files to display the contents of the command file before execution exactly as it is in the file. The :nomessage switch suppresses the message line of an error display when an error occurs.

```
> option :prompt="Test> " :perm
```

This command changes the CIP prompt to Test>. If this command is executed inside of a command file, it changes the prompt of the interactive CIP executing the command file to this new value. By placing this command in his USERUP.COM file a user can determine his CIP prompt at log on time.

Using Prompts

None.

Notes on Usage

The :perm switch is used in command files to set OPTION for the interactive CIP used to execute the command file. Note that OPTION applies to only the most recent interactive CIP.

Related CIP Commands

None.

password

Functional Description

Use this command to assign or change your password.

Command Line Syntax

Mnemonic	password
Switches	:log

Parameters

None.

Switches

:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type :nolog or :log to override the default.

Examples

```
> password
```

This command generates the following messages, one after the other:

```
    New Password    >
    Verify Password>
    Password modified.
```

The first time you log on to a system, you should set up a password for your account (if the system manager has not done it for you).

If no password has previously been specified, follow these steps:

Step 1 Type only ASCII characters, ranging from ! to ~, after the prompt for New Password. (To see the ASCII character table, type **ascii?** on the CIP command line.)

If you type spaces or any invalid characters, they are ignored. The echo is suppressed as you type it so others cannot see the new password.

Step 2 Type the new password again, exactly as you did in step 1, after the prompt for Verify Password.

If you type something that does not match the new password, the following messages appear:

```
Error - New password did not verify. Try again.

New Password>
```

If this happens, repeat steps 1 and 2.

Step 3 After the WMCS has verified your password, this message appears:

```
Password modified.
```

```
> password :nolog

    Old Password    >
    New Password    >
    Verify Password>
```

In this example a password has already been specified for the user. The first step in this case is to type in the previous password.

To change a password you must enter the previous one. As you type the password, the echo is suppressed for security. If you type your old password incorrectly, the following message appears on your screen:

```
     PASSWORD -- User validation error.
```

If you type your old password five times incorrectly, a 30-second delay is executed. This helps prevent password stealing. Once the old password is typed correctly, follow the steps in the previous example.

The :nolog switch suppresses the message "Password Modified".

Using Prompts

```
> password :nolog

Old Password    >
New Password    >
Verify Password>
Password modified.
```

This command performs the same function as the last example. The passwords are not echoed to the screen when they are typed.

Notes on Usage

You can only change the password for the default user as specified by SYS$USERNAME.

You have only 20 seconds to type the password for each of the three prompts or the following message appears:

```
     PASSWORD -- Input timeout error
```

Related CIP Commands

logon	Log on to a system
userprof	Set up the user profiles on a system

Functional Description

Use this command to dump the physical memory or the logical memory of a process.

Command Line Syntax

Mnemonic	peek
Required parameter	Address
Optional parameters	Nbytes Offset

Switches

Memory dump control	:array :base= :override	:list :long	:pid= :simple	:byte :word
Display	:pause			

Parameters

Address	Function	Required. Use this parameter to specify the memory address at which to start dumping memory.
	Default	None.
	Syntax	Type the address in hexadecimal in this format: [*]address[(+/-)offset].

Type a percent sign, %, preceding the address to specify an address in decimal. This parameter may be the address of a location in memory in which the actual address of memory to be dumped is stored; i.e., this parameter may be the address of a pointer. If so, precede the address with an asterisk, *, to specify that the actual address is obtained by going indirect through the given address. An optional positive or negative offset may be specified by typing a plus or minus sign and an offset value after the address. The offset default base is hexadecimal. Precede the offset value with a percent sign to specify an offset in decimal. If indirection is specified, the offset is added to the actual address after the indirection has been performed.

Nbytes	Function	Optional. Use this parameter to specify the number of bytes of memory to dump.
	Default	16 bytes (one line of dump display)
	Syntax	Type a positive number. The default base is hexadecimal. Precede the parameter with a percent sign to specify the value in decimal.
Offset	Function	Optional. Use this parameter to specify the offset to a pointer (when using the :list switch) or to specify the number of array elements (when using the :array switch).
	Default	0 (for :list) or 1 (for :array)
	Syntax	Type a hexadecimal number. Precede the parameter with a percent sign to specify a value in decimal. A negative value may be specified for a pointer offset.

Switches

:array	Function	Use this switch to enable "array" mode. In array mode, the memory specified by the Address parameter is treated as the address of an array, with the number of array elements specified by the Offset parameter. After memory is dumped for the number of bytes specified by the Nbytes parameter, the address is

incremented by the number of bytes dumped. The memory dump iterates the number of times specified by the Offset parameter. This mode allows a user to do a dump of each element in an array in memory.

Default :simple

Syntax Type **:array** to enable array mode.

NOTE: This switch is mutually exclusive with the :list and :simple switches.

:base=

Function Use this switch to specify a value to add to the Address parameter before memory dumping starts. This switch is useful when something is known to exist at a specific address and the memory to be dumped is relative to the address. The base address is specified with this switch, and the relative address with the Address parameter.

Default :base=0

Syntax Type **:base=** followed by a hexadecimal value and an optional offset value.

Precede the value with a percent sign to specify a decimal base. The value may be the address of a location in memory in which the actual value is stored. If so, precede the value with an asterisk to specify that the actual value is obtained by going indirect through the given address. An optional positive or negative offset may be specified by following the value with plus or minus sign and an offset value. The offset default base is hexadecimal. Precede the offset with a percent sign to specify an offset value in decimal. If indirection is specified, the offset is added to the actual value after the indirection has been performed.

:byte

Function Use this switch to specify that memory is accessed one byte at a time. This is required if peeking at I/O devices which require byte accesses.

Default :byte

Syntax Type **:nobyte**

NOTE: This switch is mutually exclusive with the :word and :long switches.

:list Function Use this switch to enable "list" mode. In list mode, the memory specified by the Address parameter is treated as the address of a node in a linked list. The offset to a pointer to the next node in the list is specified by the Offset parameter. After memory is dumped for the number of bytes specified by the Nbytes parameter, the next address is retrieved by going indirect through the address obtained by adding the offset to the memory address. The memory dump iterates until a pointer is encountered which contains all zero bits or all one bits. This mode allows a user to do a dump of each node in a linked list in memory.

Default :simple

Syntax Type **:list**

NOTE: This switch is mutually exclusive with the :array and :simple switches.

:long Function Use this switch to specify that memory is accessed one longword (4 bytes) at a time. This is required if peeking at I/O devices which require longword accesses. The Nbytes parameter should be a multiple of four. If it is not, not all of the specified bytes will be dumped, since only longwords are accessed. For example, the Nbytes parameter is 7, only 4 bytes will be dumped.

Default :byte

Syntax Type **:long**

NOTE: This switch is mutually exclusive with the :byte and :word switches.

:override Function Use this switch to disable valid address checking. This is required to peek at I/O addresses. Memory addresses are checked for validity before dumping. Physical memory addresses must be in the range $200000 through $200000+number-of-bytes of physical memory. For example, on a system with one megabyte of memory, valid physical memory addresses are $200000 through $2fffff. Logical memory addresses require the specification of a process with the :pid= switch and memory allocated to the given process at the logical addresses. If the logical or physical memory does not exist, PEEK terminates with an error message. This switch overrides address checking with its error message and process termination.

Default :nooverride

Syntax Type **:override**

:pause

Function Use this switch to pause while dumping memory when a node in a linked list, an array element, or 256 bytes have been dumped. Pressing any key continues the memory dump.

Default :pause

Syntax Type **:nopause**

:pid=

Function Use this switch to specify the process ID (PID) or process name. If the process does not exist, PEEK will terminate with an error message. When a logical address is encountered, PEEK translates it into a physical memory address using the values of the mapping registers of the given process. If a logical address is encountered and no process has been specified, PEEK will terminate with an invalid physical address error.

Default None.

Syntax Type **:pid=** followed by a PID (default base is hexadecimal) or process name.

NOTE: The process MUST be executing on the same system as PEEK. A PID of 0 cannot be used.

:simple

Function Use this switch to do a simple dump memory, i.e., not in list or array mode.

Default :simple

Syntax Type **:list** or **:array**

NOTE: This switch is mutually exclusive with the :array and :list switches.

:word

Function Use this switch to specify that memory is accessed one word (2 bytes) at a time. This is required if peeking at I/O devices which require word accesses. The Nbytes parameter should be a multiple of two. If it is not, not all of the specified bytes will be dumped, since only words are accessed. For example, if the Nbytes parameter is 7, only 6 bytes will be dumped.

Default :byte
Syntax Type **:word**

NOTE: This switch is mutually exclusive with the :byte and :long switches.

Examples

> peek 200000

This command generates the following kind of display:

```
00200000  E4 24 00 00  FF 22 FF 00  AA 55 00 00  4E F9 00 20   .$..."...U..N..
```

> peek 202000 %400

This command generates the following kind of display:

```
00202000  70 02 60 00  08 A4 12 80  7E 01 60 00  FE 92 50 43   p.`.....~.`...PC
00202010  53 52 44 30  44 31 44 32  44 33 44 34  44 35 44 36   SRD0D1D2D3D4D5D6
00202020  44 37 41 30  41 31 41 32  41 33 41 34  41 35 41 36   D7A0A1A2A3A4A5A6
00202030  41 37 55 53  53 53 57 30  4D 30 57 31  4D 31 57 32   A7USSSW0M0W1M1W2
00202040  4D 32 57 33  4D 33 57 34  4D 34 57 35  4D 35 57 36   M2W3M3W4M4W5M5W6
00202050  4D 36 57 37  4D 37 3F 3F  61 00 09 28  2C EA 01 22   M6W7M7??a..(,.."
00202060  61 00 09 A8  47 D2 49 F9  00 20 20 0E  43 F9 00 20   a...G.I..  .C..
00202070  0D 00 20 0E  90 8D 0C 40  00 40 6F 0A  61 00 08 FA   .. ....@.@o.a...
00202080  61 00 09 88  60 EC 30 1C  12 1B 2E 19  0C 40 3F 3F   a...`.0......@??
00202090  66 58 BB CE  67 08 61 00  08 E0 61 00  09 6E 4A 2A   fX..g.a...a..nJ*
002020A0  00 24 67 0C  7E 44 41 F9  00 20 0D 08  61 00 F6 B4   .$g.~DA.. ..a...
002020B0  4A 2A 00 25  67 0C 7E 41  41 F9 00 20  0D 28 61 00   J*.%g.~AA.. .(a.
002020C0  F6 A2 10 2A  00 26 67 1A  0C 00 00 30  67 14 61 00   ...*.&g....0g.a.
002020D0  08 B2 72 40  1C C0 53 81  66 FA 61 00  08 9C 61 00   ..r@..S.f.a...a.
002020E0  09 2A 2C EA  01 26 60 00  09 22 4A 01  67 84 34 00   .*,..&`.."J.g.4.
002020F0  E0 40 16 00  1C C0 1C C2  1C FC 00 3D  0C 42 41 37   .@.........=.BA7
*
00202100  66 14 20 39  00 20 0D 04  02 40 20 00  66 16 20 39   f. 9. ...@ .f. 9
00202110  00 20 0D 48  60 46 0C 42  53 53 66 08  20 39 00 20   . .H`F.BSSf. 9.
00202120  0D 44 60 38  20 07 0C 03  00 57 67 06  0C 03 00 4D   .D`8 ....Wg....M
00202130  66 2A 2C 01  1E 03 02 42  00 0F E7 42  43 EA 00 28   f*,....B...BC..(
00202140  D2 C2 4A 11  66 06 59 8E  60 00 00 D4  61 00 F9 B8   ..J.f.Y.`...a...
00202150  0C 07 00 57  67 04 61 00  FA 04 22 06  0C 01 00 01   ...Wg.a...".....
00202160  66 08 61 00  04 E2 60 00  00 B6 0C 01  00 02 66 08   f.a...`.......f.
00202170  61 00 04 CC  60 00 00 A8  0C 01 00 03  66 08 61 00   a...`.......f.a.
00202180  04 B8 60 00  00 9A 0C 01  00 5A 66 14  53 8E 1C FC   ..`......Zf.S...
```

Note that a pause (indicated by the asterisk, *) was performed after 256 bytes were dumped.

```
> peek *2002a6 :list :nopause
```

This command generates the following kind of display:

```
0022D1C0  00 22 D0 A4  00 00 00 00  44 53 30 00  00 00 00 00   ."......DS0.....

*** Next item in list ***

0022D0A4  00 22 D1 40  00 22 D1 C0  54 54 30 00  00 00 00 00   .".@."..TT0.....

*** Next item in list ***

0022D140  00 2B 50 84  00 22 D0 A4  4E 55 4C 4C  00 00 00 00   .+P.."..NULL....

*** Next item in list ***

002B5084  00 31 C9 E8  00 22 D1 40  54 54 38 00  00 00 00 00   .1...".@TT8.....

*** Next item in list ***

0031C9E8  00 00 00 00  00 2B 50 84  44 58 30 00  00 00 00 00   .....+P.DX0.....

*** End of List ***
```

Note the use of indirection on the Address parameter to locate the first node in the linked list.

```
> peek *2e56 24 2 :base=*1001d0-8 :pid=3f8 :array :nopause
```

This command generates the following kind of display:

```
*** Array element [1] ***

001E8440  00 1E 83 B8  00 00 00 00  08 42 00 90  00 00 42 42   .........B....BB
001E8450  00 00 00 03  42 42 02 06  09 00 2B 0A  00 90 FF FF   ....BB....+.....
001E8460  FF FE 00 00                                          ....

*** Array element [2] ***

001E8464  00 1E 85 50  00 00 00 00  08 42 00 00  00 00 42 42   ...P.....B....BB
001E8474  00 00 00 01  42 42 00 00  0A 00 00 00  00 00 42 42   ....BB........BB
001E8484  42 42 42 42                                          BBBB

*** End of Array ***
```

Note that the addresses are in the logical address space of a process.

Using Prompts

```
> peek
Address        > 200000
```

This command performs the same function as the first example.

Notes on Usage

PEEK requires the user to have CHSUPER (change supervisor mode) privilege.

The starting address for the memory dump is computed in the following manner. Assume the Memory Address parameter is *40-8, the :base= switch has a value of *2001d8+200, memory location 2001d8 has a value of 240000, and memory location 240240 has a value of 280000. The base address is computed first by going indirect through 2001d8 and retrieving the value 240000. The base offset +200 is added to 240000 for a result of 240200. This is added to the memory address 40 for a result of 240240. This result is used as an indirect address to retrieve the value 280000. The address offset -8 is added to 280000 for a starting address of 27fff8.

Related CIP Commands

poke	Write data into memory

poke

Functional Description

Use this command to write data to physical memory or the logical memory of a process.

Command Line Syntax

Mnemonic	poke
Required parameters	Address Data

Switches

Memory dump control	:base= :byte	:long :override	:pid=	:word
Display	:header	:log	:pause	

Parameters

Address	Function	Required. Use this parameter to specify the memory address at which to start writing data to memory.
	Default	None.
	Syntax	Type a hexadecimal address in this format: [*]address[(+/-)offset]. Precede the parameter with a percent sign, %, to specify an address in decimal. This parameter may be the address of a location in memory in which the actual address of memory to be written is stored; i.e., this parameter may

be the address of a pointer. If so, precede the parameter with an asterisk, *, to specify that the actual address is obtained by going indirect through the given address. An optional positive or negative offset may be specified by following the address with a plus or minus sign and an offset value. The offset default base is hexadecimal. Precede the offset value with a percent sign to specify an offset in decimal. If indirection is specified, the offset is added to the actual address after the indirection has been performed.

Data

Function — Required. Use this parameter to specify the data to write to memory.

Default — None.

Syntax — Type hexadecimal, decimal, or text data. Any data which has spaces as either delimiters or data must be enclosed in double quotation marks.

Hexadecimal data is specified as hexadecimal digits with one or more spaces separating data bytes, e.g., "7f ff ff ff".

Decimal data is the same as hexadecimal data except that each data byte is preceded by a percent sign, e.g., "%127 %255 %255 %255" is the same as the hexadecimal example.

Text data must be enclosed in parentheses, e.g., (ABCD) or "(AB CD EFGH)". Any special characters which must be text data must be preceded by an accept character, i.e., the characters "[]()\" must be preceded by a backslash, \. For example, (\(\)) produces the text data "()".

A repeat count may be specified for data which repeats. Square brackets, [], must enclose the repeating data and be preceded by a repeat count. There may not be a space between the repeat count and the leading bracket. An example of the hexadecimal example with a repeat count is "7f 3[ff]". Repeat counts may be nested to any level so long as the expanded data does not exceed 3072 bytes.

Switches

:base=	Function	Use this switch to specify a value to add to the Address parameter before writing data to memory. This switch is useful when something is known to exist at a specific address and the memory to be written is relative to the address. The base address is specified with this switch, and the relative address with the Address parameter.
	Default	:base=0
	Syntax:	Type **:base=** followed by a value in this format: [*]value[(+/-)offset]. The value default base is hexadecimal. Precede the value with a percent sign to specify a decimal base. The value may be the address of a location in memory in which the actual value is stored. If so, precede the value with an asterisk to specify that the actual value is obtained by going indirect through the given address. An optional positive or negative offset may be specified by following the value with a plus or minus sign and an offset value. The offset default base is hexadecimal. Precede it with a percent sign to specify an offset value in decimal. If indirection is specified, the offset is added to the actual value after the indirection has been performed.
:byte	Function	Use this switch to specify that memory is accessed one byte at a time. This is required if poking I/O devices which require byte accesses.
	Default	:byte
	Syntax	Type **:nobyte** NOTE: This switch is mutually exclusive with the :word and :long switches.
:header	Function	Use this switch to produce headings for POKE output.
	Default	:header
	Syntax	Type **:noheader**
:log	Function	Use this switch to produce a before and after listing. The output consists of an address and

a dump of the memory before and after the data was written. If you are poking data into an I/O device which MUST NOT be read, specify :nolog to disable reading the memory.

Default :log
Syntax Type **:nolog**

:long

Function Use this switch to specify that memory is accessed one longword (4 bytes) at a time. This is required to poke I/O devices which require longword accesses. The Data parameter should be a multiple of four. If it is not, not all of the specified bytes will be written. For example, if the Data parameter is 7, only 4 bytes will be written.

Default :byte
Syntax Type **:long**

NOTE: This switch is mutually exclusive with the :byte and :word switches.

:override

Function Use this switch to disable valid address checking. This is required to poke I/O addresses. Memory addresses are checked for validity before dumping. Physical memory addresses must be in the range $200000 through $200000+number-of-bytes of physical memory. For example, on a system with one megabyte of memory, valid physical memory addresses are $200000 through $2fffff. Logical memory addresses require the specification of a process with the :pid= switch and memory allocated to the given process at the logical addresses. If the logical or physical memory does not exist, POKE terminates with an error message. This switch overrides address checking with its error message and process termination.

Default :nooverride
Syntax Type **:override**

:pause

Function Use this switch to pause after writing every 128 data bytes. Pressing any key continues the data write.

Default :pause
Syntax Type **:nopause**

:pid=

Function: Use this switch to specify the process ID (PID) or process name. If the process does not exist, POKE will terminate with an error message. When a logical address is encountered, POKE translates it into a physical memory address using the values of the mapping registers of the given process. If a logical address is encountered and no process has been specified, POKE will terminate with an invalid physical address error.

Default: None.

Syntax: Type **:pid=** followed by a PID (default base is hexadecimal) or process name.

> NOTE: The process must be executing on the same system as POKE. A PID of 0 cannot be used.

:word

Function: Use this switch to specify that memory is accessed one word (2 bytes) at a time. This is required to poke I/O devices which require word accesses. The number of data bytes should be a multiple of two. If not, not all of the specified data bytes will be written, since only words are accessed. For example, if there are 7 data bytes, only 6 bytes will be written.

Default: :byte

Syntax: Type **:word**

> NOTE: This switch is mutually exclusive with the :byte and :long switches.

Examples

> **poke** 5ff004 "ff ff ff ff"

This command generates the following kind of display:

```
Address   BEFORE Poke Hexadecimal  Text    AFTER Poke Hexadecimal   Text
--------  -----------------------  ------  -----------------------  ------
005FF004  6C 64 73 20              lds     FF FF FF FF              ....
```

> **poke** 5ff004 4[ff]

This command produces the same result as the example above.

```
> poke *1004+4 "4[4[ff](This is poked data. )3[%10 %20 %30 %40]]"
                                        :base=*7f34-82   :pid=4b77
                                        :long
```

This command produces the following kind of display:

```
Address   BEFORE Poke Hexadecimal    Text      AFTER Poke Hexadecimal    Text
--------  -----------------------  --------   -----------------------  --------
00054210  2D 2D 2D 2D 2D 2D 2D 2D  --------   FF FF FF FF 54 68 69 73  ....This
00054218  2D 2D 2D 2D 2D 2D 2D 2D  --------   20 69 73 20 70 6F 6B 65   is poke
00054220  2D 2D 2D 2D 2D 2D 2D 2D  --------   64 20 64 61 74 61 2E 20  d data.
00054228  2D 2D 2D 2D 2D 2D 2D 2D  --------   0A 14 1E 28 0A 14 1E 28  ...(...(
00054230  2D 2D 2D 2D 2D 2D 2D 2D  --------   0A 14 1E 28 FF FF FF FF  ...(....
00054238  2D 2D 0A 32 37 30 30 30  --.27000   54 68 69 73 20 69 73 20  This is
00054240  20 20 20 20 20 20 20 09         .   70 6F 6B 65 64 20 64 61  poked da
00054248  09 09 3B 09 47 65 74 20  ..;.Get    74 61 2E 20 0A 14 1E 28  ta. ...(
00054250  74 68 65 20 61 64 64 72  the addr   0A 14 1E 28 0A 14 1E 28  ...(...(
00054258  65 73 73 20 6F 66 20 74  ess of t   FF FF FF FF 54 68 69 73  ....This
00054260  68 65 20 70 72 6F 63 65  he proce   20 69 73 20 70 6F 6B 65   is poke
00054268  73 73 20 74 61 62 6C 65  ss table   64 20 64 61 74 61 2E 20  d data.
00054270  20 61 6E 64 20 6C 6F 61   and loa   0A 14 1E 28 0A 14 1E 28  ...(...(
00054278  64 20 74 68 65 20 75 73  d the us   0A 14 1E 28 FF FF FF FF  ...(....
00054280  65 72 73 20 62 75 66 66  ers buff   54 68 69 73 20 69 73 20  This is
00054288  65 72 20 61 6E 64 0A 32  er and.2   70 6F 6B 65 64 20 64 61  poked da
*
00054290  37 30 30 31 20 20 20 20  7001       74 61 2E 20 0A 14 1E 28  ta. ...(
00054298  20 20 20 09 09 09 3B 09     ...;.   0A 14 1E 28 0A 14 1E 28  ...(...(
```

Note the pause after 128 bytes, indicated by the asterisk.

Using Prompts

```
> poke
Address        > 5ff004
Data           > 4[ff]
```

This command produces the same result as the first example.

Notes on Usage

POKE requires the user to have CHSUPER (change supervisor mode) privilege.

The starting address for the memory dump is computed in the following manner. Assume the Address parameter is "*40-8", the :base= switch has a value of "*2001d8+200", memory location "2001d8" has a value of "240000", and memory location "240240" has a value of "280000". The base address is computed first by going indirect through "2001d8" and retrieving the value "240000". The base offset "+200" is added to "240000" for a result of "240200". This is added to the memory address "40" for a result of "240240". This result is used as an indirect address to retrieve the value "280000". The address offset "-8" is added to "280000" for a starting address of "27fff8".

If you are poking data into I/O space which must not be read, specify :nolog to disable reading the memory. POKE reads the memory before writing the data in order to produce its "before" and "after" output.

Related CIP Commands

peek	Dump data from memory

prelink

Functional Description

Use this command to prepare libraries for linking.

Command Line Syntax

Mnemonic	prelink	
Required parameter	Filename	
Switches	:log	:pause

Parameters

Filename	Function	Required. Use this parameter to specify the name of the library definition file to be updated. These files can only reside on a disk.
	Default	None.
	Syntax	Type any valid file designation. No wildcard symbols are allowed. If no extension is specified, .DEF is added.

Switches

:log Function Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
Default The value specified by the OPTION command.
Syntax Type **:nolog** or **:log** to override the default.

:pause Function Use this switch to stop the display after each screen of information. The display resumes when you press any key.
Default The value specified by the OPTION Command.
Syntax Type **:pause** or **:nopause** to override the default.

Examples

> prelink prtlib.def

This command sets up the given library definition file. It logs each line in the file as it is updated in a display like this:

```
RRR000      SYS$DISK/PASCAL.PRT/INIT.MRL       _DS0//#7559.7565
RRR001      SYS$DISK/PASCAL.PRT/EXIT.MRL       _DS0//#3979.4616
RRR004      SYS$DISK/PASCAL.PRT/NEW.MRL        _DS0//#17544.17573
RRR006      SYS$DISK/PASCAL.PRT/MARK.MRL       _DS0//#17077.17133
RRR007      SYS$DISK/PASCAL.PRT/REL.MRL        _DS0//#17660.17712
RRR009      SYS$DISK/PASCAL.PRT/FBINIT.MRL     _DS0//#4520.4588
RRR010      SYS$DISK/PASCAL.PRT/RESET.MRL      _DS0//#17663.17684
RRR011      SYS$DISK/PASCAL.PRT/REWRITE.MRL    _DS0//#17665.17685
RRR012      SYS$DISK/PASCAL.PRT/APPEND.MRL     _DS0//#3205.3285
RRR013      SYS$DISK/PASCAL.PRT/RANDOM.MRL     _DS0//#17651.17667
RRR016      SYS$DISK/PASCAL.PRT/MBVALID.MRL    _DS0//#17541.17578
RRR017      SYS$DISK/PASCAL.PRT/MBVALID.MRL    _DS0//#17541.17578
RRR019      SYS$DISK/PASCAL.PRT/CLOSE.MRL      _DS0//#3623.3693
RRR019_     SYS$DISK/PASCAL.PRT/CLOSE.MRL      _DS0//#3623.3693
RRR020      SYS$DISK/PASCAL.PRT/MBVALID.MRL    _DS0//#17541.17578
RRR020_     SYS$DISK/PASCAL.PRT/MBVALID.MRL    _DS0//#17541.17578
```

PRELINK takes each filename and adds the FCB.SEQUENCE number to the end of the line. This helps the linker run faster. If a valid FCB.SEQUENCE number is present, the linker can open the file without searching through the directory structure. If the FCB.SEQUENCE number is invalid or not present, the linker uses the filename to open the file.

Using Prompts

```
> prelink
File name   > prtlib
```

This command performs the same function as the previous example.

Notes on Usage

Use PRELINK only on library definition files used by the WICAT LINK utility.

A .DEF file can refer to another .DEF file by listing the filename and preceding that filename with an at sign, @.

If an existing library module is replaced with an updated version, the linker loads the old version until either the old version is deleted, or until PRELINK reprocesses the library definition file.

Related CIP Commands

None.

print

Functional Description

Use this command to print text files on a printer, monitor the print queue, and perform other print-queue management functions.

Command Line Syntax

Mnemonic print

Required parameter
File List (Required only with :INSERT)
or Entry List (Required only with :WAKE, :HOLD,
:RESTART, :MODIFY, and :DELETE;
Optional with :LIST)

Optional parameter
Function (:CLOSE, :DELETE, :HALT, :HOLD, :INSERT,
:LIST, :MODIFY, :OPEN, :RESTART,
:START, :WAKE)

Switches

File selection	:before= :exclude=	:since= :confirm	:uic=	:mod
Form control	:copies= :formname= :lines= :fortran	:newline :pagehead :raw	:formatter= :header :formlength=	:number :pages= :formwidth=
Queue control	:entdelete :inform	:queue= :when=	:enthold :qpriority=	:remove
Other	:auto :sort=	:log :rewind	:full	:pause

Parameters

File List

Function

Required. Use this parameter with :INSERT to list the files to be printed.

For :WAKE, :HOLD, :RESTART, :MODIFY, and :DELETE, use this parameter to list the entry numbers to be acted upon.

For :LIST, use this optional parameter to specify the entry numbers to be listed.

For :CLOSE, :OPEN, :START, and :HALT this parameter is not used.

Default

None for :INSERT, :WAKE, :HOLD, :RESTART, :MODIFY, and :DELETE.

For :LIST the default is "1-" which is all entries.

Syntax

For :INSERT, type a list of filenames separated by commas. Wildcard symbols are allowed.

For :WAKE, :HOLD, :RESTART, :MODIFY, :DELETE, and :LIST functions, type a list of entry number ranges separated by commas.

Function

Function

Optional. Use this parameter to specify the action to be performed on the given queue.

These are the 11 functions:

:CLOSE closes the specified queue. No more entries may be inserted into that queue. All queued up entries continue to be processed. Operator privilege is required to perform this action.

:DELETE deletes a list of entries from the specified queue.

:HALT halts the specified queue. No more entries in the queue can be executed. Active entries continue to run. Entries can still be inserted into the queue, but they won't be executed. Operator privilege is required to perform this action.

:HOLD keeps a list of entries from executing. They maintain their position in the queue, but they are skipped over when their turn arrives.

:INSERT inserts a list of files into the specified queue. This is the default if no action is specified.

:LIST displays the entries in a given queue. It also lists their status.

:MODIFY modifies a list of entries in a given queue. Any parameters specified when an entry is inserted can be modified later with this command.

:OPEN is the opposite of :CLOSE. It allows new entries to be entered into the queue. Operator privilege is required to perform this action.

:RESTART restarts a list of active entries. The given processes are terminated and then restarted. If you :HALT a queue and then :RESTART, all activity in the queue including active entries) is stopped.

:START is the opposite of :HALT. It allows entries to be executed again. Operator privilege is required to perform this action.

:WAKE is the opposite of :HOLD. It allows a list of held entries to be executed.

Default :INSERT
Syntax Type one of the 11 actions listed above.

Switches

:auto	Function	Use this switch to perform the desired operation without any confirmation.
	Default	:noauto
	Syntax	Type :auto
:before=	Function	Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.

Default All files that match the File List parameter are selected.

Syntax Type **:before=** followed by a date or time in the standard date and time syntax.

:confirm

Function Use this switch to confirm or deny the desired operation on each file.

Default :noconfirm

Syntax Type **:confirm**

:copies=

Function Use this switch to specify how many copies of the file are to be printed. This switch is only valid with the :INSERT and :MODIFY Functions.

Default :copies=1

Syntax Type **:copies=** followed by any valid integer.

:entdelete

Function Use this switch to specify whether the given entry is to be deleted from the queue when the entry has completed execution. If the entry is not deleted, it is placed in a hold state.

Default :entdelete

Syntax Type **:noentdelete**

:enthold

Function Use this switch to specify whether the given entry should be placed in a hold state immediately upon entry into the queue. This means it is impossible for the entry to be executed before it is awakened.

Default :noenthold

Syntax Type **:enthold**

:exclude=

Function Use this switch to exclude files from the File List parameter.

Default All files that match the File List parameter are selected.

Syntax Type **:exclude=** followed by a list of file designations, separated by commas, any one of which may contain wildcard characters.

:formatter=

Function Use this switch to specify what formatter program to use when this entry is executed.

Default :formatter=SYS$DISK/SYSEXE/QPRINT.EXE.

Syntax Type **:formatter=** followed by any valid file designation. Wildcard symbols are not allowed.

:formlength=	Function	Use this switch to specify the number of lines on each printed page. As the page is printed, a FORMFEED is inserted into the text after this many lines.
	Default	The form length specified by the DSTAT :length= switch.
	Syntax	Type **:formlength=** followed by any positive integer. Type a zero if you do not want any FORMFEEDS inserted at all.
		NOTE: If a header page is to be printed, it will contain this many lines unless a zero is specified. If a zero is specified, it assumes a 60 for this value.
:formname=	Function	Use this switch to specify the type of forms to be used when this entry is executed. If the form name for the queue (as specified by the DSTAT :formname= switch) does not match exactly, the entry is placed in a FORMWAIT state. After the correct forms are loaded and the form name for the queue is updated with the DSTAT command, the entry can be executed with :WAKE.
	Default	:formname=none
	Syntax	Type **:formname=** followed by any valid string up to 16 characters long.
:formwidth=	Function	Use this switch to specify the length of the lines on a printed page. If a line that is to be printed is longer than this value, it is truncated to this value.
	Default	The form width specified by the DSTAT :width= switch.
	Syntax	Type **:formwidth=** followed by any positive integer. Specify a zero if you do not want any lines to be truncated.
		NOTE: If a header page is to be printed, it will not be any wider than this number unless a zero is specified. If a zero is specified, it assumes an 80 for this value.

:fortran	Function	Use this switch to strip carriage control characters from files created by redirecting a FORTRAN program's output to a file.
	Default	:nofortran
	Syntax	Type **:fortran**
:full	Function	Use this switch to display all information about the file.
	Default	:nofull
	Syntax	Type **:full**
:header	Function	Use this switch to print a header page before each file (and each copy of a file) is printed. The header contains the filename, the time the file was printed, the username, the name of the owner of the file, etc.
	Default	:header
	Syntax	Type **:noheader**
:inform	Function	Use this switch to send a message to the queuer that the entry has been completed.
	Default	:noinform
	Syntax	Type **:inform**
:lines=	Function	Use this switch to specify the lines to be printed. If the :pages= switch is specified, it overrides any values specified here.
	Default	All lines are printed.
	Syntax	Type **:lines=** followed by a range of integers. Line numbering begins at 1.
:log	Function	Use this switch to specify whether log messages are displayed.
	Default	The value specifed by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**

:newline	Function	Use this switch to specify whether linefeed characters are converted into a carriage return-linefeed sequence before being printed.
	Default	:newline
	Syntax	Type **:nonewline**
:number	Function	Use this switch to specify that a line number is to be added at the left margin of each line that is printed.
	Default	:nonumber
	Syntax	Type **:number**
:pagehead	Function	Use this switch to print a header at the top of each page. This header contains the page number, the name of the file, and the current time. Two blanks lines are printed before the normal text of that page is printed.
	Default	:nopagehead
	Syntax	Type **:pagehead**
:pages=	Function	Use this switch to specify the pages to be printed.
	Default	All pages are printed.
	Syntax	Type **:pages=** followed by a range of integers. Page numbering begins at 1.
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes after the user presses any character.
	Default	The value specified by the OPTION command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:qpriority=	Function	Use this switch to specify the priority for the entry to be inserted.
	Default	The base priority of the queue as specified by the DSTAT :baseprior= switch.
	Syntax	Type **:qpriority=** followed by an integer between 0 and 15, with 0 as the highest priority and 15 as the lowest.
:queue=	Function	Use this switch to specify the queue this function is to operate on.

	Default	:queue=SYS$PRINT
	Syntax	Type **:queue=** followed by the devicename of a QUEUE class device whose type is PRINTER. No wildcard characters are allowed.
:raw	Function	Use this switch to specify that the data being written are not to be modified or examined. This switch is necessary when printing graphics-type data to a printer.
	Default	:noraw
	Syntax	Type **:raw**
:remove	Function	Use this switch to delete the file being printed after it is printed.
	Default	:noremove
	Syntax	Type **:remove**
:rewind	Function	Use this switch to rewind the tape before file search.
	Default	:norewind
	Syntax	Type **:rewind**
:since=	Function	Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:since=** followed by a date or time in the standard syntax.
:sort=	Function	Use this switch to specify the order in which the files are accessed; files may be accessed in alphabetical order by filename, by file extension, in UIC order, or by creation date (oldest file first).
	Default	Files are accessed in alphabetical order by filename.
	Syntax	Type **:sort=** followed by one of the following keywords: **extension**, **uic**, or **date**.
:uic=	Function	Use this switch to select only those files that are part of the given list and are owned by the specified user or list of users.

Default — All files that match the specified list are selected.

Syntax — Type **:uic=** followed by a list of UICs or usernames.

:when=

Function — Use this switch to specify when this entry is to be entered into the queue and available for execution.

Default — :when=+0

Syntax — Type **:when=** followed by a date or time in the standard date and time syntax.

Examples

> print abc.dat

This command inserts the file ABC.DAT into the queue SYS$PRINT for printing. Then the name of the queue and the file's entry number appear on the screen:

```
Queue _pq0
_DS0/USERS/ABC.DAT.1 queued as entry 44
```

> print *.dat :pagehead :number

This command prints all files with the extension .DAT in the default directory. Each file is printed with a header at the top of each page and a line number in the left margin. PRINT also logs each file and its entry number as shown:

```
Queue _pq0
_DS0/USERS/A.DAT.5 queued as entry 53
_DS0/USERS/ABC.DAT.1 queued as entry 54
_DS0/USERS/B.DAT.1 queued as entry 55
_DS0/USERS/CIP.DAT.2 queued as entry 56
_DS0/USERS/CMDST.DAT.2 queued as entry 57
_DS0/USERS/DSTAT.DAT.5 queued as entry 58
_DS0/USERS/OPTION.DAT.1 queued as entry 59
_DS0/USERS/PRINT.DAT.2 queued as entry 60
```

> print :list

This command generates a display like the following:

```
Queue _PQ0
Entry Pri Username      Entry Name                   Status
----- --- --------  -----------------  ------------------------------
   65  7  USER      CMDST.DAT.2        Active, 55% printed, copy 1 of 1
   66  7  USER      DSTAT.DAT.5        Pending
   67  7  USER      OPTION.DAT.1       Pending
   68  7  USER      PRINT.DAT.2        Pending
```

Note that the percentage of the file being printed, along with which copy is being printed, is displayed.

> print :list :full

This performs the same function as the previous example, but with a full display like the following:

```
Queue _PQ0

_DS0/USERS/CMDST.DAT.2
Entry     : 73       Priority  : 7       Username  : USER
File size : 1.9      FormLength: 60      Entry name: CMDST.DAT.2
Copies    : 1        FormWidth : 132     Form name : None
Pages     : All      Lines     : All     Header
NoInform    Newline  NoNumber    NoPagehead NoRemove   NoRaw
Active, 55% printed, copy 1 of 1

_DS0/USERS/DSTAT.DAT.5
Entry     : 74       Priority  : 7       Username  : USER
File size : 1.4      FormLength: 60      Entry name: DSTAT.DAT.5
Copies    : 1        FormWidth : 132     Form name : None
Pages     : All      Lines     : All     Header
NoInform    Newline  NoNumber    NoPagehead NoRemove   NoRaw
Pending

_DS0/USERS/OPTION.DAT.1
Entry     : 75       Priority  : 7       Username  : USER
File size : 0.3      FormLength: 60      Entry name: OPTION.DAT.1
Copies    : 1        FormWidth : 132     Form name : None
Pages     : All      Lines     : All     Header
NoInform    Newline  NoNumber    NoPagehead NoRemove   NoRaw
Pending

_DS0/USERS/PRINT.DAT.6
Entry     : 76       Priority  : 7       Username  : USER
File size : 0.4      FormLength: 60      Entry name: PRINT.DAT.6
Copies    : 1        FormWidth : 132     Form name : None
Pages     : All      Lines     : All     Header
NoInform    Newline  NoNumber    NoPagehead NoRemove   NoRaw
Pending
```

> print :delete 1- :auto

This command deletes all entries in the queue automatically (without confirmation) and logs each entry as it is deleted:

```
Queue _pq0
  Entry 85 _DS0/USERS.WORKBENCH.PROJECTS.180101201.D/A.DAT.5 Deleted.
  Entry 86 _DS0/USERS.WORKBENCH.PROJECTS.180101201.D/ABC.DAT.1 Deleted.
  Entry 87 _DS0/USERS.WORKBENCH.PROJECTS.180101201.D/B.DAT.1 Deleted.
  Entry 88 _DS0/USERS.WORKBENCH.PROJECTS.180101201.D/CIP.DAT.2 Deleted.
  Entry 89 _DS0/USERS.WORKBENCH.PROJECTS.180101201.D/CMDST.DAT.2 Deleted.
  Entry 90 _DS0/USERS.WORKBENCH.PROJECTS.180101201.D/DSTAT.DAT.5 Deleted.
  Entry 91 _DS0/USERS.WORKBENCH.PROJECTS.180101201.D/OPTION.DAT.1 Deleted.
  Entry 92 _DS0/USERS.WORKBENCH.PROJECTS.180101201.D/PRINT.DAT.11 Deleted.
```

> print 203 :modify :qpriority=10 :copies=3 :when=tomorrow

This command modifies an entry that is already entered. It changes its priority, the number of copies to print, and when it is to be printed. Note that any parameter or switch specified at the time the entry is printed can be modified. PRINT logs each entry as it is modified:

```
Queue _pq0
Entry 203 _DS0/USERS/OPTION.DAT.1 Modified.
```

> print :close :queue=_pq1

This command closes the queue _PQ1. Any user trying to add more files to this queue receives this error message:

```
Queue _PQ1 closed.
```

Note that if you list the entries of a queue after it has been closed or halted this kind of display appears:

```
Queue _PQ1  Closed
Entry Pri Username     Entry Name                         Status
----- --- --------  ------------------  ---------------------------------
    4  7  USER      A.DAT.5             Pending
    5  7  USER      ABC.DAT.1           Pending
    6  7  USER      B.DAT.1             Pending
    7  7  USER      CIP.DAT.2           Pending
    8  7  USER      CMDST.DAT.2         Pending
    9  7  USER      DSTAT.DAT.5         Pending
   10  7  USER      OPTION.DAT.1        Pending
   11  7  USER      PRINT.DAT.14        Pending
```

Using Prompts

```
> print
File list     > abc.dat
```

This performs the same function as the first example.

```
> print :hold :queue=_pq2
Entry list    > 25-27
```

The :HOLD function causes a new prompt to appear. The user is prompted for a list of entry numbers to be held from the specified queue. Entries 25, 26 and 27 will not be executed.

Notes on Usage

Of the 11 actions that may be specified, only one can be executed at a time.

It is possible to have two printers assigned to the same queue. In this case, the system automatically selects the first free printer to use. First, a rotor list must be set up with the ROTOR command. For example:

```
     > rotor pqrot _tt2,_tt3
```

The devices _TT2 and _TT3 have the printers on them. Second, the rotor list name must be specified as the associative device for the print queue. Third, the number of active jobs must be specified. For example:

```
     > dstat _pq2 :assocdev=pqrot :numactive=2
```

A rotorable print queue is now set up. The system routes the next job to print on the first free printer. With this command string the user can specify many printers on the same queue at the same time. However, the :NUMACTIVE Switch must be set to the same number of devices in the rotor list.

The following is an example of a print header page:

```
********  __BARTLEBY_DS0/USERS.NEAL/ABC.DAT.1   14-Jan-1985 19:28:20  ********
********  __BARTLEBY_DS0/USERS.NEAL/ABC.DAT.1   14-Jan-1985 19:28:20  ********
********  __BARTLEBY_DS0/USERS.NEAL/ABC.DAT.1   14-Jan-1985 19:28:20  ********


   A    BBBBBB   CCCCC         DDDDDD     A    TTTTTTT          1
  A A   B     B C     C        D     D   A A      T           111
 A   A  B     B C              D     D  A   A     T             1
A     A BBBBBB  C              D     D A     A    T             1
AAAAAAA B     B C              D     D AAAAAAA    T             1
A     A B     B C     C  ...   D     D A     A    T     ...     1
A     A BBBBBB   CCCCC   ...   DDDDDD  A     A    T     ...   11111


N     N EEEEEEE    A    L
NN    N E         A A   L
N N   N E        A   A  L
N  N  N EEEEE   A     A L
N   N N E       AAAAAAA L
N    NN E       A     A L
N     N EEEEEEE A     A LLLLLLL



                         File    : __BARTLEBY_DS0/USERS.NEAL/ABC.DAT.1
                         SysName : Bartleby
                         Owner   : NEAL
                         Entry   : 45
                         Created : 14-Jan-1985 18:30:44
                         Queued  : 14-Jan-1985 19:28:19
                         Printed : 14-Jan-1985 19:28:20



********  __BARTLEBY_DS0/USERS.NEAL/ABC.DAT.1   14-Jan-1985 19:28:20  ********
********  __BARTLEBY_DS0/USERS.NEAL/ABC.DAT.1   14-Jan-1985 19:28:20  ********
********  __BARTLEBY_DS0/USERS.NEAL/ABC.DAT.1   14-Jan-1985 19:28:20  ********
```

The three lines at the top and bottom of the page show the the filename and the printing time. These lines are padded out to the specified width of the printer with asterisks. The width can be specified for each entry with the :formwidth= switch, or for the device with the DSTAT :width= switch.

The filename is printed in large block letters, followed by the username in block letters.

The block of information near the bottom of the page contains the file's complete pathname, the system name, the owner's name, the file's entry number in the queue, and the times the file was created, queued up, and printed.

Related CIP Commands

submit	Submit processes for batch execution in background mode.

prompt

Functional Description

Use this command in a command file to display information to the user and to request user input.

Command Line Syntax

Mnemonic	prompt		
Optional parameter	Message		
Switches	:logical=	:newline	:response

Parameters

Message	Function	Optional. Use this parameter to specify the message to be displayed to the user.
	Default	A blank line.
	Syntax	Type a message enclosed in double quotation marks. Use accept sequences to embed control characters into the message.

Switches

:logical=	Function	Use this switch to specify a logical name to be assigned the value typed by the user.
	Default	:logical=response

Syntax Type **:logical=** followed by a logical name.

:newline

Function Use this switch to suppress the display of a new line after the message.
Default :newline
Syntax Type **:nonewline** to keep the cursor at the end of the message instead of moving to the front of the next line.

> NOTE: If the :noresponse switch is not specified, this switch is forced to :nonewline, regardless of the value specified by the user.

:response

Function Use this switch to display the message without waiting for user response.
Default :response
Syntax Type **:noresponse**

Examples

```
> prompt "Press RETURN to continue"
```

This command displays the message on the terminal screen, waits for the user's response, and assigns the response to the logical name RESPONSE.

```
> prompt "Type the name of your Alma Mater" :logical=almamater
> prompt "Is 'almamater' a better school than, say, Harvard?"
```

The first line prompts the user to type in the name of a school, then assigns the user's response to the logical name "almamater".

The second line uses that logical name to request further information from the user.

```
> prompt "Mounting _dx0 . . . " :nonewline :noresponse
> mnt _dx0 :nolog
> prompt "Successful" :noresponse
```

The first line displays a message on the screen, does not wait for user response, and does not print a new line.

The third line completes the display, does not wait for user response, but does move to the front of the next line.

Using Prompts

None.

Notes on Usage

This command is most useful in command files where it is desirable to interrupt the processes being initiated by the file so the user can either initiate a process or perform some function necessary to the execution of the command file.

Related CIP Commands

None.

pstat

Functional Description

Use this command to display or set the attributes of processes.

Command Line Syntax

Mnemonic	pstat			
Optional parameter	Process List			
Switches				
Process to be included	:auto	:confirm	:uic	:level=
Format of the display	:full	:log	:header	:pause
Content of the display	:alarm	:default	:child	:devices
	:eventflags	:exithandler	:files	:imagefile
	:ksam	:mail	:names	:owner
	:parent	:port	:priority	:privilege
	:processstatus	:protection	:puresize	:scheduled
	:sharedmemory	:size	:state	:status
	:systemstatus	:timeslice	:traps	:username
	:fltpoint	:timeout	:attribute	:hibcount
Attributes to be set	:eventflags=	:hibernate	:owner=	:kill
	:priority=	:privilege=	:processname=	:timeslice=
	:wake	:wakec	:attribute=	

Parameters

Process List	Function	Optional. Use this parameter to specify the processes you wish to modify.
	Default	The process ID (PID) and process name of each process on the system is displayed. If switches for process attributes are specified, the default is the user's process attributes (i.e., PSTAT's parent process).
	Syntax	Type a list of process names or PIDs. Wildcard symbols are allowed in process names.

Switches

:alarm	Function	Use this switch to display the alarm time set for a process.
	Default	:noalarm
	Syntax	Type **:alarm**
:attribute	Function	Use this switch to display the process attributes for the process.
	Default	:noattribute
	Syntax	Type **:attribute**
:attribute=	Function	Use this switch to set the attributes for a process.
	Default	Attributes assigned to the process when it was created.
	Syntax	Type **:attribute=** followed by a list of process attributes separated by commas.

The possible attributes are:

swappable	Enable swapping
prezeromem	Zero allocated memory before use
postzeromem	Zero allocated memory after use
fastencrypt	Use fast encryption on the network
desencrypt	Use DES encryption on the network
watchdog	Allow watchdog to kill a CIP
user1	Set user attribute bit 1
user2	Set user attribute bit 2
user3	Set user attribute bit 3
user4	Set user attribute bit 4

:auto	Function	Use this switch to perform the desired operation without any confirmation.
	Default	:noauto
	Syntax	Type **:auto**
:child	Function	Use this switch to display the PID of the child of a process.
	Default	:nochild
	Syntax	Type **:child**
:confirm	Function	Use this switch to confirm or deny the desired action on each process.
	Default	:noconfirm
	Syntax	Type **:confirm**
:default	Function	Use this switch to display the default device and directory of a process.
	Default	:nodefault
	Syntax	Type **:default**
:devices	Function	Use this switch to display the devices allocated to a process.
	Default	:nodevices
	Syntax	Type **:devices**
:eventflags	Function	Use this switch to display the event flags mask of a process.

Default :noeventflags
Syntax Type **:eventflags**

:eventflags=
Function Use this switch to set the event flag mask of a process.
Default The event flag mask assigned when the process was created.
Syntax Type **:eventflags=** followed by an 8-digit hexadecimal number (representing the event flag bits).

:exithandler
Function Use this switch to display the address of the exit handler for this process.
Default :noexithandler
Syntax Type **:exithandler**

:files
Function Use this switch to display the open files of a process.
Default :nofiles
Syntax Type **:files**

:fltpoint
Function Use this switch to display the type of floating point hardware being used by a process.
Default :nofltpoint
Syntax Type **:fltpoint**

:full
Function Use this switch to display all information about the file, device, or process.
Default :nofull
Syntax Type **:full**

:header
Function Use this switch to display column headers.
Default :header
Syntax Type **:noheader**

:hibcount
Function Use this switch to display the number of times the process has been hibernated.
Default :nohibcount
Syntax Type **:hibcount**

:hibernate
Function Use this switch to hibernate the specified process.

Default :nohibernate
Syntax Type **:hibernate**

:imagefile
Function Use this switch to display the image file from which the process was created.
Default :noimagefile
Syntax Type **:imagefile**

:kill
Function Use this switch to kill the process.
Default :nokill
Syntax Type **:kill**

:ksam
Function Use this switch to display the open KSAM files of a process.
Default :noksam
Syntax Type **:ksam**

:level=
Function Use this switch to select only those processes that match the Process List parameter and have the specified priority level.
Default All processes that match the Process List parameter are selected.
Syntax Type **:level=** followed by a number from 0 to 15.

:log
Function Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
Default :log
Syntax Type **:nolog**

:mail
Function Use this switch to display whether there is mail pending for a process.
Default :nomail
Syntax Type **:mail**

:names
Function Use this switch to display only the names of files and shared memory regions.
Default :nonames
Syntax Type **:names**

:owner	Function	Use this switch to display the UIC of the owner of each process.
	Default	:noowner
	Syntax	Type **:owner**
:owner=	Function	Use this switch to change the owner (UIC) of a process.
	Default	The owner who created the process.
	Syntax	Type **:owner=** followed by a UIC or a username. (Usernames are converted to the UIC they represent.)
:parent	Function	Use this switch to display the PID of the parent of a process.
	Default	:noparent
	Syntax	Type **:parent**
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when you press any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:port	Function	Use this switch to display the port a process is running on.
	Default	:noport
	Syntax	Type **:port**
:priority	Function	Use this switch to display the scheduling priority of the process. The priority is displayed in the form "current/base", where current is the priority the process has floated to and base is the priority assigned when the process was created.
	Default	:nopriority
	Syntax	Type **:priority**
:priority=	Function	Use this switch to set the base priority of a process.
	Default	The priority assigned when the process was created.
	Syntax	Type **:priority=** followed by a number from 0 to 15. Zero is the highest priority.

:privilege	Function	Use this switch to display the privileges of a process.
	Default	:noprivilege
	Syntax	Type **:privilege**
:privilege=	Function	Use this switch to set the privileges of a process.
	Default	The privileges assigned when the process was created.
	Syntax	Type **:privilege=** followed by a list of privileges separated by commas.

The possible privileges are:

setpriv	Allows process to assign any other privilege
system	Allows system access to resources
readphys	Allows physical read access to resources
writephys	Allows physical write access to resources
setprior	Allows process to set higher priority, or go realtime
chngsuper	Allows process to change to supervisor mode
bypass	Allows process to bypass file and device protection
operator	Allows process to perform operator functions
altuic	Allows process access to files as though it had the UIC of the owner of the image file
world	Allows process to affect any other process in the system
group	Allows process to affect processes with same group ID process owner
network	Allows process to go remote
setattr	Allows process to set any process attributes

:processname=	Function	Use this switch to set the process name.
	Default	The name assigned when the process was created.
	Syntax	Type **:processname=** followed by a name up to 16 characters long.

:processstatus	Function	Use this switch to list the process status bits that are set for the process.
	Default	:noprocessstatus
	Syntax	Type **:processstatus**
:protection	Function	Use this switch to display the protection mask for each process.
	Default	:noprotection
	Syntax	Type **:protection**
:puresize	Function	Use this switch to display the sizes of the pure and impure parts of a process.
	Default	:nopuresize
	Syntax	Type **:puresize**
:scheduled	Function	Use this switch to display the number of times a process has been scheduled.
	Default	:noscheduled
	Syntax	Type **:scheduled**
:sharedmemory	Function	Use this switch to display named shared memory regions of a process.
	Default	:nosharedmemory
	Syntax	Type **:sharedmemory**
:size	Function	Use this switch to display the total size of the process.
	Default	:nosize
	Syntax	Type **:size**
:state	Function	Use this switch to display the saved state (registers) of a process.
	Default	:nostate
	Syntax	Type **:state**
:status	Function	Use this switch to display the status of a process.
	Default	:nostatus
	Syntax	Type **:status**
:systemstatus	Function	Use this switch to display the status (time, up time, and free memory) of the system.

	Default	:nosystemstatus
	Syntax	Type **:systemstatus**
:timeout	Function	Use this switch to display the current timeout for a process.
	Default	:notimeout
	Syntax	Type **:timeout**
:timeslice	Function	Use this switch to display the timeslice of a process.
	Default	:notimeslice
	Syntax	Type **:timeslice**
:timeslice=	Function	Use this switch to set the timeslice of a process in milliseconds.
	Default	The timeslice assigned when the process was created.
	Syntax	Type **:timeslice=** followed by a positive integer.
:traps	Function	Use this switch to display the traps defined for a process.
	Default	:notraps
	Syntax	Type **:traps**
:uic=	Function	Use this switch to select only those processes that match the Process List parameter and are owned by the specified user or list of users.
	Default	All processes that match the Process List parameter are selected, regardless of ownership.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.
:username	Function	Use this switch to display the username of the owner of each process.
	Default	:nousername
	Syntax	Type **:username**
:wake	Function	Use this switch to wake the process immediately.
	Default	:nowake

Syntax Type **:wake**

:wakec Function Use this switch to gradually decrease the hibernate count on a process and wake it if this count goes to zero.
Default :nowakec
Syntax Type **:wakec**

Examples

> **pstat**

This command generates a display like the following:

```
   PID      Process Name
--------  ---------------
00010001  Logflush
00010040  Watchdog
00010002  Que_Manager
00010004  Swapper
0001043E  vew_KARL
0001041C  vew_BILL
000104C2  vew_GRACE
000101AB  CIP_GRACE
0001041E  CIP_KARL
000103F6  CIP_BILL
```

The first column contains the PID. This is the number used within the system to uniquely identify the process. PIDs are expressed in hexadecimal digits.

The second column contains the process name. The process name is typically composed of the image filename (or the command filename, denoted by @) followed by an underscore, _, and the username. The process name does not uniquely identify the process because two processes can have the same name.

```
> pstat :systemstatus :port :status :size :priority :scheduled
:timeslice
```

This command generates the following kind of display:

```
Time:     23-May-1985 15:00:30.22                           Free memory: 388K
Up time:            0 06:54:54.24
  PID       Process Name        Port          Status   Size Prior Schedule Times
--------  --------------  ----------------  --------  ----  ----  --------  ----
00010001  Logflush        __BARTLEBY_TT0    Wait         4  2/2    129155   100
00010040  Watchdog        __BARTLEBY_TT0    Wait        56  2/2       566   100
00010002  Que_Manager                       Wait        40  2/2      4188   100
00010004  Swapper         __BARTLEBY_TT0    Wait        56  2/2       181   100
0001043E  vew_KARL        __BARTLEBY_10KAR  Wait       140  5/7      6379    60
0001041C  vew_BILL        __BARTLEBY_1BILL  Wait        60  5/7      1941    60
000104F6  CIP_VEW         __BARTLEBY_7GRAC  ChildWait   16  7/7       109    60
000104C2  vew_GRACE       __BARTLEBY_7GRAC  ChildWait   80  7/7      1567    60
000101AB  CIP_GRACE       __BARTLEBY_7GRAC  ChildWait   64  7/7       452    60
0001041E  CIP_KARL        __BARTLEBY_10KAR  ChildWait   16  7/7       174    60
000103F6  CIP_BILL        __BARTLEBY_1BILL  ChildWait   16  7/7       218    60
```

The :systemstatus switch prints the top two lines of the display. "Time" is the system absolute date and time. "Up time" is the elapsed time since the last system boot. "Free memory" is the amount of memory still available, expressed in kilobytes.

The columns (other than PID and Process Name) are printed by the switches corresponding to the column headings. For example, the :priority switch causes the Prior column to appear in the display. The columns appear in the order the switches are specified on the command line (following the PID and Process Name columns).

> pstat 0 :full

This command generates a display like the following:

```
CIP_VEW
    PID         : 00010514    Priority  :   7/7   Port      : __BARTLEBY_7GRAC
    Child       : 00010516    Timeslice :    60   Username  : GRACE
    Parent      : 000104C2    Size      :    16   Status    : ChildWait
    Exithandler : None        Pure      :    48   Owner     : [0004,0002]
    Eventflags  : 00000000    Impure    :    16   Scheduled :           47
    Mail        : None        Hib Count :     0   Fltpoint  : None
    Alarm       : None
    Timeout     : None
    Protection  : S:DWRE,P:    ,G:  RE,O:DWRE
    Default:  __BARTLEBY_DS0/USERS.WORKBENCH.PROJECTS.180101201.E/
    Process Status:
        controlcprot
    Privilege:
        NOSETPRIV      NOSYSTEM       NOREADPHYS     NOWRITEPHYS
        NOSETPRIOR     NOCHNGSUPER    NOBYPASS       NOOPERATOR
        NOALTUIC         WORLD          GROUP        NONETWORK
        NOSETATTR
    Attribute:
          SWAPPABLE    NOPREZEROMEM   NOPOSTZEROMEM  NODESENCRYPT
        NOFASTENCRYPT    WATCHDOG
        NOUSER1        NOUSER2        NOUSER3        NOUSER4
    State:
        System stack pointer = 0032FEA6
        User stack pointer   = 001FE4C4
Registers      0        1        2        3        4        5        6        7
Data    00000007 0000000F 00000000 00050AAE 000001B8 00000600 FFFFFFFF 00000000
Address 0032F000 0034201C 001FE5A3 001FE514 00342000 0032F000 0032FEF2 0032FEA6
        Program counter = 00201984                     Status register = 2704
    Image File: __BARTLEBY_DS0/SYSEXE/CIP.EXE.1
        Owner: [0001,0001]     Username: SYSTEM
    File: __BARTLEBY_7GRACE
        Pos:           0       Lun:           1
        Access:
            writeaccess      openshared
    File: __BARTLEBY_7GRACE
        Pos:           0       Lun:           2
        Access:
            writeaccess      openshared
    File: __BARTLEBY_7GRACE
        Pos:           0       Lun:           3
        Access:
            readaccess       openshared
```

The following is an explanation of the lines in the foregoing display:

Line 1 contains the process name, CIP_VEW.

Line 2 contains the PID, Priority, and Port.

Line 3 contains the PID of the child of this process, the timeslice, and the Username. If the process had no child, the Child field would read "None".

Line 4 contains the PID of the parent of this process, the Size, and Status. If the process had no parent, the Parent field would read "None".

Line 5 contains the address of the exit handler routine, the size of the Pure (shared) code in kilobytes, and the Owner. Because this process has no exit handler, the Exithandler field reads "None".

Line 6 contains the event flag mask, the size of the Impure (non-shared) code in kilobytes, and the number of times this process has been Scheduled.

```
Mail        : None        Hib Count :     0    Fltpoint  : None
Alarm       : None
Timeout     : None
Protection  : S:DWRE,P:      ,G:  RE,O:DWRE
Default:  __BARTLEBY_DS0/USERS.WORKBENCH.PROJECTS.180101201.E/
Process Status:
    controlcprot
```

Line 7 contains the Mail, Hibernation Count, and Floating Point for the process.

Line 8 contains the time for which an Alarm is set for this process. Because there is no alarm set, it reads "None".

Line 9 contains the Timeout for the process.

Line 10 contains the Protection mask for the process.

Line 11 contains the Default device and directory for the process.

The Process Status field begins on Line 12. It contains a list of the process status bits set for the process. The possible status bits are:

abort	The process is set to abort
abortinprg	The process is aborting
alarmset	An alarm is set for the process
allswapped	The process is entirely swapped out
cleanup	The OS is cleaning up after a process has terminated
controlcprot	The process cannot be killed by [CTRL] c
creatingprcs	The process is creating another process
errordisplayed	An error has been reported for this process
errreporting	The process is outputting a crash display (i.e., an error is being reported for the process)
exithandinprg	The process's exit handler is executing
extendingfcb	The FCB file is being extended
fltpointpend	A floating point exception is pending for this process
inqueue	The process is waiting in a queue
ksamaccess	The process is executing a KSAM operation
logbadsect	The process is marking a bad sector
multcreateproc	Multiple processes are being created
nocrashdisp	Errors (crash displays) are suppressed for this process
realtime	The process is a realtime process
supervisormode	The process is executing in supervisor mode
surrogate	This is a surrogate process (applies to NSP's only)
surrogatechild	This is a surrogate child process (child of an NSP's process)
swapped	The process has swapped pages
tohibernate	The process is in OS waiting to be hibernated
trace	This process is tracing

```
Privilege:
    NOSETPRIV        NOSYSTEM         NOREADPHYS       NOWRITEPHYS
    NOSETPRIOR       NOCHNGSUPER      NOBYPASS         NOOPERATOR
    NOALTUIC           WORLD            GROUP          NONETWORK
    NOSETATTR
Attribute:
      SWAPPABLE      NOPREZEROMEM     NOPOSTZEROMEM    NODESENCRYPT
    NOFASTENCRYPT      WATCHDOG
    NOUSER1          NOUSER2          NOUSER3          NOUSER4
```

The Privilege field is a list of process privileges. If the privilege name has "NO" in front of it, the process does not have that privilege.

The Attribute field is a list of process attributes. If the attribute name has "NO" in front of it, the process does not have that attribute.

```
        State:
            System stack pointer = 0032FEA6
            User stack pointer   = 001FE4C4
    Registers     0        1        2        3        4        5        6        7
    Data     00000007 0000000F 00000000 00050AAE 000001B8 00000600 FFFFFFFF 00000000
    Address  0032F000 0034201C 001FE5A3 001FE514 00342000 0032F000 0032FEF2 0032FEA6
            Program counter = 00201984                     Status register = 2704
```

The State field contains the state of the process as saved when it was scheduled out. This field is composed of the system stack pointer, user stack pointer, data registers D0 - D7, address registers A0 - A7 (note that A7 is the same as the stack pointer the process was using when it was scheduled out), program counter, and the status register. The values of these are displayed in hexadecimal digits.

```
    Image File: __BARTLEBY_DS0/SYSEXE/CIP.EXE.1
        Owner: [0001,0001]     Username: SYSTEM
```

The Image File field contains the name of the image file from which the process was created.

```
  File: __BARTLEBY_7GRACE
      Pos:          0       Lun:         1
      Access:
          writeaccess     openshared
  File: __BARTLEBY_7GRACE
      Pos:          0       Lun:         2
      Access:
          writeaccess     openshared
  File: __BARTLEBY_7GRACE
      Pos:          0       Lun:         3
      Access:
          readaccess      openshared
```

The File field contains each file the process has open at the time. For each open file the following is displayed:

Filename	
pos	Current position in the file (in bytes)
lun	Logical unit number of the file
access	A list of the access modes the file is open with

Possible access modes are:

readaccess	Read access
writeaccess	Write access
readlock	Read locked
writelock	Write locked
delete	Delete upon closing
append	Append to the file
fastread	Do fast read
opennext	Open next file
noreadahead	Do not do readahead
notruncfile	Do not truncate back the file
openifthere	Create the file if necessary
openshared	Share the parent process' default position in the file

> pstat 0 :privilege=group

This command adds group privilege to your process (i.e. to the process which is PSTAT's parent) and displays the following kind of log message:

```
00010202 CIP_VEW Altered.
```

> pstat *dir* :kill

This command lists all the processes with DIR in their process names and asks for confirmation before killing them. If you type Y they are killed and log messages like the following are displayed:

```
    00010237 dir_GRACE
    0001023A dir_GRACE
  Kill (Y or N)? > y
    00010237 dir_GRACE Killed.
    0001023A dir_GRACE Killed.
```

Using Prompts

None.

Notes on Usage

In general, you can display any attribute of any process, but State, Files, KSAM, Shared Memory, and Devices are considered to be "sensitive" attributes. To display these attributes you must own (have the same UIC as) the process or be in the same group as the process and have group privilege or world privilege.

To set process attributes, you must either own the process or have group or world privilege. To set priority you must have SETPRIOR privilege, and to add privileges you must have SETPRIV privilege.

The :kill and :hibernate operations automatically prompt for confirmation on each screenful of processes unless :auto is specified. Any time attribute switches are specified, confirmation on each process can be done with the :confirm switch.

If you specify the :files, :ksam ,:sharedmem, :devices, :traps, and :processstatus switches and the process has no files, KSAM files, etc., those fields do not show up in the display.

PSTAT is a "snapshot" of a process at a given instant in time. Whatever is shown for that instant may not have been true an instant before and probably will not be true an instant later.

If the :names switch is specified with :files, :ksam, :image, or :sharedmemory, only the filenames and shared memory names are displayed. The rest of the information about those attributes (such as access modes on files) is not displayed.

Related CIP Commands

None.

Functional Description

Use this command to purge a directory of all but the most recent versions of specific files.

Command Line Syntax

Mnemonic	pu			
Optional parameter	File List			
Switches				
Files to be purged	:before= :mod	:confirm :since=	:exclude= :uic=	:keep=
Other	:confirm	:log	:zerofiles	

Parameters

File List	Function	Optional. Use this parameter to specify the file(s) to be purged.
	Default	All files in the default directory are purged.
	Syntax	Type a list of files in the standard syntax.

Switches

:auto

Function — Use this switch to perform the desired operation without any confirmation.
Default — :noauto
Syntax — Type **:auto**

:before=

Function — Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.
Default — All files that match the File List parameter are selected.
Syntax — Type **:before=** followed by a date or time in the standard syntax.

:confirm

Function — Use this switch to confirm or deny the desired operation on each file.
Default — :noconfirm
Syntax — Type **:confirm**

:exclude=

Function — Use this switch to exclude files from the File List parameter.
Default — All files that match the File List parameter are selected.
Syntax — Type **:exclude=** followed by a list of file designations separated by commas. Wildcard characters are allowed.

:keep=

Function — Use this switch to specify the number of file versions to keep, i.e., these versions will not be deleted.
Default — :keep=1
Syntax — Type **:keep=** followed by a numeral.

:log

Function — Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
Default — The value specified by the OPTION command.
Syntax — Type **:nolog** or **:log** to override the default.

:mod

Function Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.

Default :nomod

Syntax Type **:mod**

:since=

Function Use this switch to select only those files that match the file list parameter and were created or modified since the specified date and time.

Default All files that match the File List parameter are selected.

Syntax Type **:since=** followed by a date or time in the standard syntax.

:uic=

Function Use this switch to select only those files that are part of the given list and are owned by the specified user or list of users.

Default All files that match the specified list are selected.

Syntax Type **:uic=** followed by a list of UICs or usernames.

:zerofiles

Function Use this switch to zero the sectors that the file occupies on the disk before deleting it.

Default :nozerofiles

Syntax Type **:zerofiles**

Examples

Suppose the default directory is __BARTLEBY_DS0/USERS/ and it contains the following files:

```
LETTER.TXT.1
LETTER.TXT.2
LETTER.TXT.3
LETTER.TXT.4
LETTER.TXT.5
```

When PU is executed, this kind of report appears:

```
     __BARTLEBY_DS0/USERS/LETTER.TXT.1
     __BARTLEBY_DS0/USERS/LETTER.TXT.2
     __BARTLEBY_DS0/USERS/LETTER.TXT.3
     __BARTLEBY_DS0/USERS/LETTER.TXT.4
         Delete (Y or N)? >
```

If you type Y, the following reports appear on the screen, one at a time:

```
     __BARTLEBY_DS0/USERS/LETTER.TXT.1 Deleted.
     __BARTLEBY_DS0/USERS/LETTER.TXT.2 Deleted.
     __BARTLEBY_DS0/USERS/LETTER.TXT.3 Deleted.
     __BARTLEBY_DS0/USERS/LETTER.TXT.4 Deleted.
         4 Files Deleted.
```

If you type N, the process terminates and no files are purged. If you type any letter other than Y or N, the characters are ignored.

> pu :keep=3 :auto

If the default directory is the same as the previous example, the following report appears:

```
     __BARTLEBY_DS0/USERS/LETTER.TXT.1 Deleted.
     __BARTLEBY_DS0/USERS/LETTER.TXT.2 Deleted.
2 Files Deleted.
```

Only the first two versions of the file LETTER.TXT are purged because the :keep switch changed the keep default from 1 (the most recent version) to 3 (the 3 most recent versions).

> pu _dc0/users.doc/[c-d]*

This command causes the following kind of report to appear:

```
__BARTLEBY_DS0/USERS.DOC/CIP.DAT.1
__BARTLEBY_DS0/USERS.DOC/CIP.DAT.2
__BARTLEBY_DS0/USERS.DOC/DM.DAT.1
Delete (Y or N)? >
```

If you type Y, the following reports appear one at a time:

```
__BARTLEBY_DS0/USERS.DOC/CIP.DAT.1 Deleted.
__BARTLEBY_DS0/USERS.DOC/CIP.DAT.2 Deleted.
__BARTLEBY_DS0/USERS.DOC/DM.DAT.1 Deleted.
3 Files Deleted.
```

The wildcard set [c-d]* in the command string specifies that only files beginning with C or D are purged.

Using Prompts

None.

Notes on Usage

The report "No Files Found" means no files in the directory had extra versions to be purged.

Related CIP Commands

del	Delete files

qprint

Functional Description

Use this command to format a file to be printed.

Command Line Syntax

Mnemonic	qprint			
Optional parameter	Queue Name			
Switches	:copies=	:newline	:number	:pagehead
	:header	:pages=	:lines=	:raw
	:formlength=	:formwidth=	:printnull	:fortran
	:killedout=			

Parameters

Queue Name	Function	Optional. Use this parameter to specify the name of the queue you are printing to.
	Default	SYS$PRINT
	Syntax	Type a devicename. No wildcard symbols are allowed.

Switches

:copies=	Function	Use this switch to specify how many copies of the file are to be printed.
	Default	:copies=1
	Syntax	Type **:copies=** followed by any valid integer.
:formlength=	Function	Use this switch to specify the number of lines per page.
	Default	The form length as specified by the DSTAT :length= switch.
	Syntax	Type **:formlength=** followed by a positive numeral.
:formwidth=	Function	Use this switch to specify the number of characters per line.
	Default	The form width as specified by the DSTAT :width= switch.
	Syntax	Type **:formwidth=** followed by a positive numeral.
:fortran	Function	Use this switch to do FORTRAN carriage control mapping.
	Default	:nofortran
	Syntax	Type **:fortran**
:header	Function	Use this switch to print a header page before each file (and each copy of a file) is printed.
	Default	:header
	Syntax	Type **:noheader**
:killedout=	Function	Use this switch to specify a string of data to be written to the printer if for any reason QPRINT does not go to normal completion. This is very useful for resetting printers that may be left in different modes such as graphics mode, so that the next print job will print properly.
	Default	None.
	Syntax	Type **:killedout=** followed by hexadecimal, decimal or text data.

Any data which has spaces as either delimiters or data must be enclosed in quotation marks. Hexadecimal data is specified as hexadecimal digits with one or more spaces separating data bytes, e.g., "7f ff ff ff".

Decimal data is the same as hexadecimal data except that each data byte is preceded with a percent sign, %. For example, "%127 %255 %255 %255" is the same as the hexadecimal example.

Text data must be enclosed in parentheses, e.g. (ABCD) or "(AB CD EFGH)". Any special characters which must be text data, i.e., the characters "[]()\", must be preceded by a backslash, \, to be accepted as text. For example, (\(\)) produces the text data "()".

A repeat count may be specified for data which repeats. Square brackets, [], enclose the repeating data and are preceded by a repeat count. There may not be a space between the repeat count and the leading bracket. An example of the hexadecimal example with a repeat count is "7f 3[ff]". Repeat counts may be nested to any level so long as the expanded data does not exceed 3072 bytes. The repeat count is assumed to be a hex number unless preceded by a "%" character as in %50[0d].

:lines=

Function: Use this switch to specify the lines to be printed. If the :pages= switch is specified, it overrides any values specified here.

Default: All lines are displayed.

Syntax: Type **:lines=** followed by a range of integers. Line numbering begins at 1.

:newline

Function: Use this switch to specify whether linefeed characters are converted into a carriage return-linefeed sequence before being printed.

Default: :newline

Syntax: Type **:nonewline**

:number	Function	Use this switch to specify that a line number is to be added at the left margin of each line that is displayed.
	Default	:nonumber
	Syntax	Type **:number**
:pagehead	Function	Use this switch to print a header at the top of each page. This header contains the page number, the name of the file, and the time. Two blank lines are printed before the normal text of that page appears. These lines are not numbered.
	Default	:nopagehead
	Syntax	Type **:pagehead**
:pages=	Function	Use this switch to specify the pages to be printed.
	Default	All pages are printed.
	Syntax	Type **:pages=** followed by a range of integers. Page numbering begins at 1.
:printnull	Function	Use this switch to print a null file.
	Default	:noprintnull
	Syntax	Type **:printnull**
:raw	Function	Use this switch to specify that the data being written are not to be modified or examined. This switch is necessary when printing graphics-type data.
	Default	:noraw
	Syntax	Type **:raw**

Examples

```
> qprint _pql <abc.dat >_ttl
```

This command reads ABC.DAT, formats it, and writes it to _TTl. All default parameter values are obtained from the queue _PQl.

Using Prompts

None.

Notes on Usage

Normally a user will not have to use QPRINT because the PRINT Command and queue manager do formatting.

QPRINT is expecially useful for those who want to do special things or write their own formatter.

With QPRINT, users can specify the :printtype= switch to DSTAT properly. This specifies the various options that can be used.

Related CIP Commands

print	Print text files and monitor print queue

recover

Functional Description

Use this command to rebuild the file system on a disk.

Command Line Syntax

Mnemonic	recover		
Required parameter	Devicename		
Optional parameter	Message		
Switches			
Message	:bell	:broadcast	
Others	:hibernate	:pause	:stats

Parameters

Devicename	Function	Required. Use this parameter to specify the name of the disk whose file system is to be recovered.
	Default	None.
	Syntax	Type the name of a disk device.

Message	Function	Optional. Use this parameter to specify an optional message to ali system users, telling them why the disk is being recovered.
	Default	Unless you type :nobroadcast, the following kind of message is sent to all system users: (CIP_MARTIN) RECOVER beginning on _DC0
	Syntax	Type a string enclosed in double quotation marks.

Switches

:bell	Function	Use this switch to output a bell along with a message.
	Default	:bell
	Syntax	Type **:nobell**
:broadcast	Function	Use this switch to suppress the broadcast message.
	Default	:broadcast
	Syntax	Type **:nobroadcast**
:hibernate	Function	Use this switch to hibernate all other processes on the system while RECOVER is running. Use this switch only when RECOVER cannot allocate the device, that is, when other processes have files open on thc device.
	Default	:nohibernate
	Syntax	Type **:hibernate**
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when you press any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.

:stats	Function	Use this switch to request that various statistics be displayed.
	Default	:nostats
	Syntax	Type **:stats**

Examples

> recover _dc0

This command inspects and rebuilds the file system on device _DC0. The following kind of report is displayed on the terminal:

```
(CIP_MARTIN) RECOVER beginning on _DC0

Checking the boot blocks
Verifying FCBs and building BITMAP and FCBBITMAP
Rebuilding Directory Structure

(CIP_MARTIN) RECOVER finished on _DC0
```

> recover _ds0 :hibernate "The system crashed due to a power outage"

This command rebuilds the directory structure on device _DS0 and sends the following message to all system users:

```
(CIP_JAMES) RECOVER beginning on _DS0
The system crashed due to a power outage.
```

If RECOVER is unsuccessful at allocating the device, it attempts to hibernate all other processes on the system. When RECOVER finishes, it awakens all other processes. Note that even though the :hibernate switch is specified, processes are not to be hibernated unless the device cannot be allocated. A device can be allocated if it is not already allocated, and if there are no open files on the device.

If you attempt to run RECOVER and it reports that it cannot allocate the device, try it again with the :hibernate switch.

> recover _dx0 :stats

This command recovers the device _DX0 and prints statistics regarding the boot block and file system. The following kind of display appears as RECOVER begins:

```
                ** Boot block parameters found at sector 0 **

Devname  : DX            Protection: S:  RE,P:    ,G:DWRE,O:DWRE
Label    : Transfer disk Volumeid  : 0    Created   : 11-Jan-1984 21:49
Owner    : [0007,0003]   Cache     : 30   ReadAhead
Username : LARRY         Usercache : 8    NoAutoFlush
           616           Sectorsz  : 1024 NoForcedWrite
Fcbsector: 302           Shiftcnt  : 10   Drivetype : FLOP09a
Ialloc   : 10            Alloc     : 10
```

After RECOVER finishes rebuilding the directory structure, the following report appears on the screen:

```
                           **  STATISTICS **
           * File Size *                            * Extent Size *
 0Kb to   1Kb  :          4          80%           1 Sect  :       4    80%
 1Kb to   5Kb  :          0           0%     2 to  5 Sects :       0     0%
 5Kb to  20Kb  :          1          20%     6 to 20 Sects :       1    20%
20Kb to 100Kb  :          0           0%    21 to 40 Sects :       0     0%
   Over 100Kb  :          0           0%     Over 40 Sects :       0     0%

           * File Type *                         * Number of Extents *
Data Files     :         0      0K    0%           1 Xtnt  :       5   100%
Directories    :         2      2K   10%     2 to 10 Xtnts :       0     0%
Image Files    :         0      0K    0%    11 to 30 Xtnts :       0     0%
System Files   :         3     18K   90%    31 to 70 Xtnts :       0     0%
Other Files    :         0      0K    0%    Over  70 Xtnts :       0     0%

Total Files    :          5                  Total Extents  :        5
Tot Phys Size  :      20480                  Largest # Xtnts:        1
Tot Log Size   :      20480                  Largest Extent :       16
Largest File   :      16384                  Avg # of Xtnts :        1
Avg File Size  :       4096                  Avg Xtnt Size  :        4
Disk Capacity  :     630784                  Number of FCBs :       64
FCBs - Free=         59, Primary=         5, Continuation=        0
```

Using Prompts

```
> recover
Devicename    > _dc0
```

This command performs the same function as the first example.

Notes on Usage

Whenever your system crashes, (power outage, accidental reset, system freezes and has to be rebooted, hardware malfunction, etc.) run RECOVER on all disks mounted at the time of the crash.

RECOVER makes sure no other process has access to the disk. First, it tries to allocate the disk. If the disk is allocated, no other processes can access it. If it cannot allocate the disk (other processes already have files open on the disk) and if the :hibernate switch is used, RECOVER hibernates all other processes on the system. This ensures that no changes will be made to the disk during RECOVER.

The device you specify (for parameter 1) must be mounted (see the MNT Command) before RECOVER can be run on it.

If the boot block on the disk has been corrupted and the disk cannot be mounted, mount it with the :special switch and then run RECOVER.

If RECOVER does not find a valid boot block at sector 0 of the disk (when the disk is mounted as :special), it attempts to construct one. If RECOVER can construct a boot block, it will continue to rebuild the disk. Then you can dismount and remount it regularly. If RECOVER cannnot construct a boot block, RECOVER terminates. That is, RECOVER does not attempt to recover the FCBs or directories if it does not find a valid boot block at sector 0 on the disk.

NOTE: Sometimes you can use Zap to fix the boot block.

RECOVER rebuilds these files:

FCB.SYS
BITMAP.SYS
FCBBITMAP.SYS
All directory files on the volume.

To rebuild these files, RECOVER uses the information in FCB.SYS. FCB.SYS contains a File Control Block (FCB) for each file on a particular device. For example, the FCB.SYS file in /ROOTDIR/ on _DC0 contains an FCB for every file on _DC0, while the FCB.SYS file in /ROOTDIR/ on the disk in drive _DX0, contains an FCB for every file on that volume.

Suppose you type the following character string on the CIP command line:

```
> recover _dc0
```

This report appears on the screen when you strike [RETRN]:

```
Checking the boot blocks.
```

This tells you RECOVER is checking the values in each field of sector 0 on the volume. Sector 0 is called the boot block.

Following this check, RECOVER displays this kind of report:

```
Verifying FCBs and building BITMAP and FCBBITMAP
```

This report tells you RECOVER is using the information in SYS$DISK/ROOTDIR/FCB.SYS to rebuild BITMAP.SYS and FCBBITMAP.SYS. RECOVER also tries to correct any FCBs with incorrect data.

Following this part of RECOVER's operation, this report appears on the screen:

```
Rebuilding directory structure.
```

This report tells you RECOVER is creating an entry in the appropriate directory file for every file on the disk. Any files not assigned to a directory are assigned to the directory /LOST/. When RECOVER is completed, you can reassign any file to any directory you want.

RECOVER takes approximately 4 to 5 minutes on a CMI Winchester disk, and 30 or more minutes on an SMD 474.

Glossary of RECOVER Diagnostic Messages

RECOVER prints two formats of diagnostic messages. One is the standard three-line diagnostic that results from a non-zero status code returned by WMCS. These three-line diagnostic messages result mostly from file errors.

The second format is characterized by six asterisks. These messages result from situations detected by RECOVER. When RECOVER discovers something amiss, it displays a one-line diagnostic that begins with six asterisks.

This glossary lists all of the diagnostics in the second category and defines them.

****** The disk drive table is not initialized.

Sector zero (the boot block) has two parts. The first 256 bytes contain volume information. The second 256 bytes contain what is called a drive table. The drive table is information needed by the driver to know how to access the drive. If this second half of the boot block is not initialized, this message appears. See the BTUP utility for information on how to initialize the drive table.

****** Sector 1 does not contain valid backup FCBs.

RECOVER reads the sector immediately following the boot block. This message indicates the first 256 bytes of that sector do not resemble FCB zero. No corrective action is taken.

****** Unable to rebuild boot block.

If RECOVER attempts to rebuild the boot block on a disk and either cannot get the device status for the disk or cannot find the first sector of the FCB.SYS file, this message is displayed.

****** Boot block check failed - Rebuilding boot block.

This message announces RECOVER is attempting to reconstruct a boot block for the volume.

****** Setting record length to 256 for _DC0/ROOTDIR/FCB.SYS.

FCB 0 is assigned to the FCB.SYS file whose record length is (by definition) 256. If RECOVER finds the record length for FCB.SYS is not 256, this message is presented and 256 is reassigned as the record length.

****** Setting record length to nnn for _DC0/ROOTDIR/FCBBITMAP.SYS.

FCB 2 is assigned to the FCBBITMAP.SYS file, whose record length (by definition), is one half the sector size. If RECOVER finds this is not the case, it reports this message and sets the record length to one half the sector size.

****** Invalid sector size = nnn

The sector size field of a boot block is not equal to 512 or 1024 nnn represents the unacceptable sector size). No corrective action is taken.

****** Invalid sector size/shift count combination. Being corrected.

If the sector size field of the boot block is 512 and the shift count field is not equal to 9, or the sector size is 1024 and the shift count is not equal to 10, this message is displayed. The shiftcount is then set to the correct value (9 or 10) depending on the sector size.

****** Invalid cachesize. Being set to nnn.

The value in the cache size field of the boot block is less than or equal to the value in the user cache field of the boot block. RECOVER sets the cache size field to 2, or to the same value as the user cache field, whichever is greater.

****** Invalid maximum cache allocation. Being set to nnn.

The value of the user cache field of the boot block is 0. (NNN is set to the total cache size divided by 2).

****** FCB file not pointed to by boot block.

The FCB sector field of the boot block points to a sector that does not contain FCB 0. An attempt is made to find the FCB file and rebuild the boot block.

****** Invalid pointer to FCB file

Either the value of the FCB sector field of the boot block is 0 or it is greater than the total number of sectors on the disk (as indicated by the sector count field of the boot block). An attempt is made to find the FCB file and rebuild the bootblock.

****** Invalid file type nnn for <fcb.seq>. Being set to Data file type.
       <filename>

The file type of the specified file is one of the file types reserved for future use by WICAT. It is reset to zero.

****** Error allocating bit number nnn

RECOVER allocates memory for BITMAP.SYS and FCBBITMAP.SYS when RECOVER starts. If RECOVER subsequently tries to set a bit in either file, and finds the bit number is too large, e.g., it must set bit 20000 and 10000 bits are in the table, this diagnostic message is displayed. The error is internal to RECOVER.

****** Error allocating memory for BITMAP

RECOVER allocates memory for building BITMAP.SYS and FCBBITMAP.SYS. This diagnostic message indicates the necessary amount of memory cannot be allocated; RECOVER is terminated.

****** Changing FCBBITMAP from 3.1 format to 4.0 format

If the record length of FCBBITMAP.SYS (file //2.2) is not equal to half the sector size of the volume, RECOVER assigns zero to each byte in the status portion of FCBBITMAP.SYS.

****** Sector nnn not allocated, was previously

A bit, in the version of BITMAP.SYS prior to the execution of RECOVER, was allocated and RECOVER found the same bit unallocated in the new version of BITMAP.SYS.

****** Sector nnn allocated, was not previously.

A bit, in the version of BITMAP.SYS prior to the execution of RECOVER, was unallocated and RECOVER found the same bit allocated in the new version of BITMAP.SYS.

****** Invalid record length in <fcb.seq>. Being set to one. <filename>

RECOVER encountered a file whose record length is zero, and set the record length to one.

****** Invalid extent count for <fcb.seq>. Set to 30. <filename>

Each file's primary FCB can contain information on as many as 30 extents. RECOVER found a primary FCB whose extent count field is greater than 30. RECOVER changed the value to 30.

****** Invalid filename for <fcb.seq> <filename>

RECOVER found an FCB with a filename that contained characters not allowed in filenames. RECOVER changes each disallowed character to a field. RECOVER also verifies there are no characters (allowed or otherwise) following the first null byte in the filename field of an FCB. If any characters are found after the first null byte, this message is reported and that byte is set to null.

****** Invalid file extension for <fcb.seq> <filename>

RECOVER found an FCB containing a file extension that contained characters not allowed in file extensions. RECOVER changes each disallowed character to a tilde, ~. RECOVER also verifies there are no characters (allowed or otherwise) following the first null byte in the file extension field to the FCB. If any characters are found after the first null byte, this message is displayed and a null is assigned to that byte.

****** Setting directory file version number to 1 for <fcb.seq> <filename>

The version number field of a primary FCB for a directory file must contain 1. RECOVER found an FCB for a directory file with a version number other than 1, and reassigns 1 as the version number.

****** Invalid extent field in <fcb.seq>. Extent set to zero. <filename>

An extent field in an FCB of a file had either too high a sector count or too large a base sector number. The sector count and base sector number of that extent are set to 0 and the FCB is written back to FCB.SYS.

****** Invalid extent field in FCB # nnn of <fcb.seq>. Extent set to zero. <filename>

An extent field in a secondary FCB of a file had either too high a sector count or too large a base sector number. The sector count and base sector number of that extent are set to zero and the FCB is written back to FCB.SYS.

****** Extra extent fields zeroed in <fcb.seq> <filename>

An unused extent field (beyond the extent field count) in an FCB is greater than zero. RECOVER assigns 0 as the value for the sector count and sector address fields, and the FCB is written to FCB.SYS.

***** Extra extent fields zeroed in FCB # nnn of <fcb.seq> <filename>.

An unused extent field (beyond the extent field count) in a continuation FCB is greater than zero. RECOVER assigns 0 as the value for the sector count and sector address fields, and the FCB is written to FCB.SYS.

****** Correcting FCB number on FCB # nnn

The FCB number of a continuation FCB doesn't match the record number. RECOVER sets the FCB number equal to the record number and writes the FCB back to FCB.SYS.

****** Invalid sequence number in continuation FCB # nnn. <fcb.seq> truncated. <filename>

The sequence number of a continuation FCB does not match the sequence number referred to in the FCB that pointed to the continuation FCB. The FCB containing the reference to the continuation FCB is designated the last FCB for the file and is written to FCB.SYS.

****** Continuation FCB for <fcb.seq> is invalid. File truncated. <filename>

If the FCB number of a continuation FCB of a file is beyond the end of the FCB.SYS file, this error is reported. The file is truncated.

****** Invalid usage field of continuation FCB# nnn of <fcb.seq>. Being set to 2. <filename>

The usage field of a continuation FCB is not equal to 2. The usage field is set to 2.

****** Invalid extent count for FCB # nnn of <fcb.seq>. Being set to 40. <filename>

The extent count field of a secondary FCB is greater than 40. The number of extents is set to 40.

****** FCB # nnn has usageid nnn. FCB freed

The specified FCB is not marked as available or primary, and is not referred to (by a primary FCB) as a continuation FCB. RECOVER designates the FCB as available.

****** There are nnn shared sectors in the following files:

A sector is referred to by more than one FCB. The following message is displayed for each file whose FCB refers to the sector:

****** sector nnn found in (filename), or bootblock, or backup FCB

****** Correcting checksum on FCB for <fcbseq>. <filename>

The FCB checksum does not match the data in the FCB. RECOVER puts the correct checksum in the checksum field of the FCB.

****** Correcting FCB number of FCB for fcb.seq <filename>

The FCB number field of an FCB does not match the record number. RECOVER changes the FCB number field to match the record number.

****** FCB for <fcb.seq> was marked bad. FCB freed. <filename>

FCBBITMAP.SYS indicates the data in the specified FCB are unreliable. RECOVER assigns a value to the FCB's usage field, indicating the FCB is unassigned.

****** Correcting file size information in <fcb.seq> <filename>

The physical size of a file, as indicated in one of the fields of the FCB, does not correspond to the computed file size. RECOVER changes the value, indicating the file's size, to correspond with the computed file size. Furthermore, if the FCB indicates the file's logical size is greater than its physical size, the value indicating the logical size is made equal to the value indicating the file's physical size.

****** Error attempting to allocate sector nnn

RECOVER attempted to allocate a bit in BITMAP.SYS for a sector whose logical sector number is larger than the total number of sectors on the disk.

****** Setting directory record length to 32 for <fcb.seq> <filename>

The record length of a directory file is 32 bytes. RECOVER found a directory file whose record length was not 32, and changes its record length to 32.

****** Unable to create LOST.DIR. Lost file will be put in ROOTDIR

RECOVER is unable to create or open LOST.DIR.

****** File <fcb.seq> has been renamed to /LOST/ <filename>

An FCB specified a directory file, as the directory file containing the file to which the FCB is assigned, and the directory file could not be found. The file is therefore assigned to /LOST/.

****** File <fcb.seq> has been renamed to /ROOTDIR/ <filename>

A directory file, corresponding to the directory file specified in an FCB as the directory containing the file, could not be found, and RECOVER could not create /LOST/. Therefore, the file is assigned to /ROOTDIR/.

****** Consolidating directory file <fcb.seq>. <filename>

Sometimes directory files become "stretched" and they consume more space than is required for the files in them. This happens when many files are copied to a directory and are deleted later. RECOVER shrinks these directories to the size they should be. This is done not so much to save disk space, but rather to improve access time to open these files.

****** <fcb.seq> has been successfully relocated around a bad sector.

If a data crc error is returned from WMCS when RECOVER is writing a sector of the FCBBITMAP.SYS file or any directory file, RECOVER will attempt to reposition the file around the bad sector. If it is successful in its attempt, this message is displayed.

Related CIP Commands

btup	Display or set disk boot blocks or tape labels
zap	Display a file in hexadecimal and inspect or emend it

ren

Functional Description

Use this command to rename files and directories.

Command Line Syntax

Mnemonic	ren
Required parameters	Source Destination

Switches

File selection	:before= :exclude=	:mod :uic=	:confirm	:since=
Destination specification	:builddir	:edit=		
Others	:log			

Parameters

Source	Function	Required. Use this parameter to specify the names of the files to be renamed.
	Default	None.
	Syntax	Type a list of file designations separated by commas. Wildcard symbols are allowed.

Destination	Function	Use this parameter to specify the new names of the source files.
	Default	None.
	Syntax	Type a destination file designation. Wildcard symbols are allowed in all fields.

Switches

:before=	Function	Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:before=** followed by a date or time in the standard date and time syntax.
:builddir	Function	Use this switch to create destination directories automatically if they do not exist.
	Default	:nobuilddir
	Syntax	Type **:builddir**
:confirm	Function	Use this switch to confirm or deny the desired operation on each file.
	Default	:noconfirm
	Syntax	Type **:confirm**
:edit=	Function	Use this switch to specify a series of transformations to be performed on the destination file designation. The transformation is specified as a series of "find and switch" strings. The destination file designation is searched for the first occurrence of the "find" string. If it is found, it is replaced by the "switch" string.
	Default	No transformation is performed and the destination file designation is used as is.
	Syntax	Type **:edit=** followed by a list of "find and switch" strings separated by commas. The find string is separated from the switch string by a colon. Either one or both of the strings may be empty.

:exclude=

Function Use this switch to exclude files from the File List parameter.

Default All files that match the File List parameter are selected.

Syntax Type **:exclude=** followed by a list of file designations separated by commas. Wildcard characters are allowed.

:log

Function Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

Default The value specified by the OPTION command.

Syntax Type **:nolog** or **:log** to override the default.

:mod

Function Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.

Default :nomod

Syntax Type **:mod**

:since=

Function Use this switch to select only those files that match the file list parameter and were created or modified since the specified date and time.

Default All files that match the File List parameter are selected.

Syntax Type **:since=** followed by a date or time in the standard syntax.

:uic=

Function Use this switch to select only those files that are part of the given list and are owned by the specified user or list of users.

Default All files that match the specified list are selected.

Syntax Type **:uic=** followed by a list of UICs or usernames.

Examples

> ren product.aug report.txt

This command renames the latest version of the file PRODUCT.AUG in the default directory to REPORT.TXT. The file still appears in the default directory. Note that REN is not the same as COPY. The contents of the file are not moved; only the filename is changed. The following kind of report appears on your screen:

```
_DS0/USERS.MORREY/PRODUCT.AUG.12 renamed to _DS0/USERS.MORREY/
REPORT.TXT
```

> ren product.aug /office/report.txt

This command line renames the latest version of the file PRODUCT.AUG in the default directory and puts the file into the directory /OFFICE/ under the name REPORT.TXT.

> ren /users.lewis/memos.dir /users.brown/*.*

This command renames all of the files in the directory / USERS.LEWIS.MEMOS/ and all of its subdirectories to the directory / USERS.BROWN/. Renaming a directory grafts that directory, all of its files, and all of its subdirectories and their files into a different place in the directory structure.

All files that used to be in directory /USERS.LEWIS.MEMOS/ and its subdirectories are now in /USERS.BROWN.MEMOS/ and its subdirectories:

```
                                --------
                                 Before
                                --------
                                     |
                                  users
                                     |
                    |------------------------------------|
                 lewis                                brown
                    |
             |-------------|
          memos        projects
             |
   |------------------|
9june.txt         annual
                      |

                                --------
                                 After
                                --------
                                    |
                                 users
                                    |
                 |------------------------------------|
              lewis                                brown
                 |                                    |
          |--------------|                          memos
                    projects                          |
                                         |------------------|
                                     9june.txt          annual
                                                            |
```

> ren *.*.* projects/ :confirm

This command renames all files in the default directory to the directory /PROJECTS/. Before renaming each file, the operator is prompted to confirm the operation. A prompt like the following appears for each file that matches the source file designation:

```
     Rename _DC0/KAHAN/BUDGET.DAT.1 to _DC0/PROJECTS/BUDGET.DAT (Y or N)?
>
```

Type a Y or an N in response to each prompt. Any other characters are ignored. You do not need to strike [RETRN].

Using Prompts

```
> ren :confirm
Source        > *.*.*
Destination   > projects/
```

This command performs the same function as the previous example.

Notes on Usage

REN is used for more than renaming files; it can also move files around on the same disk.

You can only rename files found on disk devices. Tape files cannot be renamed.

You cannot rename a file from one disk to another. It can only be renamed to another place on the same disk.

You cannot rename a directory file as a subdirectory of itself. For instance, suppose you have a directory file named /USERS.LEWIS/ MEMOS.DIR.1. This file cannot be renamed to /USERS.LEWIS.MEMOS/ since that generates a recursive directory structure.

REN allows you to change the version number of the file, but you cannot rename a directory and assign it a version number greater than 1.

REN does not affect the file's modification date, creation date, owner, protection, or any other attribute.

Related CIP Commands

copy	Copy files

restore

Functional Description

Use this command to restore files from a backup file.

Command Line Syntax

Mnemonic	restore			
Required parameters	Backup File			
	File List			
	Destination			
Switches				
Files to be included in restoration	:abefore=	:auic=	:mod	:uic=
	:aexclude=	:before=	:override	:exclude=
	:amod	:confirm	:since=	:arc
	:asince=			
Interaction with media	:mount=	:rewind	:volume=	
Creation of restored files	:builddir	:owner=	:protection=	:edit=
	:preserve=			
Other	:log	:verify	:showall	

Parameters

Backup File	Function	Required. Use this parameter to specify the backup file from which files are to be restored.
	Default	None.
	Syntax	Type a filename in the standard file designation syntax. Wildcard symbols are allowed.
File List	Function	Required. Use this parameter to specify the filenames to be restored from the backup file.
	Default	None.
	Syntax	Type a filename in the standard file designation syntax. Wildcard symbols are allowed.
Destination	Function	Required. Use this parameter to specify the destination for the files that are restored.
	Default	None.
	Syntax	Type a filename. Wildcard symbols are allowed in the directory name, filename, file extension, and version number fields of the file designation. However, the asterisk must be the only character in the field in which it appears.

Switches

:abefore=	Function	Use this switch to select only those files from the archive file that match the value specified for the File List and that were created or archived before a particular date and time.
	Default	All files that match the value specified are selected from the archive file.
	Syntax	Type **:abefore=** followed by the date and time.

:aexclude=	Function	Use this switch to select only those files from the archive file that match the File List parameter and do NOT match the :aexclude= switch.
	Default	All files that match the File List parameter are selected.
	Syntax	Type a list of file designations, separated by commas, any one of which may contain wildcard symbols.
:amod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :abefore= or :asince= switches.
	Default	:noamod.
	Syntax	Type **:amod**
:arc	Function	Use this switch to specify that the archival date is to be used in all date and time considerations by the :abefore= or :asince= switches.
	Default	:noarc
	Syntax	Type **:arc**
:asince=	Function	Use this switch to select only those files from the archive file that match the File List parameter and were created/modified/archived since a particular date and time.
	Default	All files that match the File List parameter are selected from the archive file.
	Syntax	Type **:asince=** followed by the date and time in the standard date and time syntax.
:auic=	Function	Use this switch to select only those files from the archive file that match the File List parameter and are owned by a particular user.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:auic=** followed by a UIC or the username of an authorized system user.

:before=

Function Use this switch to select only those files that match the File List parameter and were created or modified before the specified date and time.

Default All files that match the File List parameter are selected.

Syntax Type **:before=** followed by a date or time in the standard date and time syntax.

:builddir

Function Use this switch to create destination directories automatically if they do not exist.

Default :nobuilddir

Syntax Type **:builddir**

:confirm

Function Use this switch to confirm or deny the desired operation on each file.

Default :noconfirm

Syntax Type **:confirm**

:edit=

Function Use this switch to specify a series of transformations to be performed on the destination file designation. The transformation is specified as a series of "find and switch" strings. The destination file designation is searched for the first occurrence of the "find" string. If it is found, it is replaced by the "switch" string.

Default No transformation is performed, and the destination file designation is used as is.

Syntax Type **:edit=** followed by a list of "find and switch" strings separated by commas. The find string is separated from the switch string by a colon. Either one or both of the strings may be empty.

:exclude=

Function Use this switch to exclude files from the File List parameter.

Default All files that match the File List parameter are selected.

Syntax Type **:exclude=** followed by a list of file designations, separated by commas, any one of which may contain wildcard characters.

:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**
:mount=	Function	Use this switch to mount the backup media.
	Default	None.
	Syntax	Type **:mount=** followed by a MNT command string enclosed in double quotation marks.
:override	Function	Use this switch to override the requirement that the ARCH and BACKUP utilities work only on archive type files.
	Default	:nooverride
	Syntax	Type **:override**
:owner=	Function	Use this switch to change the owner (UIC) of the file.
	Default	The UIC assigned when the file was created or last updated.
	Syntax	Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC that they represent.
:preserve=	Function	Use this switch to preserve certain attributes of the original file when a new copy of it is created. The preservable aspects are: creation date, owner, and protection.
	Default	The original file's attributes are not preserved in the new file, i.e., the new file is created with the current date and time, the owner is the current user, and the protection is the user's default protection mask.
	Syntax	Type **:preserve=** followed by any combination of the following keywords separated by commas: **created**, **owner**, **protection**, or **all.**

:protection= Function Use this switch to modify the protection mask associated with a file.
Default The owner's protection mask.
Syntax Type **:protection=** followed by a protection mask.

Only those fields of the protection mask that are specified are altered. Fields not specified keep their original value.

For example, :protection=s:,o:rwe allows the system no privilege, while the owner is granted read, write, and execute privilege. The public and group privileges are unchanged.

:rewind Function Use this switch to rewind the tape device.
Default :norewind
Syntax Type **:rewind**

:showall Function Use this switch to display the backup filenames as they are scanned for files.
Default :noshowall
Syntax Type **:showall**

:since= Function Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.
Default All files that match the File List parameter are selected.
Syntax Type **:since=** followed by a date or time in the standard syntax.

:uic= Function Use this switch to select only those files that are part of the given list and are owned by the specified user or list of users.
Default All files that match the specified list are selected.
Syntax Type **:uic=** followed by a list of UICs or usernames.

:verify Function Use this switch to report whether the command line character string has been correctly executed.
Default :noverify
Syntax Type **:verify**

:volume= Function Use this switch to specify the volume number of the first volume in the restoration.
Default :volume=1
Syntax Type **:volume=** followed by a number.

Examples

> restore _dx0/rootdir/10jan0001.bak * *

This command restores all of the files in 10JAN0001.BAK to files by the same name in your default directory. The following kind of display appears:

```
__MEPH_DC0/USERS.FRANK/TEST.EXE.2 restored as __MEPH_DC0/USERS.AL/TEST.EXE
__MEPH_DC0/USERS.FRANK/TEST.HLP.2 restored as __MEPH_DC0/USERS.AL/TEST.HLP
__MEPH_DC0/USERS.FRANK/TEST.MCR.2 restored as __MEPH_DC0/USERS.AL/TEST.MCR
__MEPH_DC0/USERS.FRANK/TEST.MRL.2 restored as __MEPH_DC0/USERS.AL/TEST.MRL
__MEPH_DC0/USERS.FRANK/TEST.OUT.1 restored as __MEPH_DC0/USERS.AL/TEST.OUT
__MEPH_DC0/USERS.FRANK/TEST.PAS.4 restored as __MEPH_DC0/USERS.AL/TEST.PAS
__MEPH_DC0/USERS.FRANK/TEST.PRN.2 restored as __MEPH_DC0/USERS.AL/TEST.PRN
```

Each line of the display shows the original directory path and filename of the file. Following the words "restored as," the new directory path and filename appear.

> restore _dx0/rootdir/10jan0001.bak * * :preserve=all

This command generates the same kind of display as the previous example. The only difference is the values of the protection, owner and creation times of the files restored. In the first example, assuming Grace is the user doing the restoration, all of the files restored would have Grace's default protection, have Grace as their owner, and would have the current date and time as their creation date and time.

In this example, the files restored would preserve their original protection, owner, and creation date and time. Assuming Frank is the user who originally created the files before they were backed up, the newly restored files would have Frank's default protection, have Frank as their owner, and still have their original creation times.

```
> restore _dx0/rootdir/10jan0001.bak * * :edit="frank:jim.work" :builddir
```

This command string produces the following display:

```
__MEPH_DC0/USERS.FRANK/TEST.EXE.2 restored as __MEPH_DC0/USERS.JIM.WORK/TEST.EXE
__MEPH_DC0/USERS.FRANK/TEST.HLP.2 restored as __MEPH_DC0/USERS.JIM.WORK/TEST.HLP
__MEPH_DC0/USERS.FRANK/TEST.MCR.2 restored as __MEPH_DC0/USERS.JIM.WORK/TEST.MCR
__MEPH_DC0/USERS.FRANK/TEST.MRL.2 restored as __MEPH_DC0/USERS.JIM.WORK/TEST.MRL
__MEPH_DC0/USERS.FRANK/TEST.OUT.1 restored as __MEPH_DC0/USERS.JIM.WORK/TEST.OUT
__MEPH_DC0/USERS.FRANK/TEST.PAS.4 restored as __MEPH_DC0/USERS.JIM.WORK/TEST.PAS
__MEPH_DC0/USERS.FRANK/TEST.PRN.2 restored as __MEPH_DC0/USERS.JIM.WORK/TEST.PRN
```

This command restores all files from 10JAN0001.BAK to their same filenames in the default directory. However, the :edit="FRANK:JIM.WORK" switch replaces the character string FRANK in any file designation with JIM.WORK in the destination file designation. The :builddir switch creates the resulting directory path, if it does not already exist. Thus the files are restored to __MEPH_DC0/USERS.JIM.WORK/even if the directories JIM and JIM.WORK did not exist earlier.

The following command line performs the same function:

```
     > restore _dx0/rootdir/10jan0001.bak * users.jim.work/* :builddir
```

All files are restored from 10JAN0001.BAK to their original filenames in the directory /USERS.JIM.WORK/. If the directory path /USERS.JIM.WORK/ does not exist, it is created.

Using Prompts

```
> restore
> Backup file  > 10jan0001.bak
> File list    > *
> Destination  > users.jim.work/* :builddir
```

This performs the same function as the previous example.

Notes on Usage

RESTORE can be used to restore a portion or all of the files backed up during a particular backup session. RESTORE can deal with the many volumes of media which are often involved in a backup.

Suppose you have backed up your entire system disk to floppy diskettes and you now wish to restore the system disk. Insert a diskette into drive _DX0 and mount it. Then type the following on the command line and strike [RETRN]:

```
     > restore _dx0/rootdir/* * _dc0/*/* :builddir
```

When all of the files are restored from the diskette in drive _DX0, the following prompt appears on your screen:

```
     Insert volume 2
     Press RETURN when ready
```

Remove the diskette in _DX0, place another diskette in that drive, and strike [RETRN]. The following prompt appears on your terminal screen:

```
     Label = backup2
     Is this the correct volume (Y or N)? >
```

In other words, you read the label assigned to the volume to determine whether the volume you inserted is the next one on which you continued the backup.

If the volume is incorrect, type N and this prompt appears:

```
     Insert volume 2
     Press RETURN when ready
```

If you type Y to indicate the inserted volume is correct, the backup file is continued on that volume and the restoration process continues.

If you want to restore only a portion of a multi-volume backup, remember you do not have to start with the first volume of the backup, but can start with the volume of the backup that contains the first instance of the backup container file containing the files you wish to restore.

Also you can restore from only selected backup files on a volume. For example, if a particular volume contains backup files: 10JAN0001.BAK, 10JAN0002.BAK, 10JAN0003.BAK ... 10JAN0049.BAK, 10JAN0050.BAK, and the files you wish to restore are contained only in files: 10JAN0020.BAK - 10JAN00025.BAK, you can specify restoration only from those files with the following command line:

```
     > restore _dx0/rootdir/10jan002(0-5).bak * _dc0/*/* :builddir
```

Related CIP Commands

arch	Create or modify archive files
backup	Create backup copies of files
mnt	Mount a device

rew

Functional Description

Use this command to rewind a mounted tape.

Command Line Syntax

Mnemonic	rew
Optional parameter	Devicename

Parameters

Devicename	Function	Optional. Use this parameter to specify the name of the device to be rewound.
	Default	SYS$DISK
	Syntax	Type a devicename in the standard device designation syntax. Wildcard symbols are not allowed. If a leading underscore is missing, it is added automatically.

Switches

None.

Examples

```
> rew
```

This command rewinds whatever device the logical name SYS$DISK is assigned to. This command does not output any data.

```
> rew _mt0
```

This command rewinds the device _MT0.

Using Prompts

None.

Notes on Usage

This command only has meaning on TAPE and TAPESPECIAL class devices. An error is returned if REW is executed on any other class of device.

Related CIP Commands

skip	Position a mounted tape

rotor

Functional Description

Use this command to add, delete, or list devices in a rotor list.

Command Line Syntax

Mnemonic	rotor			
Required parameters	Rotor Name Device List			
Optional parameter	Function (:INSERT, :DELETE, :LIST)			
Switches				
Function modification	:auto :class=	:exclude= :uic=	:confirm	:pause

Parameters

Rotor Name	Function	Required (with :INSERT or :DELETE). Use this parameter to specify the name of the rotor list to which devices will be added, deleted, or listed.
	Default	None for :INSERT or :DELETE. For :LIST the default is *, or all rotor lists.
	Syntax	Type any valid rotor list name or logical name which translates to an ASCII string of 8 characters or less.

Device List	Function	Required (with :INSERT or :DELETE). Use this parameter to specify the name(s) of the devices to be added to or deleted from the rotor list.
	Default	None for :INSERT or :DELETE. For :LIST the default is *, or all devices.
	Syntax	Type any valid devicename list. Wildcard symbols are allowed.
Function	Function	Optional. Use this parameter to specify the action to be performed on the devices.
	Default	:INSERT
	Syntax	Type one of the three actions listed below: **:INSERT** inserts the specified devices into the specified rotor list. This is the default if no action is specified. **:DELETE** deletes or removes the specified devices from the rotor list. **:LIST** displays all rotor lists and the devices in each list.

Switches

:auto	Function	Use this switch to perform the desired operation without any confirmation.
	Default	:noauto
	Syntax	Type **:auto**
:confirm	Function	Use this switch to confirm or deny the desired operation on each file or device.
	Default	:noconfirm
	Syntax	Type **:confirm**
:class=	Function	Use this switch to select only those devices that match the Device List parameter and belong to the specified class of devices.
	Default	All devices that match the Device List parameter are selected.
	Syntax	Type **:class=** followed by a list of class types separated by commas.

:exclude=	Function	Use this switch to exclude devicenames from the Device List parameter.
	Default	All devices that match the Device List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of device designations, separated by commas, any one of which may contain wildcard characters.
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when you press any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:uic=	Function	Use this switch to select only those devices that match the Device List parameter and are owned by the specified user or list of users.
	Default	All devices that match the Device List parameter are selected.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> rotor rtrname _tt4,_tt5,_tt6
```

This command adds the devices _TT4, _TT5, and _TT6 to the rotor list called RTRNAME. If the rotor list did not previously exist, ROTOR creates it.

```
> rotor :list
```

This command lists all rotor lists and the devices in each list. The following kind of report is displayed:

```
     Rotor Name  Rotor Elements
     ----------  ------------------------------------------
     MYROTOR2    _TT12      _TT5       _TT8       _TT9

     Rotor Name  Rotor Elements
     ----------  ------------------------------------------
     MYROTOR     _TT0       _TT1       _TT11
```

```
> rotor rtrname * :delete :uic=system :confirm
```

This command deletes from the rotor list called RTRNAME all devices owned by SYSTEM. Each device is listed on the screen for for confirmation by the user before it is deleted.

Using Prompts

```
> rotor
Rotor name   > rtrname
Device list  > _tt4,_tt5,_tt6
```

This performs the same function as the first example.

ROTOR prompts for the rotor name and device list if either the :INSERT or :DELETE functions are used. If :LIST is used, neither the rotor name nor the device list is required because a wildcard is assumed for both parameters.

Notes on Usage

ROTOR is used, usually by the system manager, to create lists of devices which share common characteristics. Devices typically found in a rotor list are a set of devices, any one of which meets a particular user requirement. An example would be a set of printer lines, all serving identical model printers. The user may not care which printer a listing appears on. He/she is only desirous of printing the file on a currently inactive printer.

Another example would be to group a set of modem or other communication lines together into a rotor list. Again, any available line is adequate for communication use. By using rotor lists in conjunction with the ALLOC and DEALLOC commands the user may request use of a communications line based not on its specific name, but rather its availability.

Adding or deleting names to/from a rotor list requires operator privilege. Listing names found in a rotor list requires no privileges.

Related CIP Commands

---

alloc	Find and reserve an available device
dealloc	Deallocate a device which is allocated

scan

Functional Description

Use this command to search files for lines that match a particular pattern.

Command Line Syntax

Mnemonic	scan			
Required parameters	File List Pattern			
Switches				
File selection	:before= :mod	:since=	:exclude=	:uic=
Display control	:context= :log	:number	:header	:pause
Other	:confirm	:exact	:rewind	

Parameters

File List	Function	Required. Use this parameter to specify the names of the files to be scanned.
	Default	None.
	Syntax	Type a list of file designations separated by commas. Wildcard symbols are allowed.

Pattern

Function — Required. Use this parameter to specify the pattern to be scanned for. If any portion of a line matches the pattern you type for this parameter, the line is displayed.

Default — None.

Syntax — Type a pattern string. If spaces are used, the string must be enclosed in double quotation marks. The following characters can be used as wildcard symbols in the text you specify for the pattern:

= The equal sign matches, or signifies, any single character.

# The number sign matches any number of characters or character strings that match the character or character strings immediately to the right of the number sign you type as part of the pattern. For example, if you type #01 as part of the pattern, SCAN searches for a one preceded by any number of zeros.

If you type #(0/2)1, SCAN searches for a one preceded by any number of 0s or 2s (the use of parentheses is explained below).

=# These two characters (typed next to each other) match any string of characters and spaces, i.e., the equal sign matches any single character or space, and the number sign matches any number of characters that match the equal sign.

/ Use the slash to separate alternative variables within the pattern. For example, by typing the (annual/monthly) report you tell SCAN that you want to know of any phrases that read, "the annual report," as well as any that read, "the monthly report."

() Use parentheses to enclose alternative variables in the pattern.

% The percent sign matches the null string.

' Use the single quotation mark as an accept flag (see the note below). In other words, if the number sign, #, equal sign, =, or the single quotation mark, ', is part of the pattern for which SCAN must search, use the single quotation mark to indicate that SCAN should accept the character as it is typed and not as SCAN might otherwise interpret it.

For example, by typing '# on the CIP command line, you indicate that you want SCAN to search for a number sign.

The following characters must be preceded by the backslash character in order to be accepted as part of the pattern. For example, type \? to include a question mark in the pattern.

```
?
!
^

;
'
"
<
>
|
```

SCAN does not distinguish between upper- and lowercase letters.

Switches

:before=	Function	Use this switch to select only those files that match the file list parameter and were created or modified before the specified date and time.
	Default	All files that match the File List parameter are selected.
	Syntax	Type :before= followed by a date or time in the standard date and time syntax.

:confirm	Function	Use this switch to confirm or deny the desired operation on each file.
	Default	:noconfirm
	Syntax	Type **:confirm**
:context=	Function	Use this switch to specify the number of lines of context to be displayed with each line that matches the specified pattern.
	Default	:context=0
	Syntax	Type **:context=** followed by a numeral between 1 and 10.
:exact	Function	Use this switch to specify that SCAN will match only a line that completely satisfies the pattern. That is, it will not match a line that satisfies the pattern and has additional characters.
	Default	:noexact
	Syntax	Type **:exact**
:exclude=	Function	Use this switch to exclude files from the File List parameter.
	Default	All files that match the File List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of file designations separated by commas. Wildcard characters are allowed.
:header	Function	Use this switch to display the full file specification as a header.
	Default	:noheader
	Syntax	Type **:header**
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.

:mod

Function Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
Default :nomod
Syntax Type **:mod**

:number

Function Use this switch to specify that a line number is to be added at the left margin of each line that is displayed.
Default :nonumber
Syntax Type **:number**

:pause

Function Use this switch to stop the display after each screen of information. The display resumes when you press any key.
Default The value specified by the OPTION Command.
Syntax Type **:pause** or **:nopause** to override the default.

:rewind

Function Use this switch to rewind the tape device.
Default :norewind
Syntax Type **:rewind**

:since=

Function Use this switch to select only those files that match the File List parameter and were created or modified since the specified date and time.
Default All files that match the File List parameter are selected.
Syntax Type **:since=** followed by a date or time in the standard syntax.

:uic=

Function Use this switch to select only those files that are part of the given list and are owned by the specified user or list of users.
Default All files that match the specified list are selected.
Syntax Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> scan *.pas writeln
```

This command searches the most recent version of each file in the default directory with an extension of .PAS for any occurrence of the string WRITELN.

When SCAN finds a line that matches the pattern, this kind of report appears:

```
     writeln("Hi mom!");
```

```
> scan *.doc (nixon/ford/carter/reagan) :context=10
```

This command searches the latest version of all files in the default directory with an extension of .DOC for all lines that have any of the names NIXON, FORD, CARTER, or REAGAN. Then it displays the matching lines.

The :context=10 switch prints the 10 lines before the matching line and the 10 lines following it.

```
> scan huxley.txt #=Huxley :exact
```

This command searches only for lines that end with the word HUXLEY.

By default, SCAN looks for the pattern you specify anywhere on a line. But the :exact switch means SCAN searches for a line that matches the pattern in its entirety.

```
> scan *.txt "macro processor" :header :number
```

This command searches the latest version of all files in the default directory with an extension of .TXT, looking for the phrase "macro processor". The :header switch prints the name of each file being scanned. For each file the following kind of report is displayed:

```
     Scanning _DC0/USERS.HANSEN/HANDBOOK.TXT.5
```

The :number switch prints the relative line number of each line in the left margin:

```
Scanning _DC0/USERS.HANSEN/HANDBOOK.TXT.5
7: A study of macro processors is rewarding at two levels.  First from a
13: varied.  Even the term 'macro processor' is not standard; alternative
27: many different macro processors.  Some of them are tied to individual
34: automated using a macro processor.  A macro would be defined for each of
```

The display shows "macro processor" appears four times in the file HANDBOOK.TXT.5—first on line 7, then on lines 13, 27 and 34.

Using Prompts

```
> scan
File list      > *.txt
Pattern        > "macro processors" :header :number
```

This command performs the same function as the previous example.

Notes on Usage

SCAN searches each file specified in the first parameter to find any line(s) containing the pattern specified in the second parameter.

SCAN and WSCAN have the same function. WSCAN is substantially faster than SCAN, but it doesn't allow complex patterns with alternation. If you are searching for a simple pattern, WSCAN will do the job. For complicated patterns, use SCAN.

You cannot specify a pattern that spans more than one line.

The following is a list of patterns and some sample matching strings to give you an idea of the kinds of patterns that can be created.

Pattern	Description	Sample matching strings		
A/B	Matches either A or B	A	B	
A#BC	Matches A followed by any number of B's followed by C.	AC	ABC	ABBC
A#(B/C)D	Matches A followed by any number of B's or C's, followed by D.	AD	ABD	ABCD
A=B	Matches A followed by any single arbitrary character followed by B.	AxB	AyB	AAB
A#=B	Matches A followed by any number of arbitrary characters, followed by B.	AB	AxxB	Ax.AB
'=#='#	Matches the equal sign followed by any number of arbitrary characters followed by a pound sign. Remember that the ' must be accepted with the backslash. \'=#=\'#	=#	=AB#	==##
A(B/%)#C	Matches A followed by either B or the null string followed by any number of C's.	A	ABC	ACCC

Related CIP Commands

wscan		Search files for a pattern

Functional Description

Use this command to send a message to a list of terminals.

Command Line Syntax

Mnemonic	send		
Required parameters	Device List Message		
Switches	:bell	:uic=	:exclude=

Parameters

Device List	Function	Required. Use this parameter to specify the list of devices the given message will be sent to.
	Default	None.
	Syntax	Type a list of device designations separated by commas. Wildcard symbols are allowed.
Message	Function	Required. Use this parameter to specify the message to be sent to each device on the device list.
	Default	None.
	Syntax	Type a message enclosed in double quotation marks. To insert special characters in the line, accept symbols must be used. Escape sequences are not allowed in messages.

Switches

:bell	Function	Use this switch to output a bell along with a message.
	Default	:bell
	Syntax	Type **:nobell**
:exclude=	Function	Use this switch to exclude devices from the Device List parameter.
	Default	All devices that match the Device List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of device designations separated by commas. Wildcard characters are allowed.
:uic=	Function	Use this switch to select only those devices that are part of the given list and are owned by the specified user or list of users.
	Default	All devices that match the specified list are selected.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> send _tt0 hi
```

This command generates the following display on terminal _TT0 if _TT0 has the :broadcast mode enabled:

```
(CIP_USER) hi
```

Note that the process name of the process sending the message is added to the message and a bell is sounded when it appears on _TT0's screen.

> send _tt0,_tt1 "This is a test"

This commnd generates the following display on terminals _TT0 and _TT1 if they have the :broadcast mode enabled:

```
-------------------------
(CIP_USER) This is a test
-------------------------
```

> send * "This is the first line\0aThis is the second line"

This command generates the following display on the screens of all TTY class devices with the :broadcast mode enabled:

```
---------------------------------
(CIP_USER) This is the first line
This is the second line
---------------------------------
```

Note that the \0a accept sequence inserts a new line at that position in the string.

> send _tt0,_tt1 "This is a test" :nobell

This command generates the following display without the bell on terminals _TT0 and _TT1 if they have the :broadcast mode enabled:

```
-------------------------
(CIP_USER) This is a test
-------------------------
```

Using Prompts

```
> send
Device list   > _tt0
Message       > hi
```

This performs the same function as the first example.

Notes on Usage

SEND only works with TTY class devices. All non-TTY class devices in the Device List Parameter are ignored. The message is not sent to any devices with the NOBROADCAST option specified.

For protection reasons this routine converts all escape characters in the given string to spaces.

SEND uses a write timeout. If a user has pressed [CTRL] s on his terminal, any messages sent to it before he presses [CTRL] q will be lost.

Related CIP Commands

None.

serial

Functional Description

Use this command to display on the terminal screen the serial number of your system's hardware.

Command Line Syntax

Mnemonic	serial

Parameters

None.

Switches

None.

Examples

```
> serial
```

This command generates the following kind of report:

```
     Serial number is  Hex: FFFFFFFF   Decimal: -1
```

Using Prompts

None.

Related CIP Commands

version	Display WMCS version information

shlog

Functional Description

Use this command to display logical name assignments for the user process and the system.

Command Line Syntax

Mnemonic	shlog		
Optional parameter	Name		
Switches			
Names to be included in the display	:redundant	:system	:pid=
Format of the display	:pause		

Parameters

Name	Function	Optional. Use this parameter to specify the names to be included in the SHLOG display.
	Default	All logical names are displayed.
	Syntax	Type a single name. Wildcard symbols are allowed.

Switches

:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when you press any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:pid=	Function	Use this switch to display the logical name of the specified process or PID.
	Default	The logical name of your process is displayed.
	Syntax	Type **:pid=** followed by a PID or process name.
:redundant	Function	Use this switch to display all logical names that match the Name parameter, including those that are redundant.
	Default	:noredundant
	Syntax	Type **:redundant**
:system	Function	Use this switch to display logical names that are defined for the whole system.
	Default	:nosystem
	Syntax	Type **:system**

Examples

> shlog

This command generates the following kind of report:

```
--------------------- user process ---------------------
C_EXIT_STATUS = 65535
OPT$PROMPT = USER>
P0 = SYS$DISK/SYSEXE/CIP.EXE
PRINTER = _tt1
SYS$CIP = SYS$DISK/SYSEXE/CIP.EXE
SYS$ERROR = _TT0
SYS$HOME = SYS$DISK/USER/
SYS$INPUT = _TT0
SYS$OUTPUT = _TT0
SYS$PROMPT = _TT0
SYS$RESULT = 0
SYS$USERNAME = USER
--------------------- system ---------------------
DF = dmnt _dx0 :auto
DS = dir :size :create :head
HOME = cd 'sys$home'
KILL = pstat :kill
ME = stat :uic='sys$username'
MF = mnt _dx0
NEWS = type sys$disk/syslib/news.txt :pause
STAT = pstat :head :sysstat :status :size :owner :prior :times :sched
SYS$DISK = _DC0
SYS$MATH = SOFTWARE
SYS$MODEL = SYSTEM156
SYS$PIPE = sys$disk/sysdsr/pip02$156.dsr
SYS$PRINT = _pq0
SYS$SYSNAME = System name
```

The display consists of a series of logical names and their translations. The logical names listed under USER PROCESS are associated with the logon CIP of the user. The logical names listed under SYSTEM are associated with the computer system as a whole and are in effect for all users on the system.

If you use the CP command from within a VEW editing session, then execute SHLOG, the following kind of display appears:

```
----------------------- current process 0-----------------------
SYS$ERROR = _TT0
SYS$INPUT = _TT0
SYS$OUTPUT = _TT0
SYS$PROMPT = _TT0
SYS$RESULT = 0
----------------------- user process -----------------------
OPT$PROMPT = USER>
P0 = SYS$DISK/SYSEXE/CIP.EXE
SYS$CIP = SYS$DISK/SYSEXE/CIP.EXE
SYS$HOME = SYS$DISK/USER/
SYS$USERNAME = USER
VEW$MEMORY = shlog.exp
----------------------- system -----------------------
DF = dmnt _dx0 :auto
DS = dir :size :create :head
HOME = cd 'sys$home'
KILL = pstat :kill
ME = stat :uic='sys$username'
MF = mnt _dx0
NEWS = type sys$disk/syslib/news.txt :pause
STAT = pstat :head :sysstat :status :size :owner :prior :times :sched
SYS$DISK = _DC0
SYS$MATH = SOFTWARE
SYS$MODEL = SYSTEM156
SYS$PIPE = sys$disk/sysdsr/pip02$156.dsr
SYS$PRINT = _pq0
SYS$SYSNAME = System name
```

The logical names listed under CURRENT PROCESS 0 are associated with the CIP invoked from VEW. Again, the logical names associated with the logon CIP of the user are listed under USER PROCESS and those of the system are listed under SYSTEM.

> shlog :redundant

If you are in VEW, issue a CP command and execute SHLOG, this display appears:

```
---------------------- current process 0----------------------
SYS$ERROR = _TT0
SYS$INPUT = _TT0
SYS$OUTPUT = _TT0
SYS$PROMPT = _TT0
SYS$RESULT = 0
---------------------- current process -1----------------------
SYS$ERROR = _TT0
SYS$INPUT = _TT0
SYS$OUTPUT = _TT0
SYS$RESULT = 0
---------------------- user process ----------------------
OPT$PROMPT = USER>
P0 = SYS$DISK/SYSEXE/CIP.EXE
SYS$CIP = SYS$DISK/SYSEXE/CIP.EXE
SYS$ERROR = _TT0
SYS$HOME = SYS$DISK/USER/
SYS$INPUT = _TT0
SYS$OUTPUT = _TT0
SYS$PROMPT = _TT0
SYS$RESULT = 0
SYS$USERNAME = USER
VEW$MEMORY = shlog.exp
---------------------- system ----------------------
DF = dmnt _dx0 :auto
DS = dir :size :create :head
HOME = cd 'sys$home'
KILL = pstat :kill
ME = stat :uic='sys$username'
MF = mnt _dx0
NEWS = type sys$disk/syslib/news.txt :pause
STAT = pstat :head :sysstat :status :size :owner :prior :times :sched
SYS$DISK = _DC0
SYS$MATH = SOFTWARE
SYS$MODEL = SYSTEM156
SYS$PIPE = sys$disk/sysdsr/pip02$156.dsr
SYS$PRINT = _pq0
SYS$SYSNAME = System name
```

The :redundant switch displays logical names defined at different levels of the user's processes under the headings CURRENT PROCESS 0 and CURRENT PROCESS -1.

Using Prompts

None.

Notes on Usage

If SHLOG is used to display the equivalence of a name which has had no equivalence assigned to it, the following kind of display appears:

```
     >shlog win
     WIN not found.
```

In other words, if no equivalence has been assigned to the name, then the name is equivalent to itself.

Related CIP Commands

assign	Assign a logical name

shutdown

Functional Description

Use this command to terminate all WMCS activity before resetting or turning off the system.

Command Line Syntax

Mnemonic	shutdown	
Required parameter	Delay	
Optional parameter	Message	
Switches	:broadcast	:log

Parameters

Delay	Function	Required. Use this parameter to specify the amount of time that will pass before SHUTDOWN is executed.
	Default	None.
	Syntax	Type a relative time or an absolute time. (Do not specify a time before the current time.)
Message	Function	Optional. Use this parameter to send a message to every terminal on the system, usually to explain the reason for SHUTDOWN.
	Default	None.
	Syntax	Type any valid string of characters. More than one word must be enclosed in double quotation

marks.

Switches

:log
Function — Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
Default — The value specified by the OPTION command.
Syntax — Type :nolog or :log to override the default.

:reboot
Function — Use this switch to reboot the machine, as if the user pressed the reset button on the machine.
Default — :noreboot
Syntax — Type **:reboot**

Examples

```
> shutdown +1
```

This command generates a the following kind of display:

```
     Shutting down Mephistopheles:
       There are 13 processes
       There are 0 active queues
     Shutdown in 1 hour (Y or N)? > n
```

The SHUTDOWN display tells the user what system is shutting down (using the logical name SYS$SYSNAME), how many processes are running on the machine, and how many queues are executing something. The user is also asked to confirm the command and the specified time delay.

If you strike Y, all active queues are halted (this means as the active entries terminate no new entries are started) and a message like the following is sent to all terminals:

```
(CIP_USER) System shutdown in 1 hour
```

After the time delay expires, all active processes are killed and all devices are dismounted. Note that entries in QUEUE class devices are not lost unless the :NORESTART Switch is specified with the MNT command when the QUEUE devices are remounted after the boot.

```
> shutdown +0 :reboot "To load a new operating system"
```

This command shuts down the system immediately and sends a message like the following to each terminal:

```
(CIP_USER) System shutdown in 0 seconds
To load a new operating system
```

After the system is shut down, the :reboot switch causes it to reboot as if the user pressed the reset button.

Using Prompts

```
> shutdown
Delay          > +1
```

This performs the same function as the first example.

Notes on Usage

SHUTDOWN ensures that user has all of the necessary privileges to shut the system down. If not, an error is reported.

Related CIP Commands

None.

skip

Functional Description

Use this command to position a mounted tape.

Command Line Syntax

Mnemonic	skip
Required parameter	Devicename
Optional parameters	Tape Unit (used only if Function is :FILES) Function (:FILES, :BOT, :EOT)

Parameters

Devicename	Function	Required. Use this parameter to specify the tape device.
	Default	None.
	Syntax	Type any valid device designation. Wildcard symbols are not allowed.
Tape Unit	Function	Optional. Use this parameter with :FILES to specify the number of files to skip. A positive number skips forward (toward the end of the tape), while a negative number skips backward (toward the beginning of the tape). If the function is not :FILES, this parameter is ignored.
	Default	1
	Syntax	Type a positive or negative integer.

Function	Function	Optional. Use this parameter to specify the action to be performed on the tape device.
	Default	:FILES
	Syntax	Type one of the three actions listed below: **:BOT** positions the device to the beginning of the tape. **:EOT** positions the device to the logical end of the tape. **:FILES** positions the device a given number of files, either forward or backward.

Switches

None.

Examples

```
> skip _mt0
```

This command skips the tape on device _MT0 forward one file.

```
> skip _mt0 :bot
```

This command rewinds the tape. It is equivalent to the REW command.

```
> skip _ct0 -4 :files
```

This command positions the tape backward 4 files from its current position.

Using Prompts

```
> skip
Devicename    > _mt0
```

This performs the same function as the first example.

Related CIP Commands

rew	Rewind a mounted tape

sort

Functional Description

Use this command to sort the contents of one or more files and/or merge them.

(This version of SORT was written by P.J. Fraser of the Software Development Group, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1.)

Command Line Syntax

Mnemonic	sort			
Required parameter	File to be sorted			
Optional parameter	File to be merged			
Switches				
	+ASCending	SKip=	+DEScending	STop=
	-Duplicate	MemorySize=	-Unixfile	RecordSize=
	+Verbose	RejectFile=	Error=	KeyFile=
	Ignoreleading=	Raw=	Key=	
Key options	+ASCending	+DECimal	+DEScending	+HEXadecimal
	+Ascii	+BINary	+Caseless	+FloatingPoint
	+DICtionary	+BinarySigned	+Numeric	+BinaryFloating
	+OCTal	+Location	+Width	FRom=
	Before=	Length	SKip=	STop=
	+Ignoreleading=	Offset=	Field=	-Error

Parameters

Sort File

Function: Required. Use this parameter to specify the file to be sorted.

Default: None.

Syntax: Type a file designation. No wildcard symbols are allowed.

Merge File

Function: Optional. Use this parameter to specify the file to be sorted and then merged with the file specified in the first parameter.

Default: None.

Syntax: Type a file designation. No wildcard symbols are allowed.

Switches

Note that switches for this command can be abbreviated. Valid abbreviations must contain the uppercase letters shown in the syntax description. For example, -Duplicate can be abbreviated -d, or -du, or -dup.

+ASCending

Function: Use this switch to sort the file(s) in ascending order.

Default: The specified files are sorted in ascending order.

Syntax: Type **+ASCending** to sort in ascending order. Type **-ASCending** when you do not want files sorted in ascending order.

+DEScending

Function: Use this switch to sort the file(s) in descending order.

Default: The specified files are sorted in ascending order.

Syntax: Type **+DEScending** to sort in descending order. Type **-DEScending** when you do not want files sorted in descending order.

-Duplicate

Function: Use this switch to discard duplicate records in the sort.

Default: Duplicate records appear in the sorted material.

Syntax: Type **-Duplicate** to discard duplicate records in the output. Type **+Duplicate** when you want duplicate records to appear in the output.

-Unixfile

Function: Use this switch to separate output lines with ASCII nulls.

Default: The output appears in a UNIX format, i.e., lines of text are separated by line feed characters instead of ASCII nulls.

Syntax: Type **-Unixfile** for output in an ASCII format. Type **+Unixfile** for output in a UNIX format.

+Verbose

Function: Use this switch to print statistics about the sort when the sort is completed.

Default: No statistical report is printed.

Syntax: Type **+Verbose** to see statistics about the sort. Type **+Verbose** when you do not want a statistical report.

Error=

Function: Use this switch to indicate how errors should be handled by SORT. The five methods of error handling are stop, report, ignore, first, and last.

Default: Error=ignore

Syntax: Type **Error=** followed by one of the five methods of error handling. An explanation of each method follows:

Type **error=stop** when you want SORT to stop processing upon finding an erroneous record.

Type **error=report** when you want a message written to the terminal screen whenever an erroneous record is found. The file in which the record is located is noted when more than one file is being sorted.

Type **error=ignore** when you want SORT to ignore the fact that a record is erroneous. Such records are included in the sort.

Type **error=first** when you want all erroneous records sorted as a group and printed at the beginning of the output.

Type **error=last** when you want all erroneous records sorted as a group and printed at the end of the output.

Ignoreleading= Function Use this switch to specify characters at the front of the key to be ignored by SORT.

Default Ignoreleading=\t\n, i.e., leading blanks, tabs, and new lines are ignored.

Syntax Type **Ignoreleading=** followed by the string of characters to be ignored.

SKip= Function Use this switch to specify the characters SORT may skip when searching for a key.

Default SKip=\t\n, i.e., leading blanks, tabs, and new lines are skipped.

Syntax Type **SKip=** followed by the string of characters to be skipped.

STop= Function Use this switch to specify the characters which mark the end of a key field.

Default STop=\t\n, i.e., blanks, tabs, and new lines mark the end of the key field.

Syntax Type **STop=** followed by the string of characters which mark the end of the key field.

MemorySize= Function Use this switch to specify the maximum amount of memory SORT may use.

Default MemorySize=120 (kilobytes)

Syntax Type **MemorySize=** followed by a number of kilobytes.

RecordSize= Function Use this switch to specify the maximum length of any input record.

Default RecordSize=300 (words)

Syntax Type **RecordSize=** followed by a number of words. RecordSize may not be greater than 7000.

RejectFile= Function Use this switch to specify the file to which SORT writes erroneous records.

Default Erroneous records appear in the output.

Syntax Type **RejectFile=** followed by a file designation. If you type only **RejectFile=** without typing a file designation, the erroneous records are discarded.

Raw=	Function	Use this switch to specify the length of the input record. This causes SORT to break the input up into records of the specified size, regardless of the actual size of the records in the input file(s). The length may be given in terms of words, bytes, etc.
	Default	The length of the input record is the same as its actual length in the specified file(s) being sorted.
	Syntax	Type **Raw=** followed by a value. Note that the length must resolve to an exact number of bytes. For example, the following would be invalid: Raw=5bits

KeyFile=	Function	Use this switch to specify a file containing key strings. Each line of this file contains a separate key string.
	Default	None.
	Syntax	Type **KeyFile=** followed by a file designation.

Key=	Function	Use this switch to specify which key to sort by. More than one Key= option may be specified.
	Default	The entire record.
	Syntax	Type **Key=** followed by one or more of the key options described below, separated by commas.

Note that the following are key options, i.e., ways of specifying where keys are located in the records being sorted. Each set of key options describing a single key must be preceded by Key=, and the options must be separated by commas.

+ASCending	Function	Use this option to override -asc or +des on the command line and sort in ascending order.
	Default	+ASCending
	Syntax	Type **+ASCending**

+DEScending	Function	Use this option to override -des or +asc on the command line and sort in descending order.
	Default	+ASCending
	Syntax	Type **+DEScending**
+Ascii	Function	Use this option to sort in ASCII collating sequence. +ASCII treats the input as if it consists of unsigned binary characters.
	Default	+Ascii
	Syntax	Type **+Ascii**
+Caseless	Function	Use this option to require SORT to ignore case when comparing ASCII characters. Thus, keys beginning with "A" or "a" are grouped together. Use this option with ASCII input only.
	Default	Uppercase is sorted before lowercase.
	Syntax	Type **+Caseless**
+DICtionary	Function	Use this option to require SORT to ignore everything that isn't a letter or a digit. Use this option with ASCII input only.
	Default	None.
	Syntax	Type **+DICtionary**
+Numeric	Function	Use this option to sort numerical ASCII key fields. If the number begins with a zero, it is sorted as octal; otherwise, it is sorted as decimal.
	Default	None.
	Syntax	Type **+Numeric**
+OCTal	Function	Use this option to sort octal numerical ASCII fields. The number does not need a leading zero, but may consist only of the digits 0 to 7.
	Default	None.
	Syntax	Type **+oct**

+DECimal	Function	Use this option to sort decimal numerical ASCII fields. Numbers are sorted as decimals regardless of leading zeroes.
	Default	None.
	Syntax	Type **+DECimal**
+HEXadecimal	Function	Use this option to sort hexadecimal numerical ASCII fields.
	Default	None.
	Syntax	Type +HEXadecimal
+BINary	Function	Use this option to sort binary numerical ASCII fields.' The number may consist only of ones and zeroes.
	Default	None.
	Syntax	Type **+BINary**
+FloatingPoint	Function	Use this option to sort floating point numerical ASCII fields.
	Default	None.
	Syntax	Type **+FloatingPoint**
+BinarySigned	Function	Use this option to sort signed binary numerical ASCII fields. The +bs key may be a maximum of 32 bits long.
	Default	None.
	Syntax	Type **+BinarySigned**
+BinaryFloating	Function	Use this option to sort floating binary numerical ASCII fields. The +bf key may be a maximum of 64 bits long.
	Default	None.
	Syntax	Type **+BinaryFloating**
+Location	Function	Use this option to sort according to the location of a key field in the record.
	Default	None.
	Syntax	Type **+Location**

+Width

Function Use this option to sort according to the width of a particular field in the record. The wider the field, the later it comes in the collating sequence.
Default None.
Syntax Type **+Width**

FRom=

Function Use this option to indicate where the key starts in a record.
Default FRom=0, i.e. the key begins in column 0 of the record.
Syntax Type **FRom=** followed by a column number or the number of words, bytes, bits, etc.

Before=

Function Use this option to specify that the key is located before a given position.
Default Before=length of record, i.e. the key may be anything before the end of the record.
Syntax Type **Before=** followed by a record position.

Length=

Function Use this option to specify the length of the key.
Default Everything after the Before= position, unless Field= is specified.
Syntax Type **Length=** followed by a length value in bits, bytes, words, etc.

SKip=

Function Use this option to indicate all the characters that may be skipped before beginning to search for the field. The Field= option must be specified in conjunction with this option.
Default SKip= as specified in the command line.
Syntax Type **SKip=** followed by a character string.

STop=

Function Use this option to indicate the characters which mark the end of a key. The Field= option must be specified in conjunction with this option.
Default STop= as specified in the command line.
Syntax Type **STop=** followed by a character string.

-Ignoreleading=	Function	Use this option to specify all characters that may be ignored at the beginning of a key field after the SKip= characters have been skipped. This option may be used even if Field= is not specified for the key.
	Default	IgnoreLeading= as specified in the command line.
	Syntax	Type **IgnoreLeading=** followed by a character string. Type **-IgnoreLeading** to specify that no characters be ignored once a key field has been located.
Offset=	Function	Use this option to indicate that the key begins at the given offset from the beginning of the field. This option takes effect once the SKip= characters and the IgnoreLeading= characters have been skipped.
	Default	Offset=0
	Syntax	Type **Offset=** followed by a number.
Field=	Function	Use this option to specify which field in a list is the actual key.
	Default	Field=0
	Syntax	Type **Field=** followed by a number.
-Error	Function	Use this option to indicate that the entire record should NOT be marked as erroneous even if a particular key is in error.
	Default	The entire record is marked as erroneous if a key is in error.
	Syntax	Type **-Error**

Examples

> sort clients.000

This command sorts the contents of the file CLIENTS.000 in ascending ASCII order and writes the result to the terminal screen.

```
> sort clients.000 >output.000 key=il=" "
```

This command sorts the contents of CLIENTS.000 in ascending ASCII order and puts the result in the file OUTPUT.000. The IgnoreLeading= (il=) option tells SORT to ignore any leading blanks that appear at the beginning of each line.

```
> sort clients.000 key=fr=41,b=48,+des,+n key=fr=0,b=30
```

This command sorts the contents of CLIENTS.000 according to two keys. The first key is numeric, beginning after character 41 and ending at character 48 (the initial character of the record is character 0). The records are sorted according to this key in descending (+des) numerical (+n) order.

Records with the same first key are sorted according to a second key, which begins with the initial character of the record and ends at character 30.

```
> sort clients.000 newclients.000 >customers.000
```

This command sorts two files, CLIENTS.000 and NEWCLIENTS.000, in ascending ASCII order and then merges them. The result is put in the file CUSTOMERS.000.

Using Prompts

None.

Notes on Usage

Unless you specify an output file for SORT, the output from the command is written only to the terminal screen provided your terminal screen is designated SYS$OUTPUT).

SORT sorts the contents of one or more files according to one or more keys. Key fields may overlap one another if desired.

Filenames, switches and options can be typed on the command line in either upper- or lowercase and in any order.

Key strings that contain blanks must be surrounded by double quotation marks.

The KeyFile= option may appear anywhere on the command line that Key= can. Each line in the KeyFile is a separate key string. These key strings are the same as those given with Key=, but without enclosing quotation marks. The order of the key strings in the KeyFile determines their order of importance.

Every key used in the sorting process should have one Key= specification. For example, if a file of students is to be sorted by student name and student number within name, there must be a Key= describing where the name is found in each record and a second Key= describing where the student number is found. The order in which Key= options appear on the command line determines their order of importance.

SORT can also sort standard input either through a pipe or through the left angle bracket, <. The standard input cannot be a TTY.

Key location options (Before=, FRom=, etc.) recognize the following units typed after the number:

Ascii characters, bytes, and characters equivalent to eight bits. This is the default if no units are specified after the number.

Bcddigits equivalent to four bits.

Bits equivalent to a single bit.

Doublewords equivalent to two full words (32 bits).

Nibbles equivalent to half a byte (four bits).

Words equivalent to a full word (16 bits).

All locations are indexed from zero. Thus, **FRom=10c** indicates that the key begins on the eleventh character of the record.

Key location options may be concatenated into one long position. Thus, **FRom=10w1by2b** indicates that the field begins with the third bit in the second byte of the 11th word of the record.

When a position consists of more than one key location option, each option MUST have a unit specified explicitly. Thus, **FRom=10** defaults to "10c" but **FRom=10w1** is invalid.

The position descriptors may also be delimited by spaces, commas, or plus signs. For example, **FRom="10w 1by" ,Before="11w,1by"** or **Length=3w+2bits**.

Negative numbers are also allowed in key location options. For example, **Before=10w-1c.** However, the final result must be positive when translated into bits.

The numbers given in key loction options will be taken as octal if they have a leading zero or as hexadecimal if preceded by 0x; otherwise, the numbers will be taken as decimal. Thus, **Before=0x10w,010by,10bits** is equivalent to **Before=16w,8by,10bits.**

In general, keys may begin on any bit boundary; however, if the Field= or Ignore= option appears in the key string, the key can only begin on a byte boundary.

Errors may occur in the course of searching for a key when:

> The key is found with a length less than or equal to zero bits.
>
> The incorrect format is specified. For example, the key option is +n, but the record contains non-numeric data.
>
> The key is too short. For example, Length=9bytes is specified, but the record is only two bytes long.
>
> The key is missing. For example, FRom=30c is specified when the record is only 10 characters long.

If any of these situations occur, the key is in error. This would normally imply that the entire record is erroneous; however, if the -Error key option is specified in the key-string for the erroneous key, the error is disregarded and the record is not marked as erroneous.

If the record is marked as erroneous, the Error= disposition is checked. If Error=Stop is specified on the command line, SORT terminates.

If the Error= disposition is not Stop, SORT attempts to include the record in the sorting process. Thus SORT tries to correct the erroneous key.

If the key is merely shorter than expected, it is padded to the correct length by adding zero bits on the right. With character data, this means appending ASCII nulls (octal 000); with numeric data, this essentially means adding 0 characters. The expanded key is added to the key information on the end of the record so that it can participate in the sorting process.

If the key is missing or invalid, a key consisting of zero bits is supplied. With character data, this means a string of ASCII nulls; with numeric data, this means the number zero. The supplied key is added to the key information on the end of the record.

If the Error= code is Report or Ignore, erroneous records are written to the RejectFile (if one exists) after all the records have been sorted. Thus the RejectFile is always sorted, making it easier to merge erroneous records with the correct ones, if desired.

If the Error= code is First or Last, erroneous records appear in the appropriate location within the sorted output and are NOT written to the RejectFile.

If -Error is specified in the key-string, the erroneous key is padded or supplied with zero bits without marking the entire record as erroneous. The sort takes place using the zero value. Since the record is not regarded as erroneous, it is not written to the RejectFile.

When extracting key information from a record, SORT considers the key options in the following order of importance:

Before= to determine the absolute possible end of the key

FRom= to determine the absolute possible beginning of the key

Field= to determine the field of the key; also involves jumping over an appropriate number of "stop" and "skip" characters

Ignore= to ignore unwanted characters at the beginning of the actual key

Offset= to determine an offset from the first unignorable character

Length= to obtain a key of the required length, once the various unwanted items at the beginning of the key have been passed over.

The SKip, STop, and IgnoreLeading key options recognize the following C escape sequences:

\n	new line
\r	carriage return
\f	form feed
\b	backspace
\v	vertical tab
\t	horizontal tab
\\	backslash
\ddd	character represented by octal number ddd

Note that the character following the backslash can be typed in either upper- or lowercase.

Related CIP Commands

wsort	WICAT sort program

Functional Description

Use this command to report space allocation on a device.

Command Line Syntax

Mnemonic	sp		
Optional parameter	Device List		
Switches			
Display format	:bytes	:kbytes	:pause
Device selection	:confirm	:exclude=	:uic=

Parameters

Device List	Function	Optional. Use this parameter to specify the device to be displayed.
	Default	The default device.
	Syntax	Type a list of mounted devicenames separated by commas.

Switches

:bytes	Function	Use this switch to display the amount of space in bytes.
	Default	The amount of space is displayed in sectors.
	Syntax	Type **:bytes**
:confirm	Function	Use this switch to confirm or deny the desired operation on each device.
	Default	:noconfirm
	Syntax	Type **:confirm**
:exclude=	Function	Use this switch to exclude devices from the Device List parameter.
	Default	All files or devices that match the Device List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of device designations separated by commas. Wildcard characters are allowed.
:kbytes	Function	Use this switch to display the amount of space in Kbytes.
	Default	The amount of space is displayed in sectors.
	Syntax	Type **:kbytes**
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when you press any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:uic=	Function	Use this switch to select only those devices that are part of the given list and are owned by the specified user or list of users.
	Default	All files that match the specified list are selected.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> sp
```

If the default disk is __BART_DS0, this command displays the following kind of report:

```
     Disk usage of __BART_DS0
     Free =       48287 sectors.
     Used =      372711 sectors.
     Bad  =           2 sectors.
     Total=      421000 sectors.
```

The devicename appears in the first line of the display. The second line shows the number of unused sectors, i.e., sectors not assigned to any file. Line 3 shows the number of used sectors, i.e., sectors containing data assigned to files. Line 4 shows the number of bad sectors. A bad sector is one the WMCS tried to use but found it unreliable. The WMCS will not write to these sectors. The last line of the display shows the total number of sectors.

```
> sp _df0
```

If _DF0 is mounted, SP generates this kind of report:

```
     Disk usage of _DF0
     Free =        1000 sectors.
     Used =         232 sectors.
     Bad  =           0 sectors.
     Total=        1232 sectors.
```

This command displays the space available on _DF0, whether or not _DF0 is the default disk.

Using Prompts

None.

Notes on Usage

The disk you specify in the Device List Parameter must be mounted before SP can report on it.

Related CIP Commands

cd	Change the default directory and device
def	Display the name of the default device and directory
dev	Display the list of mounted devices

sscopy

Functional Description

Use this program to transfer files from one WMCS system to another across serial communication lines.

Command Line Syntax

Mnemonic	sscopy	
Required parameters	Function	(B, S, SA, R, RA)
	Port	
	Source	(Required only if function is not B)
	Destination	(Required only if function is not B)

Parameters

Function	Function	Required. Use this parameter to specify the action performed as part of the transer of the file(s).
	Default	None.

Syntax Type one of the following five actions:

B (background) This function is performed on the remote system. The SSCOPY with this function receives direction from the SSCOPY on the local system.

R (receive) Use this function on the local system to request files on the remote system be received by the local system, i.e., sent from the remote system to the local system.

S (send) Use this function on the local system to request files on the local system be sent to the remote system.

RA (receive hexadecimal ASCII) This function is the same as R above except all data are converted to hexadecimal ASCII before being transmitted, and converted back to binary when they are received.

SA (send hexadecimal ASCII) This function is the same as S above except all data are converted to hexadecimal ASCII before being transmitted, and converted back to binary when they are received.

Port

Function Required. Use this parameter to specify the port of the communication channel. This port must already be mounted.

Default None.

Syntax Type the devicename of a serial communication port.

Source

Function Required. Use this parameter to specify the local filenames to be sent to the remote system, or to specify the remote filenames to be received on the local system. Do not use this parameter with B.

Default None.

Syntax Type a list of file designations separated by commas. Wildcard symbols are allowed.

Destination	Function	Required. Use this parameter to specify the filenames to which the source files are copied. Field substitution occurs with this parameter.
	Default	None.
	Syntax	Type a destination file designation.

Switches

None.

Examples

The following procedure for executing SSCOPY assumes you have a serial cable connected to serial port _TT1 on the system at which you are working (the local system), the cable is connected to serial port _TT15 on another system (the remote system), such port characteristics as baud rate, etc., are the same for both ports (use the DSTAT command to achieve this), and you want to receive a file from the remote system to the system on which you are working.

Step 1 Type the following on the CIP command line and strike [RETRN]:

```
talkt _ttl
```

The cursor jumps to the left margin of the line below the CIP command line. You are now conversing with the remote system through _TTl on your system to _TT15 on the remote system.

Step 2 Strike [RETRN] and log on to the remote system.

Step 3 Type the following on the command line of the remote system and strike [RETRN].

```
sscopy b _tt15
```

This command-line character string initiates the background copy of SSCOPY on the remote system. This background version of SSCOPY watches for instructions to be received on _TT15.

Step 4 Type the following:

```
[CRTL] z [CTRL] e
```

This sequence of characters breaks the connection between your terminal and _TT1. Characters you type after this point are not transmitted to the remote system. When you strike the [CTRL] e, the cursor reappears next to the CIP prompt and you are operating again on your home system.

Step 5 Type the following kind of character string on the CIP command line and strike [RETRN].

```
sscopy r _tt1 _ds0/language/report.doc *
```

This character string indicates you want to copy the file named _DS0/LANGUAGE/REPORT.DOC from the remote system (R) and assign it to your default directory. Note that the source file designation (_DS0/LANGUAGE/REPORT.DOC) corresponds to the name of devices and directories on the remote system. The destination file designation (*) corresponds to the name the file receives on the local system.

The following kind of report appears on your screen:

```
File: REPORT.DOC Received.
```

Following this report, the cursor appears next to the CIP command prompt for the local system. The background version of SSCOPY on the remote system continues to wait for instruction. You can leave it waiting as long as you wish, or you can follow these steps to terminate it:

Step 1 Type the following on the CIP command line:

```
talkt _tt1
```

This command puts you in communication with _TT15 on the remote system through _TT1 on your system.

Step 2 Type [CTRL] c to terminate SSCOPY on the remote system.

The cursor appears next to the CIP prompt on the remote system.

Step 3 Type **log** and strike [RETRN] to log off the remote system.

Step 4 Type [CTRL] z [CTRL] e to break the connection between _TT1 and _TT15 on the two systems.

The cursor now appears next to the CIP prompt on the local system.

```
> sscopy s _ttl plans.doc _ds0/archive/
```

This example assumes _TTl on the local system is connected to some unspecified port on a remote system, and a background version of SSCOPY is running on that unspecified port.

SSCOPY sends the latest version of the file PLANS.DOC from the default directory on the local system through the port _TTl on the local system. A file is created on the remote system named _DS0/ARCHIVE/PLANS.DOC.

Using Prompts

```
> sscopy
COMMAND> b
PORT> _tt15
```

Note that SSCOPY does not prompt for the source and destination since those parameters are not allowed when the function is B (background).

```
> sscopy
COMMAND> r
PORT> _ttl
SOURCE> _ds0/language/report.doc
DESTINATION> *
```

This command receives the file REPORT.DOC from the remote system through the port named _TTl on the local system. The file is placed in the default directory on the local system.

Notes on Usage

Use SSCOPY to transfer files from one WMCS system to another. This program is the precursor of USSCOPY. Note that SSCOPY and USSCOPY are not compatible. That is, you cannot set up SSCOPY on one machine and communicate with it using USSCOPY, or vice versa. If both machines have USSCOPY, you should use it instead of SSCOPY since USSCOPY is much more reliable.

SSCOPY continuously monitors the integrity of the transmission, using checksums on the data packets. It reports an error and quits if a transmission error is detected.

If you have trouble sending or receiving files, you can try lowering the baud rate on the sending and receiving serial ports. Use DSTAT to adjust the baud rate.

Related CIP Commands

copy	Copy files
talkt	Communicate directly with another device
usscopy	Updated system to system copy

submit

Functional Description

Use this command to submit processes for batch execution in a background mode and to perform other batch-queue management functions.

Command Line Syntax

Mnemonic	submit	
Required parameter	Program or Entry List	(Required only with :INSERT) (Required only with :WAKE, :HOLD, :RESTART, :MODIFY, :DELETE; Optional with :LIST)
Optional parameter	Function	(:CLOSE, :DELETE, :HALT, :HOLD, :INSERT, :LIST, :MODIFY, :OPEN, :RESTART, :START, :WAKE)

Switches

Process control	:owner=	:sysinput=	:priority=	:sysoutput=
	:siteid=	:timeslice=	:syserror=	:privilege=
	:node=			
Queue	:entdelete	:qpriority=	:enthold	:queue=
	:inform	:when=		
Other	:auto	:log	:confirm	:pause
	:full			

Parameters

Program

Function

Required. Use this parameter with :INSERT to specify the program to be executed.

For :WAKE, :HOLD, :RESTART, :MODIFY, and :DELETE, use this parameter to list the entry numbers to be acted upon.

For :LIST, use this optional parameter to specify the entry numbers to be listed.

For :CLOSE, :OPEN, :START, and :HALT this parameter is not used.

Default

None for :INSERT, :WAKE, :HOLD, :RESTART, :MODIFY, and :DELETE. For :LIST the default is "1-", which is all entries.

Syntax

For :INSERT, type a program and all of its parameters. If it is more than one word it must be surrounded by double quotation marks.

For :WAKE, :HOLD, :RESTART, :MODIFY, :DELETE, and :LIST functions, type a list of entry number ranges separated by commas.

Function

Function

Optional. Use this parameter to specify the action to be performed on the given queue.
These are the 11 functions:

:CLOSE closes the specified queue. No more entries may be inserted into that queue. All queued up entries continue to be processed. Operator privilege is required to perform this action.

:DELETE deletes a list of entries from the specified queue.

:HALT halts the specified queue. No more entries in the queue can be executed. Active entries continue to run. Entries can still be inserted into the queue, but they won't be executed. Operator privilege is required to perform this action.

:HOLD keeps a list of entries from executing. They maintain their position in the queue, but they are skipped over when their turn arrives.

:INSERT inserts a program to be executed into the specified queue. This is the default if no function is specified.

:LIST displays the entries in a given queue. It also lists their status.

:MODIFY modifies a list of entries in a given queue. Any parameters specified when an entry is inserted can be modified later with this command.

:OPEN is the opposite of :CLOSE. It allows new entries to be entered into the queue. Operator privilege is required to perform this action.

:RESTART restarts a list of active entries. The given processes are terminated and then restarted. If you :HALT a queue and then :RESTART, all activity in the queue (including active entries) is stopped.

:START is the opposite of :HALT. It allows entries to be executed again. Operator privilege is required to perform this action.

:WAKE is the opposite of :HOLD. It allows a list of held entries to be executed.

Default :INSERT

Syntax Type one of the 11 actions listed above.

Switches

:auto	Function	Use this switch to perform the desired operation without any confirmation.
	Default	:noauto
	Syntax	Type :auto

:confirm	Function	Use this switch to confirm or deny the desired operation on each file or device.
	Default	:noconfirm
	Syntax	Type **:confirm**
:entdelete	Function	Use this switch to specify whether the given entry is to be deleted from the queue when the entry has completed execution. If the entry is not deleted, it is placed in a hold state.
	Default	:entdelete
	Syntax	Type **:noentdelete**
:enthold	Function	Use this switch to specify whether the given entry should be placed in a hold state immediately upon entry into the queue. This means it is impossible for the entry to be executed before it is awakened.
	Default	:noenthold.
	Syntax	Type **:enthold**
:full	Function	Use this switch to display all information about the file, device, or process.
	Default	:nofull
	Syntax	Type **:full**
:inform	Function	Use this switch to send a message to the queuer that the entry has been completed.
	Default	:noinform
	Syntax	Type **:inform**
:log	Function	Use this switch to specify whether log messages are displayed.
	Default	The value specifed by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:node=	Function	Use this switch to specify the node to execute the process on.
	Default	The process is executed on the current node.
	Syntax	Type **:node=** followed by a site ID number.

:owner=

Function Use this switch to specify the UIC the process is to have.
Default The UIC of the process doing the SUBMIT.
Syntax Type **:owner=** followed by a UIC or a username. (Usernames are converted to the UIC they represent.)

:pause

Function Use this switch to stop the display after each screen of information. The display resumes after the user presses any character.
Default The value specified by the OPTION command.
Syntax Type **:pause** or **:nopause** to override the default.

:priority=

Function Use this switch to specify the priority the process is to execute at.
Default The priority of the process doing the SUBMIT.
Syntax Type **:priority=** followed by an integer between 0 and 15, with 0 as the highest priority and 15 as the lowest.

:privilege=

Function Use this switch to set the privileges for the process.
Default The privileges of the process doing the SUBMIT.
Syntax Type **:privilege=** followed by **all** or by any of the following separated by commas: **setpriv**, **system**, **readphys**, **writephys**, **setprior**, **chngsuper**, **bypass**, **operator**, **altuic**, **world**, **group**, **setattr**, or **network**.

:qpriority=

Function Use this switch to specify the priority for the entry to be inserted.
Default The base priority of the queue as specified by the DSTAT :baseprior= switch.
Syntax Type **:qpriority=** followed by an integer between 0 and 15, with 0 as the highest priority and 15 as the lowest.

:queue=

Function Use this switch to specify the queue this function is to operate on.
Default SYS$PRINT.
Syntax Type **:queue=** followed by the devicename of a QUEUE class device whose type is PRINTER. No wildcard characters are allowed.

:siteid=	Function	Use this switch to specify the machine the process is to execute on.
	Default	The machine SUBMIT is being executed from.
	Syntax	Type **:siteid=** followed by a valid site ID.
:syserror=	Function	Use this switch to specify where standard error is supposed to be directed.
	Default	A file in the default directory with the name of PROGRAMNAME.LOG. For example, if a "SUBMIT DIR" is performed, SYS$ERROR is directed to a file called DIR.LOG in the default directory.
	Syntax	Type **:syserror=** followed by any valid file designation.
:sysinput=	Function	Use this switch to specify where standard input is supposed to be directed.
	Default	:sysinput=_NULL
	Syntax	Type **:sysinput=** followed by any valid file designation.
:sysoutput=	Function	Use this switch to specify where standard output should be directed.
	Default	:sysoutput=PROGRAMNAME.LOG
	Syntax	Type **:sysoutput=** followed by any valid file designation.
:timeslice=	Function	Use this switch to specify the timeslice the process is to have.
	Default	The timeslice of the process doing the SUBMIT.
	Syntax	Type **:timeslice=** followed by a positive integer.
:when=	Function	Use this switch to specify when this entry is to be entered into the queue and available for execution.
	Default	:when=+0
	Syntax	Type **:when=** followed by a date or time in the standard date and time syntax.

Examples

> submit dir

This command submits the DIR program into the queue SYS$BATCH for background execution. It uses standard CIP search path checking to find the image to execute. When the program is executed all output, log messages, and error messages are written to a file called DIR.LOG in the default directory. The name of the queue and the program's entry number appear on the screen:

```
Queue _BQ0
DIR.EXE queued as entry 44
```

> submit "dir sysexe/* :size :head" :inform

This command submits a DIR command that lists all files in SYSEXE and displays the size of each file. When DIR is finished executing, it sends a message to the screen that the process has finished. The program and its entry number are reported as shown:

```
Queue _BQ0
DIR.EXE queued as entry 89
```

> submit :list

This command lists all entries in the queue SYS$BATCH:

```
Queue _BQ0
Entry Qpri Username    Entry Name                      Status
----- ---- --------- ------------------- ------------------------
  114   7  USER      DIR.EXE             Active
  115   7  USER      COMPILE.EXE         Active
  117   8  USER      CHECKSUM.EXE        Pending
```

```
> submit :list :full
```

This performs the same function as the previous example, but with a full display:

```
Queue _BQ0

SYS$DISK/SYSEXE/DIR.EXE
Entry     : 114          Qpriority : 7            Username  : USER
Siteid    : 0003         Priority  : 3            Owner     : USER
Timeslice : 60           NoInform                 Entry name: DIR.EXE
Sysinput  : _null
Sysoutput : DIR.LOG
Syserror  : DIR.LOG
Directory : __NODEA_DS0/USER.TEMP/
Privilege : NOSETPRIV       NOSYSTEM        NOREADPHYS      NOWRITEPHYS
            NOSETPRIOR      NOCHNGSUPER     NOBYPASS        NOOPERATOR
            NOALTUIC        NOWORLD          GROUP
Command   : SYS$DISK/SYSEXE/DIR.EXE
Active

SYS$DISK/SYSEXE/COMPILE.EXE
Entry     : 115          Qpriority : 7            Username  : USER
Siteid    : 0003         Priority  : 3            Owner     : USER
Timeslice : 60           NoInform                 Entry name: COMPILE.EXE
Sysinput  : _null
Sysoutput : COMPILE.LOG
Syserror  : COMPILE.LOG
Directory : __NODEA_DS0/USER.TEST.WORK/
Privilege : NOSETPRIV       NOSYSTEM        NOREADPHYS      NOWRITEPHYS
            NOSETPRIOR      NOCHNGSUPER     NOBYPASS        NOOPERATOR
            NOALTUIC        NOWORLD          GROUP
Command   : SYS$DISK/SYSEXE/COMPILE.EXE TEST.C
Active

SYS$DISK/SYSEXE/CHECKSUM.EXE
Entry     : 117          Qpriority : 8         Username  : USER
Siteid    : 0003         Priority  : 3         Owner     : USER
Timeslice : 60           Inform                Entry name: CHECKSUM.EXE
Sysinput  : _null
Sysoutput : CHECKSUM.LOG
Syserror  : CHECKSUM.LOG
Directory : __NODEA_DS0/USER.WORK/
Privilege : NOSETPRIV       NOSYSTEM        NOREADPHYS      NOWRITEPHYS
            NOSETPRIOR      NOCHNGSUPER     NOBYPASS        NOOPERATOR
            NOALTUIC        NOWORLD          GROUP
Command   : SYS$DISK/SYSEXE/CHECKSUM.EXE .*/*.DAT
Pending
```

```
> submit :delete 1- :auto
```

This command deletes all entries in the queue automatically (without confirmation) and logs each entry as it is deleted:

```
     Queue _BQ0
       Entry 114 DIR.EXE Deleted.
       Entry 115 COMPILE.EXE Deleted.
       Entry 117 CHECKSUM.EXE Deleted.
```

```
> submit 203 :modify :qpriority=10 :priority=1 :when=tomorrow
```

This command modifies an entry that is already entered. It changes its queue priority, the process priority, and when it is to be executed. Note that any parameter or switch specified at the time the entry is queued can be modified. SUBMIT logs each entry as it is modified:

```
     Queue _BQ0
     Entry 203 DSTAT.EXE Modified.
```

```
> submit :close :queue=_BQ1
```

This command closes the queue _BQ1. Any user trying to add more entries to this queue receives this error message:

```
     Queue _BQ1 closed.
```

Note that if you list the entries of a queue after it has been closed or halted this kind of display appears:

```
Queue _bq0  Closed
Entry Qpri Username     Entry Name                   Status
----- ---- --------  ----------------  ------------------------------------
  114    7 USER      DIR.EXE           Pending
  115    7 USER      COMPILE.EXE       Pending
  117    8 USER      CHECKSUM.EXE      Pending
```

Using Prompts

```
> submit
Program      > dir
```

This performs the same function as the first example.

```
> submit :hold :queue=_bql
Entry list    > 25-27
```

The user is prompted for a list of entry numbers to be held from the specified queue. In this example, entries 25, 26 and 27 will not be executed.

Notes on Usage

Of the 11 actions that may be specified, only one can be executed at a time.

To SUBMIT a command file you must type a backslash (accept character), \, in front of the at sign, @. If you do not do this, SUBMIT treats the command file as if it were a parameter file.

The system manager can select how many batch processes can execute concurrently with the :NUMACTIVE= switch in DSTAT. When the queue is mounted the default is 1.

Note that when a batch queue is mounted, the :job switch must be specified on the MOUNT command line.

Related CIP Commands

print	Print text files and manage the print queue.

swap

Functional Description

Use this command to change the system disk.

Command Line Syntax

Mnemonic	swap
Optional parameter	Driver

Parameters

Driver	Function	Optional. Use this parameter to specify the driver for the system disk.
	Default	The driver found in the DEVCONFIG file for the system disk.
	Syntax	Type a standard file designation. Only the driver filename is required if it is found on the system disk in the directory SYSDSR/.

Switches

None.

Examples

> **swap**

This command replaces the system disk without having to shut down the system and mount another system disk.

SWAP reads the device driver for the system disk into memory, dismounts the system disk, prompts you to swap disks, and waits for your response. When you strike [RETRN], the disk you inserted into the old system disk's drive is mounted, and the label of the new system disk is displayed on the screen.

For example, suppose you are working on a dual-floppy system, where one of the floppies must be the system disk. You decide to execute SSCOPY, but /SYSEXE/SSCOPY.EXE is not on the system disk diskette. SYSEXE/ cannot be contained on a single floppy). To execute SSCOPY, you must replace the system diskette with a diskette that has a copy of SSCOPY on it. Follow these steps:

Step 1 Type **swap** on the CIP command line.

Step 2 Strike [RETRN].

The following appears on the screen:

```
Remove system disk.
Insert new System Disk
Then Press RETURN...
```

Step 3 Remove the former system diskette.

Step 4 Insert the new system diskette (into the drive of the former system disk).

Step 5 Strike [RETRN]. This kind of report appears:

```
Label="WMCS PART 2"
```

This report tells you WMCS PART 2 is the volume that is now the system disk.

Using Prompts

None.

Notes on Usage

SWAP has no way of verifying that the file designation you type for the second parameter is a device driver or that it is even the correct device driver) for the specified disk. Thus, if you assign the wrong device driver, you must reboot your system.

Related CIP Commands

mnt	Mount a device
dmnt	Dismount (remove) a device

swapper

Functional Description

Use this command to allow more processes to run in physical memory by swapping inactive processes to the disk.

Command Line Syntax

Mnemonic	swapper
Optional parameter	Filename
Switches	:filesize=

Parameters

Filename	Function	Optional. Use this parameter to specify the name of the file which will be used as the swapfile. The file may reside on any accessible disk device, not necessarily on the current machine. If the filename used as the swapfile does not already exist, SWAPPER creates the file with protections such that no process can access it without bypass privileges. This is done to prevent accidental tampering as well as to prevent the BACKUP utility from attempting to back up the swapfile.
	Default	sys$disk/rootdir/swapfile.sys
	Syntax	Type any valid disk device filename.

Switches

:filesize=

:filesize=	Function	Use this switch to specify the size of the swapfile in Kbytes. If the swapfile does not exist, SWAPPER creates it to be the specified size. If the swapfile already exists, it will be lengthened or truncated as specified by this switch. If the value specified is larger than the amount of available disk space, an error will be reported and SWAPPER will quit.
	Default	:filesize=4096 (if SWAPPER creates it; if the swapfile already exists, the default is the current file size)
	Syntax	Type **:filesize=** followed by a number between 100 and 16384 inclusive.

Examples

> **&swapper**

This command starts up SWAPPER and uses the file sys$disk/rootdir/ swapfile.sys as its swapfile. If the file does not already exist, it will be created and contain 4096 Kbytes. The file size will not be altered if the file already exists.

> **&swapper** __software_dsl/rootdir/swapper.dat :filesize=1000

This command starts up SWAPPER and uses the file __software_dsl/rootdir/ swapper.dat for its swapfile. If the file does not already exist, it will be created and contain 1000 Kbytes. If the file already exists, the size of the file will be altered to occupy 1000 Kbytes of disk space.

Using Prompts

None.

Notes on Usage

SWAPPER allows more processes to run in physical memory because it "swaps" inactive processes to the disk. However, it does not allow any single process to use more memory than actually exists on the system. The sum of memory required by all processes and the WMCS cannot exceed the sum of physical memory available plus the size of the swapfile.

SWAPPER may be started or killed at any time. When SWAPPER is running and memory requests exceed physically available space, sufficient inactive memory is moved to disk in an effort to fill the request. The swapper may be killed at any time by a PSTAT utility with adequate privilege. SWAPPER has an exit handler which will trap the exit process call. It will continue running until all memory in the swapfile can be transferred back to physical memory. After SWAPPER is killed, a "No memory available" error will be returned to memory requests should sufficient physical memory be unavailable.

The status of swapped processes can be monitored by use of the PSTAT utility. The size of a process reflects the amount of physical memory currently occupied by the process. As a process moves to and from the swapfile, the process size will fluctuate. The DM utility can also be used to monitor the status of process, system, or swapfile memory.

SWAPPER requires the SETPRIV privilege to run. It then acquires all other privileges as it starts up. Although SWAPPER can be spawned, it makes no sense to do so. When SWAPPER determines that it can successfully run, it sets its process ID to 4, its name to "swapper", its priority to 2, its timeslice to 100, and its UIC to SYSTEM.

Related CIP Commands

cip	Command Interpreter Program
dm	Display memory status
pstat	Display process status
userprof	Create or alter user profile

sysprof

Functional Description

Use this command to set up the configuration of a system. SYSPROF allows the system manager to edit all fields in the SYSCONFIG.nnn and DEVCONFIG.nnn files. If these files do not exist when SYSPROF is started up, they are created.

Command Line Syntax

Mnemonic	sysprof		
Optional parameter	Devicename		
Switches	:model=	:setupin=	:setupout=

Parameters

Devicename	Function	Optional. Use this parameter to specify the destination device where the SYSCONFIG.nnn and DEVCONFIG.nnn files are read and written.
	Default	sys$disk
	Syntax	Type a standard device designation. Wildcard symbols are not allowed.

NOTE: In all examples, "nnn" is the system model number.

Switches

:model=

Function — Use this switch to specify which class of computer hardware the SYSCONFIG.nnn and DEVCONFIG.nnn files are for.

Default — The value of the logical name SYS$MODEL on the computer to which the specified device is attached.

Syntax — Type **:model=** followed by one of the following: **100, 150, 155, 156, 160, 200, 220, 2220,** or **300.**

NOTE: WICAT system models 100, 200, 220, 2220, and 300 are converted to system model number 100. WICAT models 150, 155, 156 and 160 are converted to system model number 156.

The system model numbers correspond to the type of interconnection bus the computer hardware uses. The proprietary WICAT bus is used in the WICAT models 100, 200, 220, 2220, and 300. The system model number 100 refers to the proprietary WICAT bus. The industry-standard MULTIBUS (trademark of Intel Corp.) bus is used in the WICAT models 150, 150-6 (156), 155 and 160. The system model number 156 refers to the MULTIBUS bus. The system model numbers are used as the file extension name for the system configuration file, SYSCONFIG.nnn, and the device configuration file, DEVCONFIG.nnn, where "nnn" is the system model number.

:setupin=

Function — Use this switch to specify an alternate setup file to use for sys$input. If this name specifies a directory path, SYSPROF looks in that directory for the setup file based on the terminal type. If this specifies an explicit file, it uses that file.

Default — Use the setup file in sys$disk/syslib.setup/ determined by the terminal type, e.g., if your terminal is a T7000, then the file used is sys$disk/syslib.setup/setup252.sys.

Syntax — Type **:setupin=** followed by a filename.

:setupout=

Function — Use this switch to specify an alternate setup file to use for sys$output. If you specify a directory path, SYSPROF looks in that directory for the setup file based on the terminal type. If you specify an explicit file, it uses that file.

Default — Use the setup file in sys$disk/syslib.setup/ determined by the terminal type, e.g., if your terminal is a T7000, then the file used is sys$disk/syslib.setup/setup252.sys.

Syntax — Type **:setupout=** followed by a filename.

Examples

> **sysprof**

This command generates the following kind of display:

```
                          System Configuration
--------------------------------------------------------------------------------
System Model     :                          Boot Device Driver :
Operating System :                          Current Year       :

Class Handlers :  TTY           DISK          TAPE          KSAM
                  MATH1010      QUEUE         FPOINT        NETWORK


                          Device Configuration
--------------------------------------------------------------------------------
Device Name  :            Device Driver :
Drive ID     :            Drive Type    :
Device Class :

Default User     :
Authorized Users :
Exclude Users    :

Dstat :
```

Seven messages appear, one after another, on the bottom line of the terminal screen (after the foregoing display):

```
Locating devicename/ROOTDIR/SYSCONFIG.nnn.1
Locating devicename/ROOTDIR/DEVCONFIG.nnn.1
Locating boot device drivers...
Locating other device drivers...
Locating disk drive types...
Reading devicename/ROOTDIR/SYSCONFIG.nnn.1
Reading devicename/ROOTDIR/DEVCONFIG.nnn.1
```

The first two messages tell you that SYSPROF is locating the SYSCONFIG and DEVCONFIG files. If SYSPROF cannot find a file, the message is replaced with another message telling you that the file is being created.

The third message tells you that SYSPROF is locating all of the boot device driver files which WICAT Systems releases with the operating system. A boot device driver is simply a disk device driver.

The fourth message tells you that SYSPROF is locating all of the other (non-disk) device driver files.

The fifth message tells you that SYSPROF is reading the DEVICENAME/SYSDSR/DISK.CFG file in order to locate the disk drive types.

The sixth and seventh messages tell you that SYSPROF is now reading the SYSCONFIG and DEVCONFIG files. If invalid data is detected in the SYSCONFIG file, it is re-initialized with default data and a message informs you what has been done.

After SYSPROF has located and read all the data from the two files, the information from the SYSCONFIG file is placed in the fields in the upper half of the SYSPROF display, and the information from the first record in the DEVCONFIG file is placed in the fields in the lower half of the display.

Using Prompts

None.

Notes on Usage

SYSPROF should only be used by the system manager.

SYSPROF allows the system manager to set up a system's configuration by editing all fields in the /ROOTDIR/SYSCONFIG.nnn.1 and /ROOTDIR/DEVCONFIG.nnn.1 files. Note that only version 1 of these files is used. Any other version is not recognized. SYSPROF creates, reads and writes only to version 1. The files must exist in the /ROOTDIR/directory because SYSPROF does not look in any other directory.

The SYSCONFIG file contains the operating system modules to be loaded, the system boot device driver, and the year of the date. When a system is booted, this file is used to determine the configuration of the system.

The DEVCONFIG file contains this information about each device on the system:

devicename	name of the device
drive id	drive identification information
device driver	device driver file name
drive type	drive type (disk devices only)
:defaultuser=	default user (tty devices only)
:dstat=	initial device status (tty devices only)
:authorized=	authorized users (tty devices only)
:exclude=	non-authorized users (tty devices only)

Detailed information on configuring your system is presented under these headings:

The SYSPROF Display
Functions Available in SYSPROF
Field Characteristics
Cursor Movement and Control-Key Functions

The SYSPROF Display

The following is an example of the SYSPROF display with data in the fields:

```
                              System Configuration
-----------------------------------------------------------------------------
System Model     : 100                            Boot Device Driver : SMD$100
Operating System : KERNELBUG                      Current Year       : 1985

Class Handlers :    TTY              DISK           TAPE           KSAM
                 NOMATH1010          QUEUE          FPOINT         NETWORK


                              Device Configuration
-----------------------------------------------------------------------------
Device Name  : TT0        Device Driver : TTY$100
Drive ID     : 0A0        Drive Type    :
Device Class : TTY

Default User     :
Authorized Users : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users    :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

The SYSPROF display is divided into two halves; an upper half which contains the fields of the SYSCONFIG file, and a lower half which contains the fields of a single DEVCONFIG record. The SYSCONFIG information is displayed at all times and can be modified at any time by the system manager. Only one record of the DEVCONFIG file is displayed at a time.

The following is a description of each field in the SYSPROF display:

```
                              System Configuration
-----------------------------------------------------------------------------------
System Model     : 100                              Boot Device Driver : SMD$100
Operating System : KERNELBUG                        Current Year       : 1985

Class Handlers :    TTY             DISK            TAPE            KSAM
                 NOMATH1010         QUEUE           FPOINT          NETWORK


                              Device Configuration
-----------------------------------------------------------------------------------
Device Name  : TT0          Device Driver : TTY$100
Drive ID     : 0A0          Drive Type    :
Device Class : TTY

Default User      :
Authorized Users : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users     :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

System Model and Boot Device Driver Fields

The System Model field is a display-only field that cannot be edited. It displays the system model number, i.e., 100 or 156, and is for your information only. The WICAT model number information is specified by the :model= switch on the CIP command line.

The Boot Device Driver field contains the name of the boot device driver file. When a system is booted, this device driver is loaded by the boot program after the class handlers have been loaded. The possible values depend upon the system model number. They are:

Model 100	Model 156
FLNT$100	SMD$156
IMI$100	WD2$156
SMD$100	WD3$156
	WDMF$156

These values are selected by positioning the cursor at this field and pressing the spacebar. The selections are presented in alphabetical order. The first selection is shown after the last selection has been shown. Any selection may be overridden by typing name of a boot device driver. If you specify a driver different from those in the tables above, it is added to the selection list.

```
                              System Configuration
--------------------------------------------------------------------------------
System Model      : 100                        Boot Device Driver : SMD$100
Operating System : KERNELBUG                   Current Year       : 1985

Class Handlers :    TTY            DISK            TAPE            KSAM
                  NOMATH1010       QUEUE           FPOINT          NETWORK


                              Device Configuration
--------------------------------------------------------------------------------
Device Name   : TT0        Device Driver : TTY$100
Drive ID      : 0A0        Drive Type    :
Device Class : TTY

Default User       :
Authorized Users : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users      :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

Operating System and Current Year Fields

The Operating System field specifies which kernel, or core, of the WMCS should be loaded when the system is booted. The two selections are KERNEL and KERNELBUG. Pressing the spacebar selects the one not displayed. KERNEL is the normal WMCS operating system. KERNELBUG has built-in debugging facilities that experienced systems programmers can use in low-level development work. Choose KERNEL for normal system operation.

The Current Year field specifies the year for the time-of-day clock. The clock does not keep track of the year, so it must be loaded in by the WMCS at boot time. The range of acceptable years is 1985 through 2025. The year may be selected by pressing the spacebar and stopping on the desired year, or by typing it in.

```
                              System Configuration
-------------------------------------------------------------------------------
System Model     : 100                         Boot Device Driver : SMD$100
Operating System : KERNELBUG                   Current Year       : 1985

Class Handlers :    TTY             DISK            TAPE            KSAM
                  NOMATH1010        QUEUE           FPOINT          NETWORK


                              Device Configuration
-------------------------------------------------------------------------------
Device Name  : TT0          Device Driver : TTY$100
Drive ID     : 0A0          Drive Type    :
Device Class : TTY

Default User     :
Authorized Users : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users    :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

Class Handlers Field

This field specifies the class handlers which can be loaded at boot time and which configure the WMCS. All class handlers are selected by pressing the spacebar. A "NO" appearing in front of a class handler name means the class handler will not be loaded. Nothing in front of the name means the class handler will be loaded.

TTY The TTY class handler supports the terminals and serial devices on your system and should almost always be loaded.

DISK The DISK class handler supports the disk devices on your system and should almost always be loaded.

TAPE The TAPE class handler supports the tape devices on your system and should be loaded only if your system has a tape drive.

KSAM The KSAM class handler supports the Keyed Sequential Access Method of file access. It should be loaded if any program uses this file access method.

MATH1010 The MATH1010 class handler supports the old floating point math routines used by some old Pascal programs. This class handler should be loaded only if any programs are using the old floating point. The WMCS and its utilities do not require this class handler.

QUEUE The QUEUE class handler supports the print spooler(s) and batch jobs. Load this class handler if print spooling or batch jobs are used on your system.

FPOINT The FPOINT class handler supports the new floating-point math routines and hardware. This class handler should be loaded only if any programs on your system make use of the new floating point libraries.

NETWORK The NETWORK class handler supports virtual circuits and networking. This class handler should be loaded if your system has networking hardware (Ethernet, Arcnet, etc.).

```
                          System Configuration
--------------------------------------------------------------------------
System Model     : 100                        Boot Device Driver : SMD$100
Operating System : KERNELBUG                  Current Year       : 1985

Class Handlers :   TTY          DISK          TAPE          KSAM
                 NOMATH1010     QUEUE         FPOINT        NETWORK
```

```
                          Device Configuration
--------------------------------------------------------------------------
Device Name  : TT0         Device Driver : TTY$100
Drive ID     : 0A0         Drive Type    :
Device Class : TTY

Default User     :
Authorized Users : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users    :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

Device Name and Device Driver Fields

The Device Name field is a display-only field and cannot be edited. This is the name you enter when adding a device record or refer to this device. Any alphanumeric string can be used to identify the device. The recommended devicenames are:

5.25-in. floppies	_dx0, _dx1, _dx2, ...
8-in. floppies	_df0, _df1, _df2, ...
5.25-in. Winchesters	_dc0, _dc1, _dc2, ...
SMD disk drives	_ds0, _ds1, _ds2, ...
1/2-in. magnetic tape	_mt0, _mt1, _mt2, ...
1/4-in. cartridge tape	_ct0, _ct1, _ct2, ...
Parallel ports	_pp0, _pp1, _pp2, ...
Serial port	_tt0, _tt1, _tt2, ...
Memory Disk	_md0
Batch Queues	_bq0, _bq1, _bq2, ...
Print Queues	_pq0, _pq1, _pq2, ...
Pipes	Unspecified

The Device Driver field contains the name of the device driver file. When the device is mounted, this device driver is loaded by the WMCS if it has not already been loaded by a previous mount. The possible values depend upon the system model number. They are:

Model 100	Model 156
ADEI$100	ADEI$156
AUDIO$100	CENT$156
CIPH$100	CIPH$156
ENET$100	ENET$156
FLNT$100	MDSK$156
HYDRA$100	PIPE$156
IMI$100	QUE$156
MDSK$100	SMD$156
PIPE$100	TTY$156
QUE$100	VC$156
SMD$100	WD3$156
STET$100	WD2$156
TTY$100	WDMF$156
VC$100	
WD3$100	

These values are selected by positioning the cursor at this field and pressing the spacebar. The selections are presented in alphabetical order. The first selection is shown after the last selection has been shown. Any selection may be overridden by typing the name of a device driver. If you specify a driver different from those in the tables above, it is added to the selection list.

```
                           System Configuration
--------------------------------------------------------------------------
System Model       : 100                       Boot Device Driver : SMD$100
Operating System   : KERNELBUG                 Current Year       : 1985

Class Handlers :    TTY              DISK          TAPE            KSAM
                 NOMATH1010          QUEUE         FPOINT          NETWORK


                           Device Configuration
--------------------------------------------------------------------------
Device Name  : TT0          Device Driver : TTY$100
Drive ID     : 0A0          Drive Type    :
Device Class : TTY

Default User     :
Authorized Users : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users    :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

Drive ID and Drive Type Fields

The Drive ID field identifies the actual hardware. It has three parts: drive number, device type, and controller board number. The device number is represented by a numeral. For example, if the device is a hard disk, 0 represents the first hard disk on the system, 1 represents the second hard disk, etc. The device type has different meanings depending on the controller hardware. For example, A indicates a winchester disk and B a floppy disk for a WD3 controller. For the IPE controller, A indicates a serial port and B a parallel port. The controller board number represents the number of the controller board to which the device is connected. The first board is numbered zero.

The Drive Type field allows you to specify the type of drive to be used. This only applies to disk class devices or to devices whose class is unknown. The drive types which appear in the selection list are:

FLOP015	1.5 Mb unformatted 8-in. floppy
FLOP09A	0.9 Mb unformatted 4 sector 5.25 in. floppy
FLOP09B	0.9 Mb unformatted 5 sector 5.25 in. floppy
IMI20	20 Mb unformatted IMI disk
IMI40	40 Mb unformatted IMI disk
SMD168A	168 Mb (512 byte sector) unformatted SMD disk
SMD168B	168 Mb (1024 byte sector) unformatted SMD disk
SMD474A	474 Mb (512 byte sector) unformatted SMD disk
SMD474B	474 Mb (1024 byte sector) unformatted SMD disk
SMD84A	84 Mb (512 byte sector) unformatted SMD disk
SMD84B	84 Mb (1024 byte sector) unformatted SMD disk
WIN12	12 Mb unformatted Winchester disk
WIN19	19 Mb unformatted Winchester disk (fast version)
WIN19S	19 Mb unformatted Winchester disk (slow version)
WIN30	30 Mb unformatted Winchester disk
WIN43	43 Mb unformatted Winchester disk

These values are selected by positioning the cursor to this field and pressing the spacebar. The selections are presented in alphabetical order. The first selection is shown after the last selection has been shown. Any selection may be overridden by typing the name of a device type. If you specify a type different from those in the table above, it is added to the selection list.

```
                              System Configuration
--------------------------------------------------------------------------------
System Model      : 100                         Boot Device Driver : SMD$100
Operating System  : KERNELBUG                   Current Year       : 1985

Class Handlers :   TTY            DISK            TAPE            KSAM
                 NOMATH1010       QUEUE           FPOINT          NETWORK


                              Device Configuration
--------------------------------------------------------------------------------
Device Name  : TT0        Device Driver : TTY$100
Drive ID     : 0A0        Drive Type    :
Device Class : TTY

Default User      :
Authorized Users  : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users     :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

Device Class Field

This field is a display-only field and cannot be edited. It contains the class of the device. The device class is based upon which device driver is selected. If the name of a device driver is given and the device driver file is not in the /sysdsr/ directory of the device upon which the SYSCONFIG and DEVCONFIG files reside, the device class is specified as UNKNOWN.

The next four fields in the SYSPROF display apply only to TTY class devices. LOGON reads these fields to determine who is allowed to log on and what the environment should be after logging on.

```
                              System Configuration
------------------------------------------------------------------------------
System Model      : 100                         Boot Device Driver : SMD$100
Operating System  : KERNELBUG                   Current Year       : 1985

Class Handlers :   TTY          DISK           TAPE          KSAM
                 NOMATH1010     QUEUE          FPOINT        NETWORK


                              Device Configuration
------------------------------------------------------------------------------
Device Name  : TT0        Device Driver : TTY$100
Drive ID     : 0A0        Drive Type    :
Device Class : TTY

Default User     :
Authorized Users : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users    :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

Default User Field

Use this field to specify a valid username. If a username is specified, any person may log on to the system simply by pressing the return key of a terminal attached to the TTY port specified in the devicename field. No username or password is required. The person will have the username specified by this field, along with the default UIC, directory, etc. of the username. If no username or an invalid username (not in SYS$DISK/SYSLIB/UAF.DAT) is specified, the normal log-on procedure, i.e., prompting for username and password, is performed.

```
                           System Configuration
--------------------------------------------------------------------------
System Model     : 100                        Boot Device Driver : SMD$100
Operating System : KERNELBUG                  Current Year       : 1985

Class Handlers :   TTY           DISK           TAPE           KSAM
                 NOMATH1010      QUEUE          FPOINT         NETWORK


                           Device Configuration
--------------------------------------------------------------------------
Device Name  : TT0        Device Driver : TTY$100
Drive ID     : 0A0        Drive Type    :
Device Class : TTY

Default User     :
Authorized Users : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users    :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

Authorized Users Field

Use this field to specify a list of usernames, separated by commas. The usernames specified in this field are the only usernames allowed to log on through the TTY port specified in the devicename field. If no usernames are specified, any username may log on to the port. If only invalid usernames are specified, no one can log on to the port.

```
                          System Configuration
--------------------------------------------------------------------------
System Model      : 100                       Boot Device Driver : SMD$100
Operating System  : KERNELBUG                 Current Year       : 1985

Class Handlers :    TTY          DISK         TAPE          KSAM
                 NOMATH1010      QUEUE        FPOINT        NETWORK


                          Device Configuration
--------------------------------------------------------------------------
Device Name  : TT0         Device Driver : TTY$100
Drive ID     : 0A0         Drive Type    :
Device Class : TTY

Default User      :
Authorized Users  : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users     :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

__Exclude Users Field__

Use this field to specify a list of usernames, separated by commas. The usernames specified in this field are the usernames __not__ allowed to log on through the TTY port specified in the devicename field. If only invalid usernames are specified, all users can log on to the port.

```
                          System Configuration
---------------------------------------------------------------------------
System Model     : 100                       Boot Device Driver : SMD$100
Operating System : KERNELBUG                 Current Year       : 1985

Class Handlers :   TTY          DISK            TAPE            KSAM
                 NOMATH1010     QUEUE           FPOINT          NETWORK


                          Device Configuration
---------------------------------------------------------------------------
Device Name  : TT0        Device Driver : TTY$100
Drive ID     : 0A0        Drive Type    :
Device Class : TTY

Default User     :
Authorized Users : DICK,HARRY,BOBBY,TOM,SUSAN,VALERIE
Exclude Users    :

Dstat : :termtype=t7000 :controldefault :baudrate=9600 :hostsync=software
```

Dstat Field

Use this field to specify a command line to the DSTAT utility when a user logs on. The system manager sets up certain characteristics for a TTY port when the system is booted. However, a user may set up the port to be in a different state and then log off. A value in this field guarantees the terminal will come up in a known state when the next user logs on. The value of this field should be the same as a command line given to the DSTAT utility.

Functions available in SYSPROF

Each of the following functions is executed by first striking [ESC][ESC], then typing the mnemonic on the SYSPROF function line that appears at the bottom of the screen. The position of the cursor when you strike [ESC][ESC] does not affect the execution of any of these functions.

> NOTE: These functions only apply to device records unless noted otherwise.

ab

Abort this session of SYSPROF.

FUNCTIONAL DESCRIPTION

AB aborts all changes made during an editing session. The SYSCONFIG and DEVCONFIG files are the same as before the editing session began.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

ex	Exit SYSPROF
sa	Save modifications made so far

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type ab on the SYSPROF function line.

If the SYSCONFIG and DEVCONFIG information has not been modified, the screen in cleared, and you return to your CIP prompt. If modifications have been made, this prompt appears on the function line:

```
SYSCONFIG and/or DEVCONFIG files have been modified - Abort?
```

Step 3 Type y or n. If you type some other character, the following message briefly appears on your screen:

```
Please answer 'Y' or 'N'.
```

If you type Y, the screen is cleared and you return to your CIP prompt. If you type N, the cursor returns to the position it had on the screen before executing AB.

ad

Add a new device record.

FUNCTIONAL DESCRIPTION

AD creates a new device record, assigns it the devicename you specify, and adds it to the list of device records.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

dl	Delete a device record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type ad on the SYSPROF function line. This prompt appears on the function line:

```
Add Device Name>
```

Step 3 Type the devicename to be assigned to the device. The devicename must be unique to the DEVCONFIG file and cannot be longer than nine characters (including the underscore). The underscore preceding the devicename is optional.

Step 4 Strike [RETRN]. The new device record appears in the bottom half of the SYSPROF display. All other fields in the record are empty.

If no devicename is specified, the following message is displayed on the function line:

```
Please enter a valid devicename.
```

cp

Spawn a CIP while still in SYSPROF.

FUNCTIONAL DESCRIPTION

CP allows you to execute a CIP while staying in SYSPROF.

CORRESPONDING CONTROL-KEY FUNCTIONS

[ESC] c

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **cp** on the SYSPROF function line. Your CIP prompt appears.

When you log off from the CIP, you are returned to SYSPROF.

dl

Delete a device record.

FUNCTIONAL DESCRIPTION

DL deletes the device record displayed in the bottom half of the screen.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

ad	Add a new device record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type dl on the SYSPROF function line. This prompt appears on the function line:

```
Delete record ?
```

Step 3 Type y if you want to delete the current record or n if you do not want to delete it. If characters other than Y or N are typed, the following message is displayed on the function line:

```
Please type 'Y' or 'N'.
```

Exit SYSPROF

FUNCTIONAL DESCRIPTION

EX terminates the editing session, saves any changes made during the editing session, and returns you to the CIP.

CORRESPONDING CONTROL-KEY FUNCTIONS

{-}

RELATED ESCAPE-KEY FUNCTIONS

ab	Abort SYSPROF
sa	Save modifications made so far

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **ex** on the SYSPROF function line. The following message appears at the bottom of the screen:

```
Exiting...
```

The screen is cleared and the cursor reappears next to your CIP prompt.

he

Call to the screen a display of SYSPROF functions and features.

FUNCTIONAL DESCRIPTION

HE displays a list of functions available in SYSPROF.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **he** on the SYSPROF function line. The first page (or screenful) of a three-page help display appears on your screen.

The first page contains a list of the escape-key functions available.

The second page lists functions for deletion and cursor movement.

The third page describes the different field types.

Step 3 Strike [RETRN] to cycle through the pages and return to the SYSPROF display.

Type [CTRL] c at any time to return to the SYSPROF display without showing the remaining help pages.

The complete help display looks like this:

```
                 SYSPROF HELP DISPLAY - Major Commands              1 of 3
--------------------------------------------------------------------------

AB - Abort this program and do not save any modifications
AD - Add a device configuration record
CP - Spawn a CIP
DL - Delete the displayed device configuration record
EX - Exit this program and save all modifications
HE - Show this HELP display
MP - Page to the next (MP) or previous (-MP) device configuration record
PR - Create a print file of configuration records ((SYS/DEV)CONFIG.PRN)
RS - Redisplay the screen
SA - Save all modifications
SF - Show the first device configuration record
SH - Show a device configuration record
SL - Show the last device configuration record
SR - Search for a device configuration record
UD - Undo any changes made to the device configuration record

                                          Press any key to continue . . .
```

```
           SYSPROF HELP DISPLAY - Deletion and Cursor Movement      2 of 3
--------------------------------------------------------------------------

DB - Delete from Beginning of field to cursor
DC - Delete Character under cursor (DC) or character before cursor (-DC)
DE - Delete from cursor to End of field
DF - Delete entire Field
DW - Delete to beginning of next (DW) or end of previous (-DW) word

EL - move to End of last field on current Line
FL - move to Front of first field on current Line
MC - Move to the next (MC) or previous (-MC) Character
ME - Move to the End of the form (end of last field)
MF - Move to the beginning of next (MF) or end of previous (-MF) Field
ML - Move to the beginning of the next (ML) or previous (-ML) Line
MP - Page to the next (MP) or previous (-MP) device configuration record
MT - Move to the Top of the form (beginning of first field)
MW - Move to beginning of next (MW) or end of previous (-MW) Word
PL - Position up (-PL) or down (PL) a line

Each command here can be performed with a single keystroke.  The keys which
perform the commands vary according to the setup file for your terminal.

                                          Press any key to continue . . .
```

```
          SYSPROF HELP DISPLAY - Field Types and Characteristics        3 of 3
______________________________________________________________________________
The field types in the SYSPROF form are: Toggle, Decimal, String and Selection.

Toggle:     Two values only.  Example: DESENCRYPT/NODESENCRYPT.  Pressing the
            spacebar changes the current value to the other value.

Decimal:    Variable number of digits.  Example: 1985.  There is never a
            leading zero.  Deletion type editing shifts digits to the left.
            Insertion type editing shifts digits to the right.  Rightmost digit
            is not lost unless the entire decimal field is full of digits.
            Pressing the spacebar increments the value in the field.

String:     Variable number of characters.  Deletion type editing shifts
            characters to the left.  Insertion type editing shifts characters
            to the right.  Any printable character may be entered.  If the
            string is larger than the field, horizontal scrolling is
            performed.

Selection: Exactly like string/decimal except embedded spaces are not allowed.
            Pressing the spacebar selects the next item in a list.  Typing any
            printable character (except a space) overrides the item in the field.

                                    Press any key to continue . . .
```

mp

Move to the next (or the previous) device record in the DEVCONFIG file.

FUNCTIONAL DESCRIPTION

MP calls to the screen the device record immediately following the device record currently on the screen.

This message appears on the bottom line of the screen when the last device record in DEVCONFIG.nnn is already on the screen:

```
No next record ...
```

-MP calls to the screen the device record immediately preceding the device record currently on the screen. If you are on the first device record in the DEVCONFIG.nnn file and try paging backward, this message appears:

```
No prior record ...
```

To move forward more than one record at a time, type a number (representing the number of pages to be moved) before the MP. To move backward, type a negative number before the MP.

CORRESPONDING CONTROL-KEY FUNCTIONS

[CTRL] p	(for MP)
[CTRL] h	(for -MP)

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 | Strike [ESC][ESC]

Step 2 | Type mp on the SYSPROF function line. This report appears momentarily on the screen:

```
Paging to next record
```

The next record appears on your screen.

pr

Create a text-file copy of SYSPROF records.

FUNCTIONAL DESCRIPTION

PR creates two files, SYSCONFIG.PRN and DEVCONFIG.PRN, in the directory in which the SYSCONFIG.nnn and DEVCONFIG.nnn records are located.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **pr** on the SYSPROF function line. This report appears momentarily on the screen:

```
Printing devicename/ROOTDIR/SYSCONFIG.nnn.1 records...
```

During the display of this message, the SYSCONFIG.nnn record is written to the file SYSCONFIG.PRN. This report then appears on the screen:

```
Printing devicename/ROOTDIR/DEVCONFIG.nnn.1 records...
```

During the display of this message, the DEVCONFIG.nnn records are written to the file DEVCONFIG.PRN. After the records are written, the message is erased from the screen and the cursor is returned to its original position on the screen.

rp

Repeat the previous command.

FUNCTIONAL DESCRIPTION

Use this function to repeat the last function or action you performed.

CORRESPONDING CONTROL-KEY FUNCTIONS

[CTRL] z   Repeat previous command

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1   Strike [ESC][ESC]. The number one appears at the bottom of your screen on the VEW function line.

Step 2   If you want to repeat the function once, skip this step. Otherwise, type a number representing the number of times you want the last function repeated.

NOTE: The repetition number only repeats the last action the specified number of times if that action was a function, not just a keystroke.

Step 3   Type **rp**. The last function or macro you executed or the last keystroke you typed is repeated.

If no previous command exists, nothing happens.

rs

Refresh the screen.

FUNCTIONAL DESCRIPTION

RS restores the display on the screen. This is useful when a message or other interruption disrupts the display.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1	Strike [ESC][ESC]
Step 2	Type **rs** on the SYSPROF function line. The screen is restored to its original display.

sa

Save the modifications made so far in the DEVCONFIG file.

FUNCTIONAL DESCRIPTION

SA writes out all modified device records to the DEVCONFIG file without exiting SYSPROF. This is useful when you have a large number of modifications and want to protect them from system failure.

CORRESPONDING CONTROL-KEY FUNCTIONS

{,}

RELATED ESCAPE-KEY FUNCTIONS

ex	Exit SYSPROF
ab	Abort the editing session

EXECUTION

Step 1 | Strike [ESC][ESC]

Step 2 | Type **sa** on the SYSPROF function line. This prompt appears on the function line:

```
Saving modifications...
```

After writing all the modified records to the DEVCONFIG file, the cursor returns to the position it had prior to the SA command.

sf

Show the first device record.

FUNCTIONAL DESCRIPTION

SF allows you to show the first device record in the DEVCONFIG.nnn file.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

sh	Show a specific device record
sl	Show the last device record
sr	Search for a device record

Execution

Step 1 Strike [ESC][ESC]

Step 2 Type **sf** on the SYSPROF function line. This message appears momentarily on the SYSPROF function line:

```
Showing the first record . . .
```

The first device record is displayed on the screen.

sh

Show a specific device record.

FUNCTIONAL DESCRIPTION

SH allows you to specify which device record you want displayed on the screen.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

Related ESCAPE-KEY Functions

sf	Show the first device record
sl	Show the last device record
sr	Search for a device record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type sh on the SYSPROF function line. This prompt appears on the SYSPROF function line:

```
Show Device Name>
```

Step 3 Type the name of a device. (The underscore preceding the devicename is optional.

Step 4 Strike [RETRN]. If the specified devicename has a record, that device record is displayed on the screen. Otherwise, this message appears on the screen:

This devicename does not exist. Please enter another name.

sl

Show the last device record.

FUNCTIONAL DESCRIPTION

SL allows you to show the last device record in the DEVCONFIG.nnn file.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

Related ESCAPE-KEY Functions

sf	Show the first device record
sh	Show a specific device record
sr	Search for a device record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type sl on the SYSPROF function line. This message appears momentarily on the function line:

```
Showing the last record . . .
```

The last device record is displayed on the screen.

sr

Search for and display a device record.

FUNCTIONAL DESCRIPTION

SR allows you to give a name or partial name of a device and display a device record on the screen. If the given devicename does not have a record, the record of the device whose name alphabetically follows the given devicename is displayed.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

sf	Show the first device record
sh	Show a specific device record
sl	Show the last device record

EXECUTION

Step 1 Strike [ESC][ESC]

Step 2 Type **sr** on the SYSPROF function line. This prompt appears on the function line:

```
Show Device Name>
```

Step 3 Type the name or partial name of a device. (The underscore preceding the devicename is optional.)

For example, type **tt** to find the first device whose name starts with "tt".

Step 4 Strike [RETRN]. A device record is displayed on the screen.

ud

Undo changes made on a record.

FUNCTIONAL DESCRIPTION

UD allows you to undo changes made to a record. If the cursor is in a field in the upper half of the form, i.e., in the SYSCONFIG record, any changes made to the SYSCONFIG record are undone. If the cursor is in a field in the lower half of the form, i.e., in a DEVCONFIG record, any changes made to the DEVCONFIG record are undone. Changes made to the SYSCONFIG record can be undone at any time. Changes made to a DEVCONFIG record can be undone only while the record is displayed on the screen. Any function on the first help display page -- except CP, HE, and RS -- makes the changes permanent (unless SYSPROF is aborted with the AB function).

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 | Strike [ESC][ESC]

Step 2 | Type **ud** on the SYSPROF function line. This message appears momentarily on the function line:

```
Undoing changes...
```

The device record is displayed on the screen as it was before the changes were made.

Field Characteristics

The SYSPROF display is a form-oriented display. This means that the screen contains fields in which you may type information and the rest of the screen you cannot affect. The SYSPROF utility has four different kinds of fields. The cursor movement and control-key functions operate differently in each type of field. The four field types are: toggle, decimal, string and selection. The SYSPROF display fields and their types are:

Toggle	Decimal	String	Selection
Operating System	Current Year	Boot Device Driver	Boot Device Driver
TTY		Device Driver	Current Year
DISK		Drive ID	Drive Type
TAPE		Drive Type	
KSAM		Default User	
MATH1010		Authorized Users	
QUEUE		Exclude Users	
FPOINT		Dstat	
NETWORK			

Toggle fields have only two possible values. The values are toggled from one to the other by pressing the spacebar when the cursor is in the field.

Decimal fields contain only decimal digits. There is never a leading zero. Digits may be deleted and inserted with the normal editing commands.

String fields can contain any printable ASCII characters. Characters may be deleted and inserted with the normal editing commands. If the file record data is larger than the string field can display, the data automatically scrolls horizontally within the field boundaries.

Selection fields are like Decimal or String fields except that pressing the spacebar selects an item in a list to display. Continually pressing the spacebar selects items from the list. The list is in alphabetical order. Items in the list may be overridden by typing characters in the field. When the cursor leaves the field, the field contents are incorporated as a new item in the list.

Three fields — System Model, Device Name, Device Class — are not editable and are display-only. They are under the control of SYSPROF and are for your information.

Cursor Movement and Control-Key Functions

If the cursor is after the last character in a field, the right-arrow key or any forward word/field movement key moves the cursor to the beginning of the next editable field.

If the cursor is on the last character of a field, any key moves the cursor to the beginning of the next editable field.

If the cursor is in the first position in a field, the left-arrow key or any backward word/field movement key moves the cursor to the end of the previous editable field.

The up-arrow and down-arrow keys move the cursor from a field on one line to another field on another line within the form. If the cursor is in a field in the first line of the form when you strike the up-arrow key, the cursor is moved to a field in the last line of the form. If the cursor is in a field in the last line of the form when you strike the down-arrow key, the cursor is moved to a field in the first line of the form.

Arrow keys — Each arrow key moves the cursor in the direction indicated by the arrow on the keycap of the arrow key.

[CTRL] a — Moves the cursor to the beginning of the first field on a line in the form.

[CTRL] b — Deletes from the character to the left of the cursor to the beginning of a word or end of a word, whichever is encountered first. A word is composed only of alphanumeric characters.

[CTRL] d — Deletes all characters from the character under the cursor to the end of the field.

[CTRL] e — Moves the cursor to the end of the last field of the form.

[CTRL] f — Moves the cursor to the beginning of the next word or field, whichever is encountered first.

[CTRL] g — Moves the cursor to the end of the last field on a line in the form.

[CTRL] h — Pages to the previous device record.

[CTRL] k — Deletes all characters from the character under the cursor to the end of a word or the beginning of a word, whichever is encountered first.

[CTRL] o Moves the cursor to the beginning of the first field in the next line of the form.

[CTRL] p Pages to the next device record.

[CTRL] r Moves the cursor to the end of the previous word or field, whichever is encountered first.

[CTRL] t Moves the cursor to the beginning of the first field in the form.

[CTRL] u Deletes all characters from the character to the left of the cursor to the beginning of the field.

[CTRL] v Deletes the character in the cursor.

[CTRL] w Moves the cursor to the beginning of the first field in the previous line of the form.

[CTRL] y Deletes the contents of a field.

[CTRL] z Repeats the last function.

[DEL] Deletes the character to the left of the cursor. [DEL] cannot be used to move the cursor from one field to another.

[RETRN] Moves the cursor to the beginning of the next field in the form.

Related CIP Commands

config	Configure a system
nsysprof	Configure a system for networking

talkt

Functional Description

Use this command to connect your terminal directly to another device.

Command Line Syntax

Mnemonic	talkt
Required parameter	Devicename

Switches

Connection control	:allocate	:priority	:dial	:reverse
	:hangup	:userdata=	:logical=	:x25
Data recording and line control	:file=	:remoteecho	:localecho	:converse
		:xonxoff		

Parameters

Devicename	Function	Required. Use this parameter to specify the name of the remote device (usually a terminal or communication port) to which characters from your terminal will be sent.
	Default	None.
	Syntax	Type a device or rotor name in the standard syntax.

Switches

:allocate

Function: Use this switch to allocate a specific device or to select one from the specified rotor list. This switch must be used if you indicate a rotor name for the port to talk to.

Default: Open and use the specified device.

Syntax: Type **:allocate**

:converse

Function: Use this switch to create a conversation between two terminals. All characters entered at either terminal are echoed at their source as well as transmitted to the other terminal. Carriage returns are echoed and sent as carriage return-line feeds.

Default: No echo at the source terminal.

Syntax: Type **:converse**

:dial=

Function: Use this switch to specify a logical name or ASCII string which will be used to place a call on an intelligent communications device. Since different communications devices expect different sequences of characters, consult the owner's manual for the type of device you are using.

Default: Does not issue a dial command on the remote device.

Syntax: Type **:dial=** followed by a string of up to 32 characters which will be sent to the remote device as a dial buffer. If the string contains blanks or special characters, the string must be enclosed in double quotation marks. When used in conjunction with the :X25 SWITCH, only the first 12 characters will be used as a network dial address.

:file=

Function: Use this switch to specify a file to which will be written a copy of all characters received from the remote device.

Default: Displays characters received from the remote device on the screen only.

Syntax: Type **:file=** followed by any valid filename.

:hangup	Function	Use this switch to issue a hangup request to the remote device before returning to the calling program (usually CIP).
	Default	:nohangup
	Syntax	Type **:hangup**
:localecho	Function	Use this switch to echo all characters typed on the local terminal after sending them to the remote device. This switch causes carriage return to be echoed as a carriage return/line feed on the local device, although only a carriage return is sent to the remote device.
	Default	:nolocalecho
	Syntax	Type **:localecho**
:logical=	Function	Use this switch in conjunction with the :allocate switch to assign the name of the allocated device to a logical name instead of displaying it on sys$output.
	Default	The name of the allocated device is displayed on sys$output.
	Syntax	Type **:logical=** followed by a logical name.
:priority	Function	Used in conjunction with the :X25 Switch, this switch specifies expedited transmission as an X.25 calling option when establishing a connection on an X.25 device.
	Default	Normal transmission priority on an X.25 call.
	Syntax	Type **:priority**
:remoteecho	Function	Use this switch to echo all characters typed on the remote device after sending them to sys$output (which is usually the local screen).
	Default	:noremoteecho
	Syntax	Type **:remoteecho**
:reverse	Function	Use this switch in conjunction with the :X25 switch to specify reverse charging as an X.25 calling option when establishing a connection on an X.25 device.
	Default	:noreverse
	Syntax	Type **:reverse**

:userdata=	Function	When used in conjuction with the :X25 SWITCH, this switch allows the user to specify up to 12 characters to be placed in the user data portion of an X.25 call packet.
	Default	Places no data in the user data portion of an X.25 call packet.
	Syntax	Type **:userdata=** followed by up to 12 characters. If spaces or special characters are to be included, the string must be enclosed in quotation marks.
:xonxoff	Function	Use this switch to enable [CTRL] s and [CTRL] q flow control commands to be transmitted to the remote device.
	Default	:noxonxoff
	Syntax	Type **:xonxoff**
:x25	Function	Use this switch to specify that the remote device is an X.25 port. A side effect is that :xonxoff is automatically implied, and that dial and hangup operations are carried out with X.25 packet specifications in mind.
	Default	:nox25
	Syntax	Type **:x25**

Examples

```
> talkt _ttl
```

This command connects your terminal to terminal _TTl. Characters typed on your terminal are transmitted to _TTl. Characters received from _TTl are displayed on your terminal. To exit TALKT, type [CTRL] z, [CTRL] e.

```
> talkt x25lines :alloc :x25 :dial=LA_office :file=record.txt :hangup
```

This command string connects your terminal to an available device allocated from the rotor list called X25LINES. The DIAL Switch creates an X.25 format dial packet using the X.25 address found from the logical name translation of "LA_office". It makes an X.25 call and enters transparent mode between your terminal and whatever is attached to the "LA_office" line. Then it sends all characters received from the remote device to your terminal.

The :file= switch puts a copy of all incoming characters in the file called RECORD.TXT. When TALKT terminates, the :hangup switch issues a hangup command to the device that was allocated from the rotor list and subsequently used as the communication line.

Using Prompts

```
> talkt
Devicename     > _ttl
```

This command performs the same function as the first example.

Notes on Usage

This command is typically used to connect your terminal to a remote computer, either for interactive use or for downloading a file. The :localecho switch can be used when communicating with a remote device or machine which does not do its own echoing. It can be thought of as a half duplex capability. The :remoteecho switch combined with the :localecho switch provides a convenient way for two users to communicate with each other using two terminals.

Note that since both locally typed characters and characters received from the remote device are sent to each screen, it is possible for characters from the two terminals to become intermixed on both screens. Carriage returns are echoed as carriage return-line feeds to your terminal if :localecho is in effect. However, only the carriage return is sent to the remote device. :converse is equivalent to using :localecho and :remoteecho together except that carriage returns are always converted to carriage return-line feeds.

Within TALKT several different "escape" sequences are honored in order to perform special tasks. (An escape sequence is a multiple keystroke sequence which rarely occurs in normal data flow and is used to "escape" into another mode of operation.) These sequences are always started with a [CTRL] z. An optional single digit, n, may follow the [CTRL] z and indicates the escape sequence is to be sent to the nth TALKT away from your TALKT. The optional digit provides a way to talk through the local machine to a second machine, through the second machine to a third machine, etc. and be able to send an escape sequence to the nth machine down the line.

After typing [CTRL] z and the optional digit, additional functions may be specified by typing one of the following:

[CTRL] b Transmit a break character to the remote device.
[CTRL] c Enter a CIP from TALKT. After logging out of the new CIP, return to the same state in TALKT.
[CTRL] e Exit TALKT.
[CTRL] f Close the open file which is a copy of all incoming characters. Prompt for the name of a new file to be created which will contain a copy of all future characters received. If no new filename is entered, characters are displayed only on the screen.
[CTRL] z Transmit a [CTRL] z to the remote device.

When downloading a file from another machine it is sometimes possible for the remote machine to send characters faster than TALKT can receive and process them. For this reason, a method of flow control is often needed for the remote port. TALKT does not change the method of flow control on the remote port. The appropriate flow control method (usually software or hardware) should be set up using "DSTAT :hostsync=" on the remote port before TALKT is invoked. On older machines it is possible to lose incoming characters if disk activity is present during downloading. A suggested method for avoiding character loss is to refrain from using the disk during the download and write the file to the memory disk. Copy the file to a regular disk after the download is complete.

Related CIP Commands

---

alloc	Find and reserve an available device
rotor	Add, delete or list devices in a rotor list
usscopy	Updated system to system copy

tcopy

Functional Description

Use this command to read or write a foreign tape.

Command Line Syntax

Mnemonic	tcopy
Required parameters	Source Destination Block size

Switches

File selection	:before=	:confirm	:exclude=	:mod
	:numfiles=	:skipfile=	:since=	:uic=
File manipulation	:continue	:convert=	:destform	:replace
	:skipblk=	:srceform=	:swap	:tapemark
	:transform=			
Other	:log			

Parameters

Source	Function	Required. Use this parameter to specify the source file(s) for a WMCS (native) source device or to specify the source device for a non-WMCS (foreign) source device.
	Default	None.
	Syntax	For a native device, type a file list. For a foreign device, type a devicename (including the underscore, _).

Destination	Function	Required. Use this parameter to specify the destination file for a native destination device or to specify the destination device for a foreign destination device.
	Default	None.
	Syntax	For a native device type a filename (wildcard symbols are not allowed). For a foreign device, type a devicename (including the underscore).
Block Size	Function	Required. Use this parameter to specify how many bytes of data will be read from the source and written to the destination at a time.
	Default	None.
	Syntax	Type a positive integer no greater than 4096.

Switches

:srceform=	Function	Use this switch to specify the format of source file records for record format conversion when copying. Source file records can be either fixed-length records consisting of a certain number of bytes per record (the records may contain trailing blanks which pad the data in the record out to the required length) or variable-length records which are terminated by a certain character or character sequence. Recognized record terminator characters or character sequences are (ascii) LF (line feed - 0A hexadecimal), CR (carriage return - 0D hex), CRLF (carriage return followed by line feed - 0D0A hex), or NULL (null - 00 hex).
	Default	Fixed-length records whose length is equal to the specified block size.
	Syntax	For fixed-length records, type :srceform= followed by a positive integer specifying the number of bytes per record (which must divide evenly into the specified block size). For variable length records, type :srceform= followed by LF, CR, CRLF, or NULL.

:destform=

Function: Use this switch to specify the format of destination file records when copying. Destination file records can be either fixed-length records consisting of a certain number of bytes per record (the records may contain trailing blanks which pad the data in the record out to the required length) or variable-length records which are terminated by a certain character or character sequence. Recognized record terminator characters or character sequences are (ascii) LF (line feed - 0A hexadecimal), CR (carriage return - 0D hex), CRLF (carriage return/line feed - 0D0A hex), or NULL (null - 00 hex).

Default: Fixed-length records whose length is equal to the specified block size.

Syntax: For fixed-length records, type **:destform=** followed by a positive integer (which must divide evenly into the specified block size). For variable length records, type **:destform=** followed by LF, CR, CRLF, or NULL.

:replace

Function: Use this switch to replace the specified source record terminator with the specified destination record terminator. (In the case of fixed-length records, the record terminator is considered to be the trailing blanks.)

Default: :replace

Syntax: Type **:noreplace**

:convert=

Function: Use this switch to convert ASCII source to EBCDIC in the destination, or to convert EBCDIC source to ASCII in the destination.

Default: Does not do any conversion.

Syntax: Type **:convert=** followed by ASCII or EBCDIC.

:skipfile=

Function: Use this switch to skip a specified number of files on the tape before beginning to copy data.

Default: :skipfile=0

Syntax: Type **:skipfile=** followed by a positive integer to skip forward or a negative interger to skip backward.

:skipblk=	Function	Use this switch to skip forward a specified number of blocks on the tape before beginning to copy data.
	Default	:skipblk=0
	Syntax	Type **:skipblk=** followed by a positive integer.
:swap	Function	Use this switch to exchange each pair of source bytes before writing them to the destination.
	Default	:noswap
	Syntax	Type **:swap**
:continue	Function	Use this switch to continue copying a file even if some of the data is unreadable.
	Default	:nocontinue
	Syntax	Type **:continue**
:tapemark	Function	Use this switch to write a tapemark to the destination tape before copying begins.
	Default	:notapemark
	Syntax	Type **:tapemark**
:numfiles=	Function	Use this switch to specify the number of files to copy from a foreign source tape.
	Default	:numfiles=1
	Syntax	Type **:numfiles=** followed by a positive integer.
:exclude=	Function	Use this switch to exclude files or devices from the Source parameter.
	Default	All files or devices that match the Source parameter are selected.
	Syntax	Type **:exclude=** followed by a list of file or device designations separated by commas. Wildcard characters are allowed.
:uic=	Function	Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.
	Default	Selects all files that match the specified list.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.

:since=	Function	Use this switch to select only those files that match the file list parameter and were created or modified since the specified date and time.
	Default	Selects all files that match the file list parameter.
	Syntax	Type **:since=** followed by a date and/or time in the standard syntax.
:before=	Function	Use this switch to select only those files that match the Source parameter and were created or modified before the specified date and time.
	Default	Selects all files that match the Source parameter.
	Syntax	Type **:before=** followed by a date and/or time in the standard date and time syntax.
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**
:confirm	Function	Use this switch to confirm or deny the desired operation on each file or device.
	Default	:noconfirm
	Syntax	Type **:confirm**
:transform=	Function	Use this switch to specify whether to do record format transformation before or after conversion between ASCII and EBCDIC.
	Default	:transform=AFTER
	Syntax	Type **:transform=** followed by **BEFORE** or **AFTER**.
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.

Examples

```
> tcopy _mt0 games.pas 1024
```

This command copies one file from foreign tape _MT0 (which was written with a block size of 1024 bytes/block) to a WMCS file, GAMES.PAS, in the default directory. No data transformation is performed. Data is copied starting from where the tape is currently positioned and continues until a filemark or the end of the tape. Then this log message appears:

```
_MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
```

```
> tcopy _mt0 games.pas 1024 :srceform=80 :destform=lf
```

This command copies a file from foreign tape _MT0 to the file GAMES.PAS in the default directory. The source file format is fixed-length records of 80 bytes/record. It is converted to the destination file format of variable-length records terminated by an ASCII line feed character (0A hexadecimal).

Trailing blanks (ASCII - 20 hexadecimal) in the 80-byte source records are stripped off and a line feed is appended before copying to the destination. The source file is assumed to be in ASCII.

```
> tcopy _mt0 games.pas 1024 :srceform=80 :destform=lf :convert=ASCII
```

This command copies a file from _MT0 to GAMES.PAS. The source records are transformed from fixed-length to variable-length records AFTER the source is converted to ASCII (from EBCDIC). Then they are copied to the destination.

```
> tcopy _mt0 games.pas 1024 :srceform=80 :destform=lf :noreplace
```

This command copies a file from _MT0 to GAMES.PAS. The trailing blanks in the fixed-length source records are NOT replaced by a line feed. Instead, a line feed is appended to them and then copied to the destination.

```
> tcopy _mt0 games.pas 1024 :numfiles=10
```

This command copies the next 10 files from _MT0 to increasing versions of GAMES.PAS. In other words, if GAMES.PAS doesn't already exist, the first file is GAMES.PAS.1, the next is GAMES.PAS.2, and so on. If there are less than 10 more files on the tape, it copies as many as are there with no errors given. Then the following log messages appear:

```
    _MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
    _MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
    _MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
    _MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
    _MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
    _MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
    _MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
    _MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
    _MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
    _MT0 to _DS0/SOURCE.USERS/GAMES.PAS Copied.
```

```
> tcopy *.* _mt0 1024 :uic=system
```

This command copies all the files with a UIC of "system" in the default directory to foreign tape _MT0. The files are copied onto _MT0 in the order the wildcarding finds them (use the log messages to keep track of the file order). No data transformation is performed. Then the following log messages appear:

```
    _DS0/SOURCE.USERS/GAMES.PAS to _MT0 Copied.
    _DS0/SOURCE.USERS/TOOLS.PAS to _MT0 Copied.
    _DS0/SOURCE.USERS/BUDGET.PAS to _MT0 Copied.
    _DS0/SOURCE.USERS/MEMOS.PAS to _MT0 Copied.
```

Using Prompts

```
> tcopy
Source         > _mt0
Destination    > games.pas
Block size     > 1024
```

This command performs the same function as the first example.

Notes on Usage

If copying directly from one foreign tape to another, their block sizes must be the same. If copying from disk to disk, then the block size is your choice.

Cipher tapes to be read or written must be 9 track, 1600 bpi.

Foreign cartridge tapes cannot be read or written.

Tapes with a block size larger than 4096 bytes per block cannot be read or written.

Foreign tapes must be mounted as class "TapeSpecial".

You need to know the block size and the record format of files on any foreign tape you wish to read. If you do not know the block size of the tape, you can "experiment" with various block sizes. A warning is issued if the actual block size is larger than the one you specify.

Requesting too small a block size causes you to lose data, but requesting too large a block size does not. The record format of the source can frequently be determined by copying off a file, doing no data transformation, and examining the destination file with the DUMP utility.

You can copy from tape to tape, tape to disk, disk to tape, or disk to disk. In all cases tapes can be either foreign or native (WMCS format). Foreign disks cannot be read or written.

You cannot copy from foreign tape to foreign tape if each tape has a different block size. Copy the source tape to disk first and then copy the disk file(s) to the destination tape.

TCOPY requires an integral number of records per block if a fixed-length record size is specified.

If the source record format and the destination record format are the same, TCOPY does a block by block copy without worrying about record formats.

Standard WMCS data files are in ASCII and consist of variable-length records terminated by a line feed 0A hexadecimal). Files in this format can be displayed by the TYPE utility and edited with the VEW editor. WMCS files in other formats (EBCDIC or fixed-length records or variable-length records terminated by other than a line feed) may be displayed using the DUMP utility. Therefore if you copy a file off a foreign tape into a WMCS file, you can display it using DUMP to verify the file is in the desired format.

When copying to a foreign destination tape, a file mark is written after each file copied to the tape.

Related CIP Commands

dump	Display the contents of a file in hexadecimal

time

Functional Description

Use this command to display and set the system time.

Command Line Syntax

Mnemonic	time
Optional parameter	Date and Time
Switches	:prompt

Parameters

Date and Time	Function	Optional. Use this parameter to set the system time clock.
	Default	Display the time only.
	Syntax	Type a WMCS date and time string.

Switches

:prompt	Function	Use this switch to prompt for optional parameters.
	Default	:noprompt
	Syntax	Type :prompt

Examples

```
> time
```

This command displays the date and time, like the following:

```
     Saturday 14-Jan-1984 17:00:14.89
```

Using Prompts

```
> time :prompt
Date and Time    > 5-jan-84_12:00:00.00
```

This command, with the :prompt switch, allows the user to set the system time using a prompt.

Notes on Usage

Times may be specified in several different ways.

Absolute dates are composed of a date and a time separated by an underscore. No spaces are allowed. The date portion may be either a keyword (today, yesterday, tomorrow) or a date with the following syntax:

```
      dd<sep>mm<sep>yy
```

<sep> represents a separator. Use any of the following characters to separate the day, month and year: hyphen, -, slash, /, colon, :, and comma ,.

dd represents the day of the month. Use a numeral between 1 and 31 inclusive.

mm represents the month. Use a numeral from 1 to 12 inclusive, or any abbreviation of the month name (ja, jan, janu, january).

yy represents the year. It is specified as a numeral. If the value of the year is less than 100, it is assumed to be 1900 plus the specified year (83 means 1983). If the value of the year is greater than or equal to 100, then it is interpreted literally as the year.

Defaults are allowed for every field. Fields may be defaulted from the right. For instance, if the year is omitted, the current year is assumed. If the month and year are omitted, the current month and year are assumed. If the day, month and year are omitted, the current day, month and year are assumed.

Here are some sample dates:

today	The current day, month and year
yesterday	Yesterday's day, month and year
tomorrow	The day, month and year for tomorrow
2-febr-83	February 2, 1983
2,f,1983	February 2, 1983
31/12/83	December 31, 1983
2	The second day of the current month and year
5-jan	The fifth day of January of the current year
5,1	The fifth day of January of the current year

The time portion of the absolute date may be the keyword "current" or the time specified in the following format:

```
hh<sep>mm<sep>ss<sep>tt
```

hh represents the hour of the day in a 24-hour clock format. Use a numeral from 0 through 23 inclusive.

mm represents the minute of the hour. Use a numeral from 0 through 59 inclusive.

ss represents the second of the minute. Use a numeral from 0 through 59 inclusive.

tt represents the fraction of a second (tick). Use a numeral from 0 to 99 inclusive.

As with the date, defaults are allowed for every field. Fields are defaulted from the right. For instance, if the ticks are omitted, zero is assumed. If the seconds and ticks are omitted, zero is assumed. If the minutes, seconds and ticks are omitted, zero is assumed. If the hours, minutes, seconds and ticks are omitted, all zeros are assumed.

Here are some sample times:

current	The current hour, minute, second and tick
17:20:10	10 seconds after 5:20 p.m.
17,20-10	10 seconds after 5:20 p.m.
9/14/3/50	3.5 seconds after 9:14 a.m.
9-14,3:5	3.05 seconds after 9:14 a.m.
9	Exactly 9:00 a.m.
9-5	9:05 a.m.

Here are some sample dates and times:

today_9	The current day, month and year at 9:00 a.m.
yesterday_9-15	Yesterdays day, month and year at 9:15 a.m.
tomorrow_10:40	The day, month and year for tomorrow at 10:40 a.m.
2-febr-83_14-10	February 2, 1983 at 2:10 p.m.
2,f,1983	February 2, 1983 at 00:00 a.m.
31/12/83_12:15	December 31, 1983 as 12:15 p.m.
2_11	The second day of the current month and year at 11:00 a.m.
5,jan_2,3	The fifth day of january of the current year at 2:03 a.m.
5/1_2:3	The fifth day of january of the current year at 2:03 a.m.
_	The current day, month and year at 00:00 a.m.
_current	Now
3	The third day of the current month and year at 00:00 a.m..
_15,30	The current date and time at 3:30 p.m.

Relative dates represent a time relative to the current date and time. The format for relative dates is:

```
+dd_hh<sep>mm<sep>ss<sep>tt
     or
-dd_hh<sep>mm<sep>ss<sep>tt
```

The (+) or (-) indicates whether the date and time is after or before the current date and time. For instance, +2_0:0:0 represents two days from now (now plus 48 hours).

dd represents the number of days before or after the current date and time. Use any numeral from 0 through 32767.

hh represents the number of hours. Use any numeral from 0 through 32767.

mm represents the number of minutes. Use any numeral from 0 through 32767.

ss represents the number of seconds. Use any numeral from 0 through 32767.

tt represents the fraction of a second. Use any numeral from 0 through 99.

There are several short forms of representing relative time. If any field is omitted, it is assumed to be zero.

Here are some examples of relative times:

+2	Represents two hours from now
+0:5	Represents five minutes from now
+0:0:30	Represents 30 seconds from now
-48	Represents two days ago (48 hours ago)
+0,60	Represents one hour (sixty minutes) from now
-1_	Represents yesterday at this same time (one day ago)
+1_2	Represents now plus one day, two hours
+1_5/20/45	Represents now plus one day, five hours, twenty minutes and forty-five seconds
+0	Represents now

Related CIP Commands

None.

translit

Functional Description

Use this command to copy source file(s) to destination file(s) with character transliteration.

Command Line Syntax

Mnemonic	translit
Required parameters	Source Destination Character Set
Optional parameter	Character Set 2

Switches

File selection	:before= :exclude=	:mod :uic=	:confirm	:since=
Destination file	:builddir :owner	:concat :preserve=	:edit=	:protection=
Transliteration	:delete	:squeeze		
Other	:log	:rewind		

Parameters

Source	Function	Required. Use this parameter to specify the source file(s).
	Default	None.
	Syntax	Type the standard syntax for a file list.
Destination	Function	Required. Use this parameter to specify the destination file(s).
	Default	None.
	Syntax	Type the standard syntax for a file designation. Wildcard symbols are allowed.
Character Set	Function	Required. Use this parameter to specify the list of characters to be substituted.
	Default	None.
	Syntax	Type a list of characters with no separators. Note that this is a list of characters not separated by anything. It can include ranges specified as a-z and exclusions specified as ^A-Z or 0-9^3, etc. Characters can also be specified by a "\" followed by 2 hexadecimal digits indicating a byte value. Any value from \00 thru \FF is acceptable. (Essentially the same syntax is true for wildcarding.) If you use CIP metacharacters in this parameter, the Character Set must be enclosed in double quotation marks. If Character Set 2 (the next parameter) is specified, each character in the first Character Set Parameter on input is converted on output to the character in the Character Set 2 Parameter whose list position is the same. (See description of Character Set 2 Parameter.) When ranges are specified, list position is determined as if the characters were listed individually. For example, "d-h" means "defgh" and a-g^b-d" means "aefg".

Character Set 2	Function	Optional. Use this parameter to specify the list of characters which the first set is converted to.
	Default	None.
	Syntax	Type a list of characters with no separators. Note that if the number of characters in Character Set 2 is shorter than in the first Character Set, the last character in Character Set 2 is repeated until the lengths match. If the first Character Set is shorter, the extra characters in Character Set 2 have no effect on the conversion.

Switches

:before=	Function	Use this switch to select only those files that match the Source parameter and were created or modified before the specified date and time.
	Default	Selects all files that match the Source parameter.
	Syntax	Type **:before=** followed by a date and/or time in the standard date and time syntax.
:builddir	Function	Use this switch to create destination directories automatically if they do not exist.
	Default	:nobuilddir
	Syntax	Type **:builddir**
:concat	Function	Use this switch to cause all of the source files to be concatenated into a single destination file.
	Default	:noconcat
	Syntax	Type **:concat**
:confirm	Function	Use this switch to confirm or deny the desired operation on each file or device.
	Default	:noconfirm
	Syntax	Type **:confirm**

:edit=	Function	Use this switch to specify a series of edits to be performed on the destination file designation. The edits are specified as a series of "find and switch" strings. The destination file designation is searched for the first occurrence of the "find" string. If it is found, it is replaced by the "switch" string.
	Default	No edits are performed.
	Syntax	Type **:edit=** followed by a list of "find and switch" strings separated by commas. The find string is separated from the switch string by a colon. Either one or both of the strings may be empty.
:exclude=	Function	Use this switch to exclude files or devices from the Source parameter.
	Default	All files or devices that match the Source parameter are selected.
	Syntax	Type **:exclude=** followed by a list of file or device designations separated by commas. Wildcard characters are allowed.
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**
:owner=	Function	Use this switch to specify the owner (UIC) of the destination file.
	Default	The UIC of the user (process) creating the file.
	Syntax	Type **:owner=** followed by a UIC or a username. Usernames are converted to the UIC they represent.

:preserve=

Function — Use this switch to preserve certain attributes of the original file when a new copy of it is created. The preservable aspects are: creation date, owner, protection.

Default — The original file's attributes are not preserved in the new file, i.e., the new file is created with the current date and time, the owner is the current user, and the protection is the user's default protection mask.

Syntax — Type **:preserve=** followed by any combination of the keywords **CREATED**, **OWNER**, **PROTECTION**, or **ALL** separated by commas.

:protection=

Function — Use this switch to specify the protection mask for the destination file.

Default — The default protection mask of the user (process) creating the file.

Syntax — Type **:protection=** followed by a protection mask.

Note that only those fields of the protection mask that are specified are altered. Fields not specified keep their original value. Suppose you typed the following value for this switch:

```
:protection=s:,o:rwe
```

This switch value allows the system no privilege, while the owner is granted read, write and execute privilege. The public and group privileges are unchanged.

:rewind

Function — Use this switch to rewind the tape device.

Default — :norewind

Syntax — Type **:rewind**

:since=

Function — Use this switch to select only those files that match the Source parameter and were created or modified since the specified date and time.

Default — Selects all files that match the Source parameter.

Syntax — Type **:since=** followed by a date and/or time in the standard syntax.

:uic=	Function	Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.
	Default	Selects all files that match the specified list.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.
:delete	Function	Use this switch to delete or remove the characters in the Character Set Parameter when copied to the destination file. Any character in the Character Set on input is deleted on output to the destination file. If Character Set 2 is present, it is ignored by this switch.
	Default	:nodelete
	Syntax	Type **:delete**
:squeeze	Function	Use this switch to reduce sequences of specified repeated characters to a single character.
	Default	:nosqueeze
	Syntax	Type **:squeeze**

Examples

```
> translit copy.txt * \00 \0a
```

This command copies the file COPY.TXT in the default directory to a new version of itself which contains (ASCII) line feed characters (0a hexadecimal) where nulls (00 hexadecimal) had been in the source file. In other words, the file COPY.TXT which contained null-terminated lines now contains (proper WMCS) linefeed-terminated lines.

```
> translit copy.txt * ^a-z^A-Z :delete
```

This command copies COPY.TXT to a new version which only contains the upper- and lowercase alphabetic characters. All other characters are deleted.

```
> translit copy.txt copy.doc \0d \0a :squeeze
```

This command copies COPY.TXT to COPY.DOC, replacing all carriage return (ASCII 0d hexadecimal) characters with line feed ASCII 0a) characters and condensing any sequences of repeated line feed characters down to one line feed character. If COPY.TXT were a file containing lines terminated by <CR><LF> (ASCII 0d0a hexadecimal, carriage return followed by line feed), this command would change all the carriage returns to line feeds and delete the extra line feeds.

Using Prompts

```
> translit
Source        > copy.txt
Destination   > *
Character set > \00
```

This command copies COPY.TXT to a new version of itself without doing any character manipulation. (The Character Set Parameter accomplishes nothing without Character Set 2 or the :delete or :squeeze switches.)

Notes on Usage

TRANSLIT is like the COPY command with character transliteration features added. Please refer to the COPY command description for examples of the non-transliteration parameters and switches.

The DUMP command may be used to see the actual byte values in a file as well as the ASCII text.

Related CIP Commands

copy	Copy a file
dump	Display the contents of a file in hexadecimal

type

Functional Description

Use this command to display the contents of text files.

Command Line Syntax

Mnemonic	type			
Required parameter	File List			
Switches				
File selection	:before= :uic=	:exclude=	:mod	:since=
Display of files	:header	:lines=	:number	:pause
Other	:confirm :override	:rewind :pause	:continuous :raw	:log

Parameters

File List	Function	Required. Use this parameter to specify the files to be typed.
	Default	None.
	Syntax	Type a list of file designations separated by commas. Wildcard symbols are allowed.

Switches

:before=	Function	Use this switch to select only those files that match the file list parameter and were created or modified before the specified date and time.
	Default	Selects all files that match the file list parameter.
	Syntax	Type **:before=** followed by a date and/or time in the standard date and time syntax.
:confirm	Function	Use this switch to confirm or deny the desired operation on each file or device.
	Default	:noconfirm
	Syntax	Type **:confirm**
:continuous	Function	Use this switch to display a file being written. Because this switch ignores the end of the file, you must use [CTRL] c to terminate TYPE. Do not use this switch with the :lines=, :pause, or :number switches.
	Default	:nocontinuous
	Syntax	Type **:continuous**
:exclude=	Function	Use this switch to exclude files or devices from the File List parameter.
	Default	All files or devices that match the File List parameter are selected.
	Syntax	Type **:exclude=** followed by a list of file or device designations separated by commas. Wildcard characters are allowed.
:override	Function	Use this switch to display a file regardless of the file type.
	Default	:nooverride
	Syntax	Type **:override**
:header	Function	Use this switch to display the full file specification as a header.
	Default	:header
	Syntax	Type **:noheader**

:lines=	Function	Use this switch to specify the lines to be displayed.
	Default	All lines will be displayed.
	Syntax	Type **:lines=** followed by a number range. Line numbering begins at 1.
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	:log
	Syntax:	Type **:nolog**
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**
:number	Function	Use this switch to specify that a line number is to be added at the left margin of each line that is displayed.
	Default	:nonumber
	Syntax	Type **:number**
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when you press any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:raw	Function	Use this switch to specify that the data being written are not to be modified or examined.
	Default	:noraw
	Syntax	Type **:raw**
:rewind	Function	Use this switch to rewind the tape device prior to searching for the file to type.
	Default	:norewind
	Syntax	Type **:rewind**

:since=

Function — Use this switch to select only those files that match the file list parameter and were created or modified since the specified date and time.

Default — Selects all files that match the file list parameter.

Syntax — Type **:since=** followed by a date and/or time in the standard syntax.

:uic=

Function — Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.

Default — Selects all files that match the specified list.

Syntax — Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> type aaa.txt
```

This command displays the contents of file AAA.TXT:

```
File __NODEA_DS0/USERS/AAA.TXT.1
     aaaaaaaa
     bbbbbbbb
     cccccccc
     dddddddd
     eeeeeeee
     ffffffff
     gggggggg
     hhhhhhhh
```

TYPE displays the contents of a file, or group of files, on your terminal screen, but does not allow you to emend the contents.

The example displays the most recent version of AAA.TXT in the default directory.

When the text of the specified file begins to scroll past on your screen, use [CTRL] s to temporarily stop the scrolling, and CTRL] q to cause it to resume. [CTRL] c terminates the execution of the command and returns you to the CIP command line.

```
> type aaa.txt :number
```

The :number switch numbers the lines in the text file at the front of each line. The contents of the file is not changed. This kind of report appears when you strike [RETRN]:

```
File __NODEA_DS0/USERS/AAA.TXT.1
        1 aaaaaaaa
        2 bbbbbbbb
        3 cccccccc
        4 dddddddd
        5 eeeeeeee
        6 ffffffff
        7 gggggggg
        8 hhhhhhhh
```

```
> type aaa.txt :lines=1-3,5,7- :noheader :number
```

Including the :noheader switch suppresses the display of the header at the top of the file.

The :lines switch specifies the lines of the file to be displayed. In this example lines 1 through 3, line 5, and line 7 through the last line are specified. These lines appear in the following display:

```
File _DS0/USERS/AAA.TXT.1
        1 aaaaaaaa
        2 bbbbbbbb
        3 cccccccc
        5 eeeeeeee
        7 gggggggg
        8 hhhhhhhh
```

Using Prompts

```
> type
File list   > *.txt
```

This command displays the contents of the latest version of each file with an extension of .TXT in the default directory.

Notes on Usage

Use TYPE only with data files.

Use the :override switch to display other types of files, or use the DUMP Command.

Related CIP Commands

dump	Display the contents of a file in hexadecimal
vew	Display and edit the contents of a file

typeb

Functional Description

Use this command to display a message in block letters.

Command Line Syntax

Mnemonic	typeb
Optional parameter	Message

Parameters

Message	Function	Optional. Use this parameter to specify the message to be displayed.
	Default	None.
	Syntax	Type up to 70 characters and spaces. The character limit for any word within the text is 10. Words longer than this are displayed, but may become illegible on a standard 80-character width terminal. Each word or character bounded by spaces is displayed on a separate line.

Switches

None.

Examples

> typeb aBc

This command generates the following display on your terminal:

```
            BBBBBB
             B    B
             B    B
      aaa a  BBBBB   cccc
     a   aa  B    B c
     a   aa  B    B c
      aaa a BBBBBB   cccc
```

> typeb aBc DeF

This command generates the following display on your terminal:

```
            BBBBBB
             B    B
             B    B
      aaa a  BBBBB   cccc
     a   aa  B    B c
     a   aa  B    B c
      aaa a BBBBBB   cccc

     DDDDDD         FFFFFFF
      D    D        F
      D    D        F
      D    D  eee   FFFFF
      D    D e  ee  F
      D    D eee    F
     DDDDDD   eeee  F
```

> typeb aBc@DeF

Note that the at sign, @, is treated like a space but allows you to put multiple words on the same line. This command generates the following display:

```
             BBBBBB                  DDDDDD         FFFFFFF
              B    B                  D    D        F
              B    B                  D    D        F
       aaa a  BBBBB   cccc            D    D  eee   FFFFF
      a   aa  B    B c                D    D e  ee  F
      a   aa  B    B c                D    D eee    F
       aaa a BBBBBB   cccc           DDDDDD   eeee  F
```

Using Prompts

None.

Notes on Usage

Up to three lines of text will fit on a standard 24 line terminal screen.

Because the at sign is used to make spaces between words on the same line, this character cannot be displayed.

Related CIP Commands

None.

typemrl

Functional Description

Use this command to display linker records in a .MRL file.

Command Line Syntax

Mnemonic	typemrl			
Required parameter	File List			
Switches				
File selection	:before= :mod	:since=	:exclude	:uic
Contents of the display	:addresses :ends :globalconstants :relocationtype	:locallabels :programs :impure	:all :filename :references :symboltype	:modules :pure :globallabels :localconstants
Format	:header	:pause	:log	

Parameters

File List	Function	Required. Use this parameter to specify the names of the .MRL files to be examined.
	Default	None.
	Syntax	Type the standard syntax for a WMCS file designation. Wildcard characters are allowed. The file extension ".MRL" is not automatically appended to file names but must be given.

Switches

:addresses	Function	Use this switch to have TYPEMRL display the linker record address field.
	Default	:noaddresses
	Syntax	Type **:addresses**
:all	Function	Use this switch to display all information from all linker records. Specifying this switch is the same as specifying all of the other linker record display switches.
	Default	:noall
	Syntax	Type **:all**
:before=	Function	Use this switch to select only those files that match the File List parameter and were created/modified before the specified date and time.
	Default	Selects all files that match the File List parameter.
	Syntax	Type **:before=** followed by a date and/or time in the standard date and time syntax.
:ends	Function	Use this switch to display "module end" (type 8) and "program end" (type 9) linker records. Only the "end" records are displayed for corresponding displayed "module start" (type 1) and "program start" (type 0) records.
	Default	:noends
	Syntax	Type **:ends**
:exclude=	Function	Use this switch to exclude files from the File List parameter.
	Default	Selects all files that match the File List parameter.
	Syntax	Type **:exclude=** followed by a list of file designations, separated by commas, any one of which may contain wildcard characters.

:filename	Function	Use this switch to display the file name on each line of the display. It will always be first on the line.
	Default	:nofilename
	Syntax	Type **:filename**
:globalconstants	Function	Use this switch to display "global constant" (type 5) linker records.
	Default	:noglobalconstants
	Syntax	Type **:globalconstants**
:globallabel	Function	Use this switch to display "global label" (type 6) linker records.
	Default	:globallabels
	Syntax	Type **:globallabels**
:header	Function	Use this switch to produce column headings and file name messages for TYPEMRL output.
	Default	:header
	Syntax	Type **:noheader**
:impure	Function	Use this switch to enable the display of impure code (type I) records.
	Default	:noimpure
	Syntax	Type **:impure**
:localconstants	Function	Use this switch to display "local constant" (type 3) linker records.
	Default	:nolocalconstants
	Syntax	Type **:localconstants**
:locallabels	Function	Use this switch to display "local label" (type 4) linker records.
	Default	:nolocallabels
	Syntax	Type **:locallabels**

:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.
:mod	Function	Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
	Default	:nomod
	Syntax	Type **:mod**
:modules	Function	Use this switch to display "module" start (type 1) linker records.
	Default	:modules
	Syntax	Type **:nomodules**
:pause	Function	Use this switch to pause after a screen of output has been produced. Pressing any key continues the display.
	Default	:pause
	Syntax	Type **:nopause**
:programs	Function	Use this switch to display "program" start (type 0) linker records.
	Default	:programs
	Syntax	Type **:noprograms**
:pure	Function	Use this switch to enable the display of pure code (type P) records.
	Default	:nopure
	Syntax	Type **:pure**
:references	Function	Use this switch to display symbol reference (type 7) linker records.
	Default	:noreferences
	Syntax	Type **:references**

:relocationtype Function Use this switch to display linker record relocation type. The relocation type is either (A) absolute or (R) relocatable.

Default :norelocationtype

Syntax Type **:relocationtype**

:since= Function Use this switch to select only those files that match the File List parameter and were created/ modified since the specified date and time.

Default Selects all files that match the File List parameter.

Syntax Type **:since=** followed by a date and/or time in the standard date and time syntax.

:symboltype Function Use this switch to display the type of symbol or record a linker record contains. The types displayed are:

Symbol Type	Linker Record
Program Start	Type 0
Module Start	Type 1
Local Constant	Type 3
Local Label	Type 4
Global Constant	Type 5
Global Label	Type 6
Reference to	Type 7
Module End	Type 8
Program End	Type 9
Impure	Type I
Pure	Type P

Default :symboltype

Syntax Type **:nosymboltype**

:uic= Function Use this switch to select only those files that are part of the given list and are owned by the specified user or list of users.

Default Selects all files that match the File List parameter.

Syntax Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> typemrl pascal.prt/init.mrl :address :end :reference
```

This command produced the following output:

```
    Scanning symbols for __MEPH_DS0/PASCAL.PRT/INIT.MRL.2

    Address   Type of Symbol   Symbol Name
    --------  ---------------  ----------------------------
    00000000  Module Start     RRR000
    00000058  Reference to     RRR009
    00000066  Reference to     RRR010
    0000006C  Reference to     RRRglob
    0000007A  Reference to     RRR009
    00000088  Reference to     RRR011
    0000008E  Reference to     RRRglob
    00000094  Reference to     RRRglob
    0000009A  Reference to     heap
    000000A6  Reference to     _HEAP
    000000B2  Reference to     RRRtrapv
    000000BA  Reference to     _SETTRP
    000000C4  Reference to     RRRchk
    000000CC  Reference to     _SETTRP
    000000D6  Reference to     RRRdiv0
    000000DE  Reference to     _SETTRP
    00000117  Module End
```

Using Prompts

```
> typemrl
File list     > sys$disk/pascal.prt/init.mrl
```

This command performs the same function as the example.

Notes on Usage

If :all is specified, any other switches explicitly given with "no" starting the switch value will override the ":all" value for the specified switches.

Related CIP Commands

dump	Dump file contents

userprof

Functional Description

Use this command to set up user accounts on a system and edit the records in the User Authorization File (UAF.DAT).

Command Line Syntax

Mnemonic	userprof	
Optional parameter	Directory	
Switches	:setupin=	:setupout=

Parameters

Directory	Function	Optional. Use this parameter to specify the destination directory for the UAF.DAT file.
	Default	sys$disk/syslib/
	Syntax	Type a standard directory designation.

Switches

:setupin=	Function	Use this switch to specify an alternate setup file to use for sys$input. If you specify a directory path, USERPROF looks in that directory for the setup file based on the terminal type. If you specify an explicit file, it uses that file.
	Default	Use the setup file in sys$disk/syslib.setup/ determined by the terminal type, e.g., if your

terminal is a T7000, then the file used is sys$disk/syslib.setup/setup252.sys.

Syntax Type **:setupin=** followed by a filename.

:setupout= Function Use this switch to specify an alternate setup file to use for sys$output. If you specify a directory path, USERPROF looks in that directory for the setup file based on the terminal type. If you specify an explicit file, it uses that file.

Default Use the setup file in sys$disk/syslib.setup/ determined by the terminal type, e.g., if your terminal is a T7000, then the file used is sys$disk/syslib.setup/setup252.sys.

Syntax Type **:setupout=** followed by a filename.

Examples

> userprof

This command allows the system manager to set up the profiles of the user accounts on the system by editing a file called UAF.DAT. It generates a display like the following:

```
USERNAME     :  DEFAULT                                No password specified
ACCOUNT NAME:                                          PRIORITY     : 7
UIC [U,G]    :  [0000,0000]                            TIMESLICE    : 60
PROTECTION   :  S:  RE,P:    ,G:  RE,O:DWRE
OWNER        :
DIRECTORY    :  SYS$DISK/SYSLIB/
COMMAND LINE:   CIP @SYS$DISK/SYSLIB/LOGON.COM


PRIVILEGE    :  NOSETPRIV      NOSYSTEM       NOREADPHYS      NOWRITEPHYS
                NOSETPRIOR     NOCHNGSUPER    NOBYPASS        NOOPERATOR
                NOALTUIC       NOWORLD        NOGROUP           NETWORK
                NOSETATTR

ATTRIBUTE    :    SWAPPABLE    NOPREZEROMEM   NOPOSTZEROMEM   NODESENCRYPT
                NOFASTENCRYPT    WATCHDOG
                NOUSER1        NOUSER2        NOUSER3         NOUSER4

SECURITY     :

---- This record is used as a template in adding new users to the system. ----
```

```
> userprof _dx0/syslib/
```

This command generates a UAF.DAT file in the directory _dx0/syslib/ and generates the same display as in the previous example.

Using Prompts

None.

Notes on Usage

The USERPROF command should only be used by the system manager.

NOTE: Do not use the VEW command to access /SYSLIB/UAF.DAT.

UAF.DAT is like a card file drawer in which there is a card, or record, containing information on each user. Each card is a user account. When a person tries to log on to the system, the WMCS searches SYS$DISK/SYSLIB/ UAF.DAT to find an account whose username matches the characters typed in response to the prompt for a username. When the WMCS finds the account matching that username, the WMCS uses the information in the User Authorization Record to:

1. Give the user access to the system.
2. Create a user process for the person logging on.

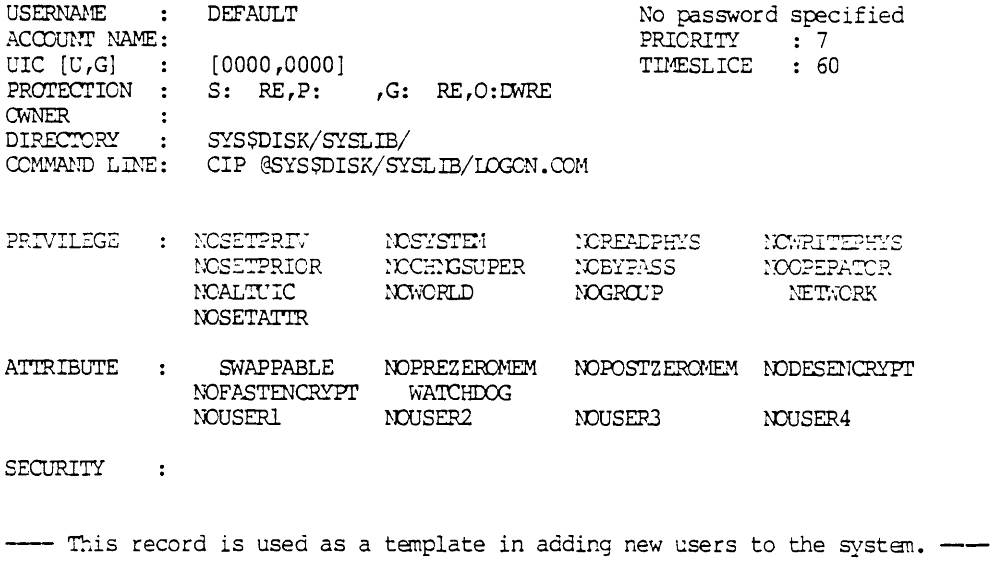
Each record in UAF.DAT is formatted so that to create or edit a user account you move the cursor to the appropriate field and fill in the blanks. However, to fill in some of the fields you must use a function, i.e., a command involving the [ESC] key.

Detailed information on creating and editing user accounts is presented under these headings:

The User Authorization Record (UAR)
Functions Available in USERPROF
Cursor Movement and Control-Key Functions
Miscellanea

The User Authorization Record (UAR)

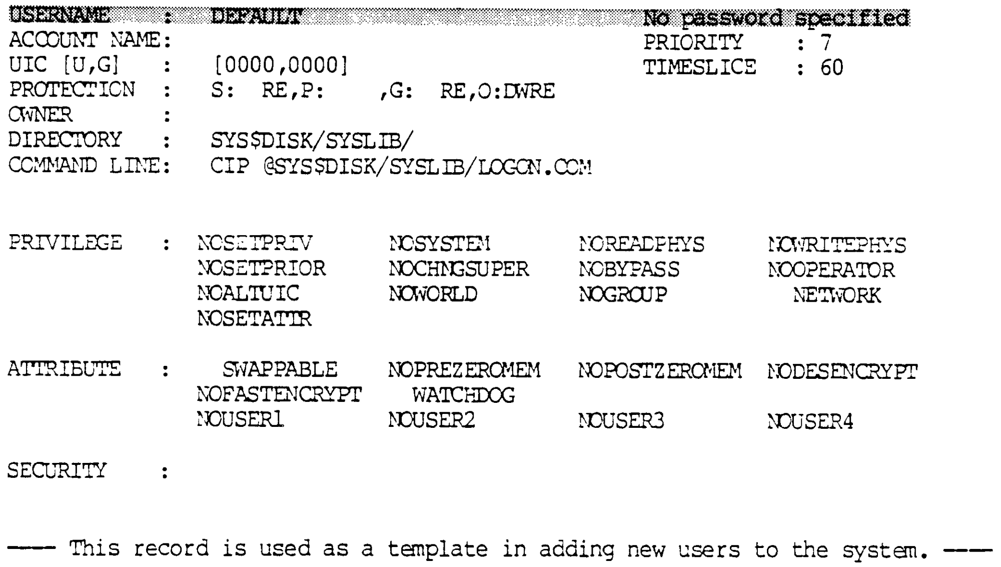
This is what your screen looks like after you type USERPROF on the CIP command line and strike [RETRN]:



```
USERNAME     :   DEFAULT                                   No password specified
ACCOUNT NAME:                                              PRIORITY    : 7
UIC [U,G]    :   [0000,0000]                               TIMESLICE   : 60
PROTECTION   :   S:  RE,P:    ,G:  RE,O:DWRE
OWNER        :
DIRECTORY    :   SYS$DISK/SYSLIB/
COMMAND LINE:    CIP @SYS$DISK/SYSLIB/LOGON.COM


PRIVILEGE    :   NOSETPRIV        NOSYSTEM         NOREADPHYS       NOWRITEPHYS
                 NOSETPRIOR       NOCHNGSUPER      NOBYPASS         NOOPERATOR
                 NOALTUIC         NOWORLD          NOGROUP            NETWORK
                 NOSETATTR

ATTRIBUTE    :     SWAPPABLE      NOPREZEROMEM     NOPOSTZEROMEM    NODESENCRYPT
                 NOFASTENCRYPT      WATCHDOG
                 NOUSER1          NOUSER2          NOUSER3          NOUSER4

SECURITY     :

---- This record is used as a template in adding new users to the system. ---
```

This is the first record in UAF.DAT. This display (the screenful of material that appears whenever you execute USERPROF) is a sample record, i.e., no account is assigned to the DEFAULT record. When an account is created, data from the DEFAULT record are copied to the new account record.

The following is a line-by-line explanation of the contents of each record in the UAF.DAT file (information on how to fill in or edit each line appears immediately after the explanation of that field):



```
USERNAME      :  DEFAULT                                  No password specified
ACCOUNT NAME:                                             PRIORITY      : 7
UIC [U,G]     :  [0000,0000]                              TIMESLICE     : 60
PROTECTION    :  S:  RE,P:     ,G:  RE,O:DWRE
OWNER         :
DIRECTORY     :  SYS$DISK/SYSLIB/
COMMAND LINE:    CIP @SYS$DISK/SYSLIB/LOGON.COM


PRIVILEGE     :  NOSETPRIV       NOSYSTEM        NOREADPHYS      NOWRITEPHYS
                 NOSETPRIOR      NOCHNGSUPER     NOBYPASS        NOOPERATOR
                 NOALTUIC        NOWORLD         NOGROUP           NETWORK
                 NOSETATTR

ATTRIBUTE     :    SWAPPABLE     NOPREZEROMEM    NOPOSTZEROMEM   NODESENCRYPT
                 NOFASTENCRYPT     WATCHDOG
                 NOUSER1         NOUSER2         NOUSER3         NOUSER4

SECURITY      :


---- This record is used as a template in adding new users to the system. ----
```

Username and Password Fields

The username you specify on this line is what the user types in response to the prompt for a username when he logs on to the system.

"No password specified" means, until you specify a password for the user account, the user can log on to the system by typing the username and striking [RETRN]. When a password is specified, "Password specified" appears in the upper right-hand corner of the record.

Follow this procedure to create a new UAR.

Step 1 Strike [ESC][ESC].

Step 2 Type **ad** on the USERPROF function line. This prompt appears on the function line:

```
New username >
```

Step 3 Type the username to be assigned to the account. You are limited to 9 characters.

Step 4 Strike [RETRN].

A username must be unique. USERPROF verifies the username is unique. Following the validation, a new record is displayed on the screen with the username you typed in the username field.

Follow this procedure to change the username on an existing account:

Step 1 Strike [ESC][ESC].

Step 2 Type **cn** on the USERPROF function line. This prompt appears on the screen:

```
New username >
```

Step 3 Type the new username. You are limited to 9 characters.

Step 4 Strike [RETRN]. If the username is validated, the new username is displayed in the username field.

Follow these steps to assign or reassign a password:

Step 1 Strike [ESC][ESC].

Step 2 Type **sp** on the USERPROF function line. This prompt appears on the function line:

```
Password >
```

Step 3 Type the password. The echo is suppressed.

Step 4 Strike [RETRN]. This prompt appears at the bottom of the screen:

```
Verify password >
```

Step 5 Type the password again.

Step 6 Strike [RETRN]. If you typed the password correctly, this message appears at the bottom of the screen:

```
The password has been changed
```

Once the password is assigned or reassigned, this statement appears in the upper right-hand corner of the UAR:

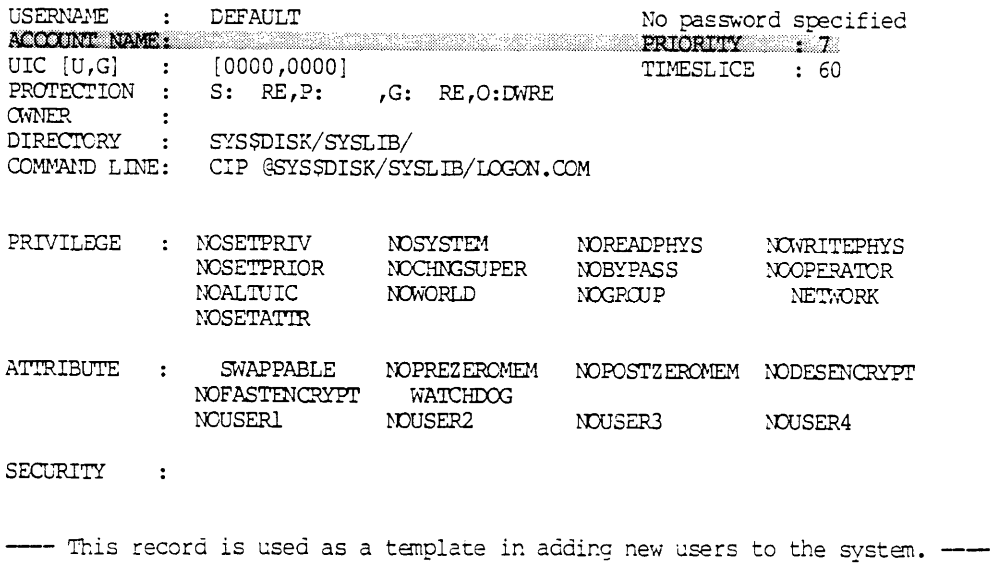
```
Password specified
```

There is no way to find out the password assigned to an account. If the user forgets the password, the system manager must use USERPROF to reassign a password to the account.

If you want to remove the password from an account, just strike [RETRN] in response to the prompt for the password in Step 3.

This statement appears in the upper right-hand corner of the UAR:

No password specified



```
USERNAME     :   DEFAULT                                  No password specified
ACCOUNT NAME:                                             PRIORITY      : 7
UIC [U,G]    :   [0000,0000]                              TIMESLICE     : 60
PROTECTION   :   S:  RE,P:     ,G:  RE,O:DWRE
OWNER        :
DIRECTORY    :   SYS$DISK/SYSLIB/
COMMAND LINE:    CIP @SYS$DISK/SYSLIB/LOGON.COM


PRIVILEGE    :  NOSETPRIV       NOSYSTEM        NOREADPHYS      NOWRITEPHYS
                NOSETPRIOR      NOCHNGSUPER     NOBYPASS        NOOPERATOR
                NOALTUIC        NOWORLD         NOGROUP           NETWORK
                NOSETATTR

ATTRIBUTE    :    SWAPPABLE     NOPREZEROMEM    NOPOSTZEROMEM   NODESENCRYPT
                NOFASTENCRYPT     WATCHDOG
                NOUSER1         NOUSER2         NOUSER3         NOUSER4

SECURITY     :


---- This record is used as a template in adding new users to the system. ---
```

Account Name and Priority Fields

Use the ACCOUNT NAME called for on this line to charge the owner of the account for any expenses incurred by the use of the account. Use the arrow keys to move to this field, then type in the desired account name. This field is limited to 31 characters.

The PRIORITY field allows you to specify the default priority for the user's processes. Type a number from 0 to 15 for the priority; 0 is the highest priority. Higher priority processes have more frequent access to the Central Processing Unit, or processor.

In other words, inasmuch as the WMCS does not complete an entire process before going to the next, but allots a little time to each process until all processes are completed, the priority assigned to each process determines the frequency with which it is scheduled. Thus, higher priority processes tend to be completed more rapidly.

To edit this line, use the arrow keys to position the cursor and then type the values you wish.

```
USERNAME     :   DEFAULT                                    No password specified
ACCOUNT NAME:                                               PRIORITY      : 7
UIC [U,G]    :   [0000,0000]                                TIMESLICE     : 60
PROTECTION   :   S:  RE,P:     ,G:  RE,O:DWRE
OWNER        :
DIRECTORY    :   SYS$DISK/SYSLIB/
COMMAND LINE:    CIP @SYS$DISK/SYSLIB/LOGON.COM


PRIVILEGE    :  NOSETPRIV      NOSYSTEM       NOREADPHYS     NOWRITEPHYS
                NOSETPRIOR     NOCHNGSUPER    NOBYPASS       NOOPERATOR
                NOALTUIC       NOWORLD        NOGROUP          NETWORK
                NOSETATTR

ATTRIBUTE    :    SWAPPABLE    NOPREZEROMEM   NOPOSTZEROMEM  NODESENCRYPT
                NOFASTENCRYPT    WATCHDOG
                NOUSER1        NOUSER2        NOUSER3        NOUSER4

SECURITY     :

---- This record is used as a template in adding new users to the system. ----
```

UIC and Timeslice Fields

The user identification code (UIC) can be displayed in the report generated by the PSTAT Command so you can identify the user that belongs to a particular process. The first four digits are for the user identification number; the second four are for the group identification number. Both fields of the UIC are in hexadecimal. A UIC of [0000,0000] is not allowed.

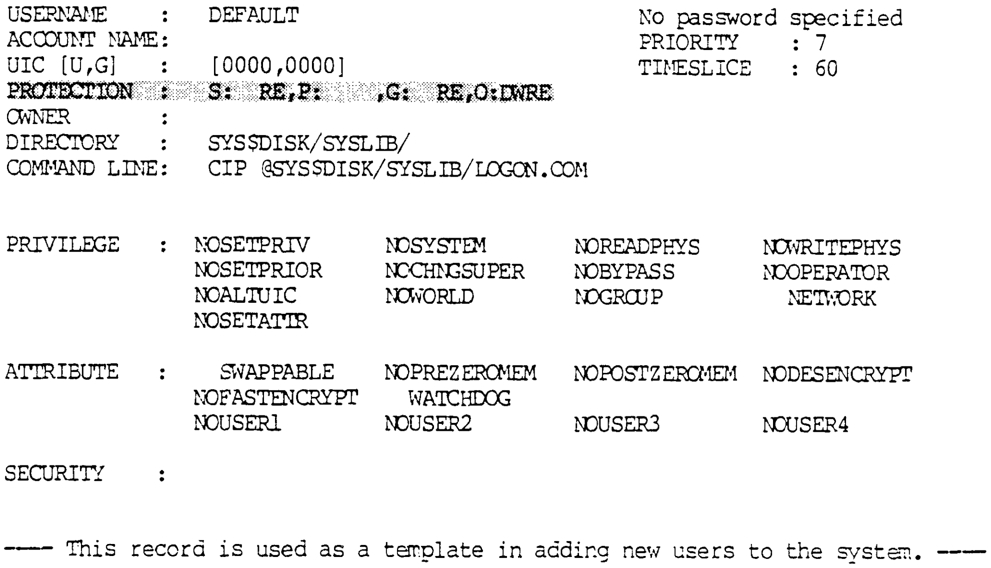
You can group the users on your system to suit your needs. For example, if Brad, Rick, and Jerry are involved primarily with data entry, you may wish to assign them to the same group. Thus Brad might have a UIC of [0001,0004], i.e., user no. 1 in group no. 4; Rick might be [0002,0004], etc.

If Mike, Grace, and Larry are involved in accounts receivable, you may wish to assign them to a different group so Brad and the members of his group cannot have access to the files with which Mike and his group work.

In other words, system privileges are the major criteria for determining how to group users. See the chapter on System Security and User Accounts, Part 2 in the _WMCS System Manager's Reference Manual_ for a more detailed description of the UIC.

The TIMESLICE field allows you to specify the default timeslice the user's processes have. This represents the maximum amount of CPU processing time allocated to the user process during each time-sharing turn, e.g., the 60 in the foregoing example indicates the timeslice assigned to the user process is 60 milliseconds. This is the _maximum_ amount of time the process gets each time it is scheduled. Therefore, if the process is blocked before completion of its turn, it cannot use the remainder of the timeslice.

To edit this line, use the arrow keys to position the cursor and then type the values you wish. What you type in the UIC field replaces the characters already in that field.

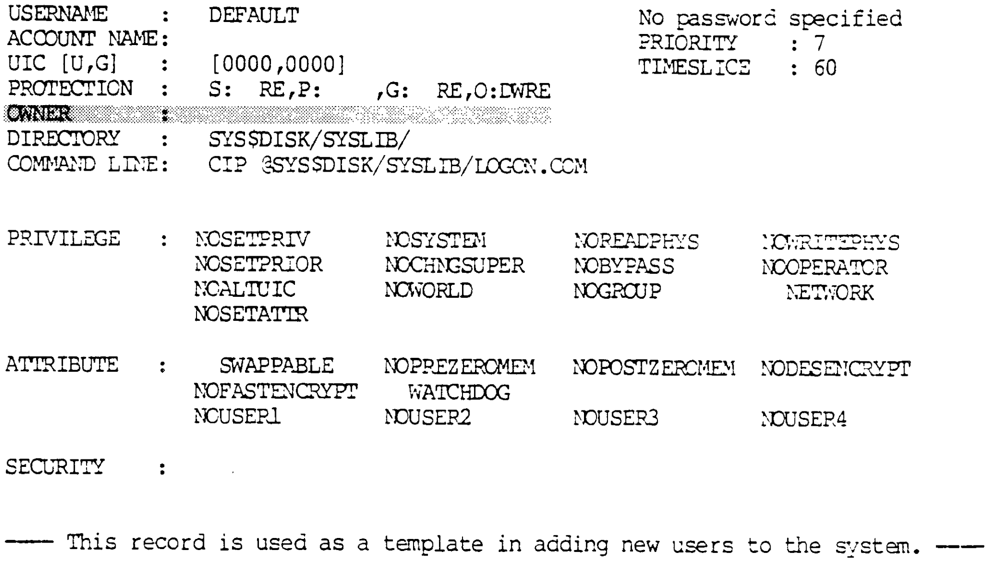


```
USERNAME     :   DEFAULT                                   No password specified
ACCOUNT NAME:                                              PRIORITY     : 7
UIC [U,G]    :   [0000,0000]                               TIMESLICE    : 60
PROTECTION   :   S:  RE,P:     ,G:  RE,O:DWRE
OWNER        :
DIRECTORY    :   SYS$DISK/SYSLIB/
COMMAND LINE:    CIP @SYS$DISK/SYSLIB/LOGON.COM


PRIVILEGE    :  NOSETPRIV       NOSYSTEM       NOREADPHYS      NOWRITEPHYS
                NOSETPRIOR      NOCHNGSUPER    NOBYPASS        NOOPERATOR
                NOALTUIC        NOWORLD        NOGROUP           NETWORK
                NOSETATTR

ATTRIBUTE    :    SWAPPABLE     NOPREZEROMEM   NOPOSTZEROMEM   NODESENCRYPT
                NOFASTENCRYPT     WATCHDOG
                NOUSER1         NOUSER2        NOUSER3         NOUSER4

SECURITY     :


---- This record is used as a template in adding new users to the system. ----
```

Protection Field

This line allows you to specify the default protection for this user account. See the chapter on System Security and File Protection in the WMCS System Manager's Reference Manual for a complete description of the default protection mask.

To edit this field use the arrow keys to position the cursor. Strike the spacebar to change the values in the fields. If the letter is displayed, that permission is given; otherwise it is not.

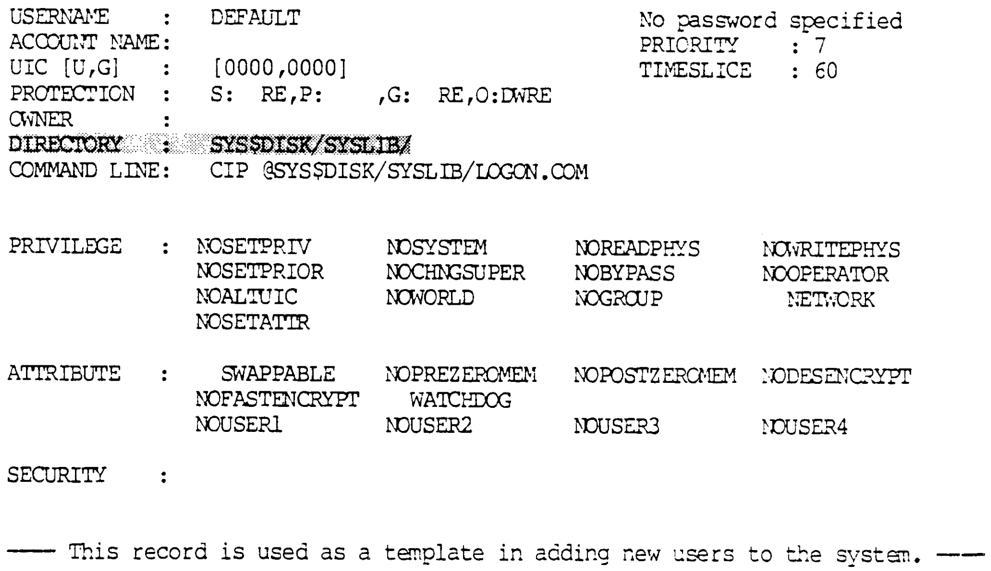


```
USERNAME     :   DEFAULT                              No password specified
ACCOUNT NAME:                                         PRIORITY     : 7
UIC [U,G]    :   [0000,0000]                          TIMESLICE    : 60
PROTECTION   :   S:  RE,P:     ,G:  RE,O:DWRE
OWNER        :
DIRECTORY    :   SYS$DISK/SYSLIB/
COMMAND LINE:    CIP @SYS$DISK/SYSLIB/LOGON.COM


PRIVILEGE    :  NOSETPRIV     NOSYSTEM      NOREADPHYS     NOWRITEPHYS
                NOSETPRIOR    NOCHNGSUPER   NOBYPASS       NOOPERATOR
                NOALTUIC      NOWORLD       NOGROUP          NETWORK
                NOSETATTR

ATTRIBUTE    :    SWAPPABLE   NOPREZEROMEM  NOPOSTZEROMEM  NODESENCRYPT
                NOFASTENCRYPT   WATCHDOG
                NOUSER1       NOUSER2       NOUSER3        NOUSER4

SECURITY     :

---- This record is used as a template in adding new users to the system. ----
```

Owner Field

This line tells you the name of the person who owns the user account. What you type on this line is for your reference, i.e., whereas portions of the UAR are read by such programs as LOGON, PSTAT, etc., this field is not.

To edit this field, use the arrow keys to position the cursor in the field and type the owner's name. You are limited to 64 characters. One owner can have several UARs, i.e., several usernames.



```
USERNAME    :   DEFAULT                              No password specified
ACCOUNT NAME:                                        PRIORITY    : 7
UIC [U,G]   :   [0000,0000]                          TIMESLICE   : 60
PROTECTION  :   S:  RE,P:     ,G:  RE,O:DWRE
OWNER       :
DIRECTORY   :   SYS$DISK/SYSLIB/
COMMAND LINE:   CIP @SYS$DISK/SYSLIB/LOGON.COM

PRIVILEGE   :  NOSETPRIV      NOSYSTEM       NOREADPHYS     NOWRITEPHYS
               NOSETPRIOR     NOCHNGSUPER    NOBYPASS       NOOPERATOR
               NOALTUIC       NOWORLD        NOGROUP          NETWORK
               NOSETATTR

ATTRIBUTE   :    SWAPPABLE    NOPREZEROMEM   NOPOSTZEROMEM  NODESENCRYPT
               NOFASTENCRYPT    WATCHDOG
               NOUSER1        NOUSER2        NOUSER3        NOUSER4

SECURITY    :

---- This record is used as a template in adding new users to the system. ----
```

Directory Field

Use this line to specify the user-account default directory, i.e., the directory the user is in after logging on to the system.

For example, if you create /USERS/ as a subdirectory of /ROOTDIR/, and want to have all your user-account default directories in /USERS/, you would use this line to specify the following kind of pathname:

SYS$DISK/USERS.AL/

This tells the LOGON Program that when the user who belongs to this account logs on, /USERS.AL/ is to be his default directory.

If you put a nodename at the start of the default directory pathname, the user's default directory will be on a remote node when he first logs on.

Use the arrow keys to move the cursor to this field. Then type the pathname to the user-account default directory. This field is limited to 64 characters.

```
USERNAME     :   DEFAULT                            No password specified
ACCOUNT NAME:                                       PRIORITY    : 7
UIC [U,G]    :   [0000,0000]                        TIMESLICE   : 60
PROTECTION   :   S:  RE,P:     ,G:  RE,O:DWRE
OWNER        :
DIRECTORY    :   SYS$DISK/SYSLIB/
COMMAND LINE:    CIP @SYS$DISK/SYSLIB/LOGON.COM

PRIVILEGE    :  NOSETPRIV      NOSYSTEM       NOREADPHYS      NOWRITEPHYS
                NOSETPRIOR     NOCHNGSUPER    NOBYPASS        NOOPERATOR
                NOALTUIC       NOWORLD        NOGROUP           NETWORK
                NOSETATTR

ATTRIBUTE    :    SWAPPABLE    NOPREZEROMEM   NOPOSTZEROMEM   NODESENCRYPT
                NOFASTENCRYPT    WATCHDOG
                NOUSER1        NOUSER2        NOUSER3         NOUSER4

SECURITY     :

---- This record is used as a template in adding new users to the system. ---
```

Command Line Field

Use this line to specify the name of the file containing the interface with the WMCS that is executed as the user process, i.e., that becomes the user process when the user logs on. Even though the CIP is usually the interface you specify, not all users may need the CIP. Some may need to have only canned application programs, or special-purpose command interpreters.

LOGON.COM is a command file in /SYSLIB/ (on the system disk) that contains logical name assignments, etc., for all the users on your system. The command file is usually passed as a parameter to the CIP.

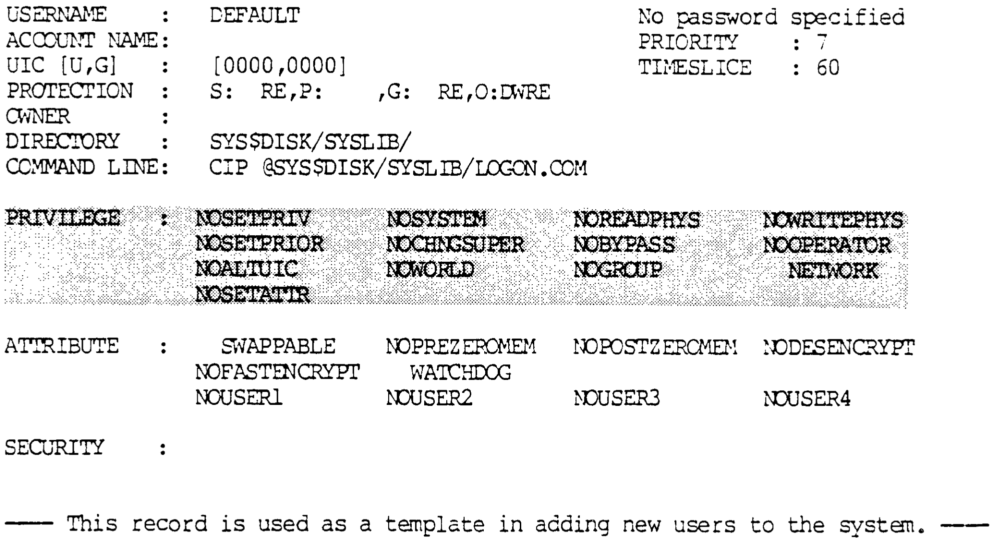
The last line in LOGON.COM is @USERUP. This executes the USERUP.COM file in the user-account default directory. In other words, even though no pathname to that directory is given, this executes the USERUP.COM file in the user-account default directory.

If you wish to have different LOGON.COM files (with different filenames) for the various groups of users on your system, type the filename for the appropriate LOGON file on this line in the UAR. This field scrolls horizontally.

If you put a nodename at the start of the command line, LOGON will execute the process on the specified node rather than on the current node. For example, the following entry in the COMMAND LINE field will cause CIP to be executed by LOGON on node "NODEA":

```
COMMAND LINE: {NODEA}CIP @SYS$DISK/SYSLIB/LOGON.COM
```

If you do not want any command file executed when the user logs on to the system, type only CIP or another program name on this line.



```
USERNAME     :   DEFAULT                              No password specified
ACCOUNT NAME:                                         PRIORITY     : 7
UIC [U,G]    :   [0000,0000]                          TIMESLICE    : 60
PROTECTION   :   S:  RE,P:    ,G:  RE,O:DWRE
OWNER        :
DIRECTORY    :   SYS$DISK/SYSLIB/
COMMAND LINE:    CIP @SYS$DISK/SYSLIB/LOGON.COM

PRIVILEGE    :  NOSETPRIV      NOSYSTEM      NOREADPHYS     NOWRITEPHYS
                NOSETPRIOR     NOCHNGSUPER   NOBYPASS       NOOPERATOR
                NOALTUIC       NOWORLD       NOGROUP          NETWORK
                NOSETATTR

ATTRIBUTE    :    SWAPPABLE    NOPREZEROMEM  NOPOSTZEROMEM  NODESENCRYPT
                NOFASTENCRYPT    WATCHDOG
                NOUSER1        NOUSER2       NOUSER3        NOUSER4

SECURITY     :

---- This record is used as a template in adding new users to the system. ----
```

Privilege Field

These lines contain the privileges of the user.

Whenever a user logs on to the system, or initiates a process, that process has the privileges assigned in the user's UAR. The privileges assigned to a process determine the extent to which that process can affect other processes or system hardware, and the degree of access that process has to files created by other processes. (Process privileges are defined in the chapter on System Security and User Accounts, Part 2, in the WMCS System Manager's Reference Manual.

If the mnemonic name appears in the UAR, the user has that privilege. If a "NO" appears next to the privilege name, the user does not have that privilege.

SETPRIV   Set Privilege

If this privilege appears in the UAR, the process is able to set any other privilege for its own or any other process that it can affect.

SYSTEM   System Access

This gives the process system-access to files, i.e., system user is one of four classes of users.

READPHYS Read Physical

The process can read physical blocks on the disk, or other media, and copy system tables into its own process memory.

WRITEPHYS Write Physical

This allows the process to write physical blocks on the disk, or other media, format disks, and update the system's internal tables.

SETPRIOR Set Priority

The process can change its process priority, timeslice, or become a realtime process.

CHNGSUPER Supervisor

This allows the process to change its run mode to supervisor mode. Once the process is in supervisor mode, it has unrestricted access to the system.

BYPASS Protection Bypass

The process can inspect, modify, and delete any file on the system.

OPERATOR Operator

This allows the process to set the system clock, format disks, and set the scheduling priority ratio.

ALTUIC Alternate UIC

The process can access a file as though the process has the same UIC as the owner of the file.

WORLD World Access

This allows the process to affect processes outside its group.

GROUP Group Access

The process can affect any process whose group ID matches the group ID of its own process.

NETWORK Network Access

The process is allowed to perform remote operations.

SETATTR Set Attributes

The process can change its attributes.

Strike the spacebar with the cursor next to each privilege name to toggle the values. If a "NO" appears next to the privilege name, it disappears when toggled. If only the privilege name appears, a "NO" is inserted and that privilege is removed when you strike the spacebar.

If you do not wish to toggle a field, strike [RETRN] and the cursor will move to the next privilege name without modifying the current name.

```
USERNAME     :   DEFAULT                                    No password specified
ACCOUNT NAME:                                               PRIORITY     : 7
UIC [U,G]    :   [0000,0000]                                TIMESLICE    : 60
PROTECTION   :   S:  RE,P:     ,G:  RE,O:DWRE
OWNER        :
DIRECTORY    :   SYS$DISK/SYSLIB/
COMMAND LINE:    CIP @SYS$DISK/SYSLIB/LOGON.COM


PRIVILEGE    :  NOSETPRIV      NOSYSTEM       NOREADPHYS     NOWRITEPHYS
                NOSETPRIOR     NOCHNGSUPER    NOBYPASS       NOOPERATOR
                NOALTUIC       NOWORLD        NOGROUP          NETWORK
                NOSETATTR

ATTRIBUTE    :    SWAPPABLE    NOPREZEROMEM   NOPOSTZEROMEM  NODESENCRYPT
                NOFASTENCRYPT    WATCHDOG
                NOUSER1        NOUSER2        NOUSER3        NOUSER4

SECURITY     :


---- This record is used as a template in adding new users to the system. ----
```

Attribute Field

These lines contain the attributes of the user.

Whenever a user logs on to the system, or initiates a process, that process has the attributes assigned in the user's UAR. The attributes assigned to a process determine whether various unrelated things may happen to a process before or after it executes.

If the attribute name appears in the UAR, the user has that attribute. If a "NO" appears next to the attribute name, the user does not have that attribute.

SWAPPABLE

If this attribute appears in the UAR, the process may be swapped out by the swapper process.

PREZEROMEM

If this attribute appears in the UAR, the WMCS will be instructed to zero each page of memory before allocating it to the process.

POSTZEROMEM

If this attribute appears in the UAR, the WMCS will be instructed to zero each page of memory the process had allocated after the process terminates.

DESENCRYPT

If this attribute appears in the UAR, if the process ever sends data to another node on the network, the data will be encrypted using the DES encryption algorithm.

FASTENCRYPT

If this attribute appears in the UAR, if the process ever sends data to another node on the network, the data will be encrypted using a fast encryption algorithm. If both DESENCRYPT and FASTENCRYPT appear in the UAR, DESENCRYPT will be performed.

WATCHDOG

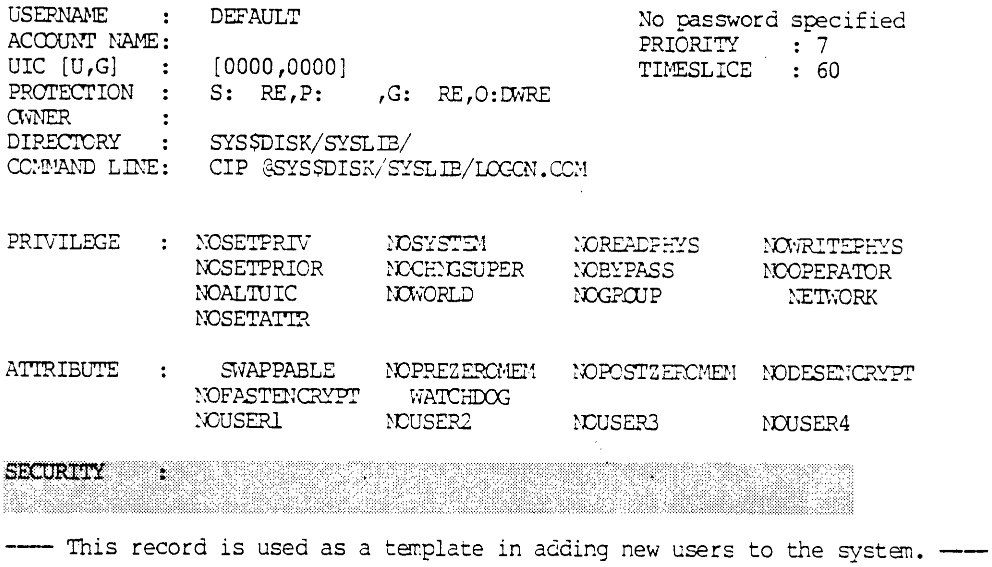
If this attribute appears in the UAR, the process may be killed by the watchdog process after a period of inactivity.

USER1, USER2, USER3, USER4

These attribute bits are reserved for user definition and implementation.

Strike the spacebar with the cursor next to each attribute name to toggle the values. If a "NO" appears next to the attribute name, it disappears when toggled. If only the attribute name appears, a "NO" is inserted and that attribute is removed when you strike the spacebar.

If you do not wish to modify a particular attribute, use the arrow keys to move the cursor to another field, or strike [RETRN] to move to the next attribute field without modifying the current attribute field.



```
USERNAME     :   DEFAULT                                    No password specified
ACCOUNT NAME:                                               PRIORITY      : 7
UIC [U,G]    :   [0000,0000]                                TIMESLICE     : 60
PROTECTION   :   S:  RE,P:     ,G:  RE,O:DWRE
OWNER        :
DIRECTORY    :   SYS$DISK/SYSLIB/
COMMAND LINE:    CIP @SYS$DISK/SYSLIB/LOGON.COM

PRIVILEGE    :  NOSETPRIV      NOSYSTEM       NOREADPHYS      NOWRITEPHYS
                NOSETPRIOR     NOCHNGSUPER    NOBYPASS        NOOPERATOR
                NOALTUIC       NOWORLD        NOGROUP           NETWORK
                NOSETATTR

ATTRIBUTE    :    SWAPPABLE    NOPREZEROMEM   NOPOSTZEROMEM   NODESENCRYPT
                NOFASTENCRYPT    WATCHDOG
                NOUSER1        NOUSER2        NOUSER3         NOUSER4

SECURITY     :

---- This record is used as a template in adding new users to the system. ----
```

Security Field

This field allows you to specify a program to be executed in place of the standard password. If a filename appears here, this program is executed after LOGON reads the username. If the abort status and condition code returned from this program are zero, LOGON proceeds. Otherwise, the user is denied access to the system. This security program replaces the standard password checking.

This field should be used when very rigorous security is needed before allowing users on the system. To specify the result returned to LOGON, see the _EXPROC system call in the WMCS Programmer's Reference Manual.

Functions available in USERPROF

Each of the following functions is executed by first striking [ESC][ESC], then typing the mnemonic on the function line that appears at the bottom of the screen. The position of the cursor when you strike [ESC][ESC] does not affect the execution of any of these functions.

ab

Abort this session of USERPROF.

FUNCTIONAL DESCRIPTION

AB exits this editing session of USERPROF without writing any of the changes made during the session to the UAF.DAT file.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

ex	Exit USERPROF

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **ab** on the USERPROF function line. If no modifications have been made, the screen is cleared, and you return to your CIP prompt. If any modifications have been made to the UAF.DAT file, this prompt appears on the function line:

```
UAF.DAT file has been modified - Abort?
```

Step 3 Type y or n. If you type some other character, the following message briefly appears on your screen:

```
Please answer 'Y' or 'N'.
```

If you type Y, the screen is cleared and you return to your CIP prompt. If you type N, the cursor returns to the position it had on the screen before executing AB.

ad

Add a user account.

FUNCTIONAL DESCRIPTION

AD creates a copy of the DEFAULT UAR, assigns the copy to the username you specify, and alphabetizes the records in UAF.DAT so every username (and the record assigned to it) appears in alphabetical order.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

dl	Delete a UAR

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type ad on the USERPROF function line. This prompt appears on the function line:

```
New Username >
```

Step 3 Type the username to be assigned to the account. The username you type must be unique to UAF.DAT. You are limited to nine characters. Blanks within the username are not allowed.

Step 4 Strike [RETRN]. Following validation, the new UAR appears on the screen with the specified username at the top of the UAR. All other values in the new UAR match the DEFAULT template.

If the username already exists, a message appears on the function line:

```
This username already exists. Please enter another name.
```

cd

Change the DEFAULT record.

FUNCTIONAL DESCRIPTION

CD allows fields on the DEFAULT template page to be modified. After this command has been executed, changes may be made in the normal manner to the DEFAULT record, until another record is displayed. If the DEFAULT record is not the current record, it will be displayed and become the current record.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1	Strike [ESC][ESC].
Step 2	Type **cd** on the USERPROF function line. If the DEFAULT record is not already displayed on the screen, it will now be displayed.
Step 3	Make the desired changes in the DEFAULT record.

cn

Change the username on an account.

FUNCTIONAL DESCRIPTION

CN changes the username assigned to the UAR on the screen. The new username must be unique to UAF.DAT. You cannot change these usernames: DEFAULT, SYSTEM.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **cn** on the USERPROF function line. This prompt appears on the screen:

```
Change Username to>
```

Step 3 Type the new username. You are limited to nine characters. Blanks within the username are ignored.

Step 4 Strike [RETRN]. If the username is validated, the USERNAME field on the screen is updated to display the new name. If the username is not validated, this message is displayed on the function line:

```
This username already exists. Please enter another name.
```

cp

Spawn a CIP while remaining in USERPROF.

FUNCTIONAL DESCRIPTION

CP spawns a CIP from within USERPROF.

CORRESPONDING CONTROL-KEY FUNCTIONS

[ESC] c

RELATED ESCAPE-KEY FUNCTIONS

ex	Exit USERPROF

EXECUTION

Step 1	Strike [ESC][ESC].
Step 2	Type **cp** on the USERPROF function line. The CIP prompt appears. When you log off from the CIP, you are returned to the same UAR you were editing before spawning the CIP.

dl

Delete a UAR from UAF.DAT.

FUNCTIONAL DESCRIPTION

DL deletes one record from UAF.DAT. The target UAR must be on the screen when you execute DL.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

ad	Add a user account

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **dl** on the function line. This prompt appears on the function line:

```
Delete record ?
```

If you try to delete the DEFAULT or SYSTEM UARs, this diagnostic message appears on the function line:

```
The DEFAULT (SYSTEM) record cannot be deleted.
```

Step 3 Type **n** if you do not want to delete the UAR, or type **y** if you wish to delete it.

ex

Exit UAF.DAT, i.e., terminate the editing session.

FUNCTIONAL DESCRIPTION

EX terminates the editing session and returns you to the CIP.

CORRESPONDING CONTROL-KEY FUNCTIONS

{-}

RELATED ESCAPE-KEY FUNCTIONS

ab	Abort USERPROF
cp	Create a CIP

EXECUTION

Step 1 | Strike[ESC][ESC].

Step 2 | Type **ex.** This report appears at the bottom of the screen:

```
     Exiting
```

The screen is cleared and the cursor reappears next to your CIP prompt.

he

Call to the screen a display of commands available in USERPROF.

FUNCTIONAL DESCRIPTION

HE displays the list of commands available in USERPROF.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **he**. The first page of a three-screen help display appears on your screen. The first screenful contains a list of the escape-key functions available.

The second screenful lists functions for deletion of the field and how to move around the screen.

The third screenful describes the different field types on the screen display of a UAR.

Step 3 Press [RETRN] to cycle through the pages, then back to your current record. Press [CTRL] c at any time to return to your current record without showing the remaining help screens.

mp

Move toward the beginning or end of UAF.DAT.

FUNCTIONAL DESCRIPTION

MP calls to the screen the UAR immediately following the UAR on the screen. This message appears on the screen when the last UAR in UAF.DAT is already on the screen:

    There Is No Record Following This Record

-MP calls to the screen the UAR immediately preceding the UAR on the screen. This message appears on the screen if the DEFAULT UAR is on the screen when you execute -MP:

    No prior record found.

To move more than one page at a time type a number representing the number of pages to be moved before typing MP. For example, entering the command -5MP is the same as using -MP five times.

CORRESPONDING CONTROL-KEY FUNCTIONS

[CTRL] p	(for MP)
[CTRL] h	(for -MP)

RELATED ESCAPE-KEY FUNCTIONS

sh	Show a user account

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **mp**. This report appears momentarily on the screen:

    Paging to next record.

The next UAR appears on the screen, and the cursor appears over the first character in the ACCOUNT NAME field of that UAR.

pr

Create a text-file copy of a record in the UAF.DAT.

FUNCTIONAL DESCRIPTION

PR creates a file named UAF.PRN and assigns it to the default directory. UAF.PRN contains a copy of one or all of the UAR's in UAF.DAT.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 | Strike [ESC][ESC].

Step 2 | Type **pr** on the function line. This prompt appears on the function line:

```
Printing UAF.DAT records...
```

When the record(s) are copied to the UAF.PRN file, the cursor returns to the position it had prior to the PR command.

rp

Repeat last function.

FUNCTIONAL DESCRIPTION

RP repeats the last function you executed.

CORRESPONDING CONTROL-KEY FUNCTIONS

[CTRL] z

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 | Strike [ESC][ESC]

Step 2 | Type **rp** on the USERPROF function line. The last function is repeated.

rs

Refresh the screen.

FUNCTIONAL DESCRIPTION

RS restores the display on the screen when a message or another interruption disrupts the display.

CORRESPONDING CONTROL-KEY FUNCTIONS

[CTRL] -

RELATED ESCAPE-KEY FUNCTIONS

ud	Undo the modifications to a UAR

EXECUTION

Step 1	Strike [ESC][ESC].
Step 2	Type **rs**. The screen is erased and redisplayed so the UAR reappears on the screen as it was before the interruption.

sa

Save the modifications made so far in the UAF.DAT file.

FUNCTIONAL DESCRIPTION

SA writes out all modified user records to the UAF.DAT file without exiting the editing session. This is useful when you have a large number of modifications and want to protect them from system failure.

CORRESPONDING CONTROL-KEY FUNCTIONS

{,}

RELATED ESCAPE-KEY FUNCTIONS

ex	Exit USERPROF
ab	Abort the editing session

EXECUTION

Step 1 | Strike [ESC][ESC].

Step 2 | Type **sa** on the USERPROF function line. This prompt appears on the function line:

```
Saving modifications...
```

After writing all the modified records to the UAF.DAT file, the cursor returns to the position it had prior to the SA command.

sf

Show the first UAR in the UAF.DAT file.

FUNCTIONAL DESCRIPTION

SF displays the first record in the UAF.DAT file. This record is always the DEFAULT record. If you then page forward using the MP command, the records will be displayed in alphabetical order.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

sh	Show a specific UAR
sl	Show the last UAR in the UAF.DAT file

EXECUTION

Step 1 | Strike [ESC][ESC].

Step 2 | Type **sf** on the USERPROF function line. This prompt appears briefly on the function line:

```
Showing the first record...
```

The DEFAULT record is displayed on the screen.

sh

Show a specific UAR record.

FUNCTIONAL DESCRIPTION

SH allows you to specify the UAR you want to examine or modify.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

mp	Move page
sf	Show the first UAR in the UAF.DAT file
sl	Show the last UAR in the UAF.DAT file

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **sh**. This prompt appears at the bottom of the screen:

```
Show Username >
```

Step 3 Type the username assigned to the UAR you want to examine. The case in which you type the username is ignored.

Step 4 Strike [RETRN]. If the record exists, it is displayed on the screen. If the specified username does not exist, this diagnostic message appears on the screen:

```
This username does not exist. Please enter another name.
```

sl

Show the last UAR in the UAF.DAT file.

FUNCTIONAL DESCRIPTION

SL displays the last record in the UAF.DAT file. If you then page backward using the -MP command, the records will be displayed in reverse alphabetical order.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

sf	Show the first UAR in the UAF.DAT file
sh	Show a specific UAR

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **sl** on the USERPROF function line. This prompt appears briefly on the function line:

```
Showing the last record...
```

The last record is displayed on the screen.

sp

Assign a password to an account, or change the password already assigned to an account.

FUNCTIONAL DESCRIPTION

SP sets the password of the UAR on the screen.

CORRESPONDING CONTROL-KEY FUNCTIONS

None.

RELATED ESCAPE-KEY FUNCTIONS

cn	Change a username

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **sp** on the USERPROF function line. This prompt appears on the function line:

```
Password >
```

Step 3 Type in the new password. (If you wish to indicate no password is to be assigned to this user, strike [RETRN] after typing SP.)

Step 4 Strike [RETRN]. This report appears on the function line:

```
Verify password >
```

Step 5 Type the password again.

Step 6 Strike [RETRN]. Once the password is assigned or reassigned, this statement appears in the upper right-hand corner of the UAR:

```
Password specified.
```

If you do not enter any valid characters in step 3, this statement appears in the upper right-hand corner of the UAR:

```
No password specified.
```

sr

Search for a specific UAR.

FUNCTIONAL DESCRIPTION

SR allows you to search for a specific UAR you want to examine or modify. This differs from SH only in that if the UAR you specify does not exist, no message is displayed on the bottom of the screen but a record in the UAF.DAT file is displayed on the screen after the search completes unsuccessfully.

CORRESPONDING CONTROL-KEY FUNCTIONS

{7}

RELATED ESCAPE-KEY FUNCTIONS

sh	Show a specific UAR

EXECUTION

Step 1 Strike [ESC][ESC].

Step 2 Type **sr**. This prompt appears on the function line:

```
Search for Username>
```

Step 3 Type the username assigned to the UAR you want to search for. If it does not exist, the record with a username next in alphabetical order after the specified username is displayed. If the specified username is last alphabetically, then the last UAR is displayed on the screen.

ud

Undo changes made on the UAR.

FUNCTIONAL DESCRIPTION

UD undoes changes to cursor-editable fields of the UAR, i.e. fields that can be edited without using the [ESC][ESC] key.

However, you must perform UD immediately after the change to be corrected. Otherwise, UD is ineffectual. For example, you can use UD to change anything you just typed (or erased) in a field, but you cannot use it to undo a function like SP (RS is the only function after which UD can be executed successfully).

CORRESPONDING CONTROL-KEY FUNCTIONS

{PF1}

RELATED ESCAPE-KEY FUNCTIONS

None.

EXECUTION

Step 1 | Strike [ESC][ESC].

Step 2 | Type **ud**. This report appears momentarily on the screen:

```
Undoing changes...
```

The UAR reappears as it did immediately following the last executed escape-key function.

Cursor Movement and Control-Key Functions

The cursor appears only in UAR fields that can be edited without using a USERPROF function.

If the cursor is in the last character position on a line, any character key, the right-arrow key, or spacebar moves the cursor to the beginning of the next cursor-editable field.

If the cursor is in the first editable position in a field, the left-arrow key takes the cursor to the last character position in the previous cursor-editable field.

If the cursor is in the last character position in the last cursor-editable field of the UAR, either the spacebar or the right-arrow key moves the cursor to the first character position in the top field of the UAR.

If the cursor is in the first character position in the top field, the left-arrow key or the backspace key moves the cursor to the last character position in the bottom field.

The up-arrow and down-arrow keys move the cursor from one line to another within the UAR. If the cursor is in the top field when you strike the up-arrow key, the cursor jumps to the bottom field, and vice versa.

Use the following commands to move the cursor:

[ESC] key	[CTRL] key	Operation
el	[CTRL] g	Move to end of line
fl	[CTRL] a	Move to front of line
mc	right arrow	Move right one character
-mc	left arrow	Move left one character
me	[CTRL] e	Move to last field on screen
mf	[RETRN]	Move to next field
-mf		Move to previous field
ml	[CTRL] o	Move to front of next line
-ml	[CTRL] w	Move to front of previous line
mt	[CTRL] t	Move to first field on screen
mw	[CTRL] f	Move to next word (usually next field)
-mw	[CTRL] r	Move to previous word (usually previous field)
pl	down arrow	Move to field immediately below
-pl	up arrow	Move to field immediately above

Use the following commands to make deletions:

[ESC] key	[CTRL] key	Operation
db	[CTRL] u	Delete to beginning of field
dc	[CTRL] v	Delete character in cursor
-dc	[DEL]	Delete character to left of cursor
de	[CTRL] d	Delete to end of field
df	[CTRL] y	Delete entire field
dw	[CTRL] k	Delete next word
-dw	[CTRL] b	Delete previous word

NOTE: All [CTRL]-key commands are dependent on the correct setup file for your terminal (see the WMCS VEW User's Reference Manual). The commands described here are for the T-7000 and MG8000 terminals.

Miscellanea

You can use the COPY command to create a UAF:DAT on another device. Nevertheless, that UAF.DAT must be copied to a /SYSLIB/directory and the version must be 1.

There can be as many UARs in UAF.DAT as the system manager deems necessary. There can also be as many versions of UAF.DAT as the system manager desires, but the WMCS uses only version 1 of UAF.DAT in /SYSLIB/.

Two UARs are built-in to UAF.DAT:

DEFAULT This is always the first UAR in the file, and appears on the screen whenever USERPROF is executed.

SYSTEM This UAR is reserved for the system manager.

Related CIP Commands

logon	Log a user on to a system

usscopy

Functional Description

Use this command to transfer files from one WMCS system to another across serial communication lines.

Command Line Syntax

Mnemonic	usscopy	
Required parameters	Serial port	
	Source	(Required if function is :SEND or :RECEIVE, otherwise not allowed.)
	Destination	(Required if function is :SEND or :RECEIVE, otherwise not allowed.)
Optional parameter	Function (:BACKGROUND, :RECEIVE, :SEND)	

Switches

File selection	:before=	:mod	:confirm	:since=
	:exclude=	:uic=		
Destination specification	:builddir	:concat	:edit=	
Destination attributes	:preserve=	:protection		
Transmission	:ascii	:retries=	:bufsize=	:timeout=
	:hostsync=	:baudrate=		
Other	:log	:stats		

Parameters

Serial Port

Function: Required. Use this parameter to specify the name of the port for the communication channel. Note that the port must already be mounted.

Default: None.

Syntax: Type the devicename of a serial communication port.

Source

Function: Required. Use this parameter to specify the names of the files on the local system to be sent to the remote system, or the names of the files on the remote system to be received on the local system.

Default: None.

Syntax: Type a list of filenames separated by commas. Wildcard symbols are allowed. (Do not use this parameter with :BACKGROUND.)

Destination

Function: Required. Use this parameter to specify the names of the files to which the source files are copied. Field substitution occurs in this parameter. That is, everywhere an asterisk appears in the value typed for this parameter, the corresponding field from the corresponding source file is substituted.

Default: None.

Syntax: Type a destination file designation.

Function

Function: Optional. Use this parameter to specify the action to be performed as part of the transfer of the file(s).

Default: :background

Syntax: Type one of the following three values:

:BACKGROUND This action is performed on the remote system. The USSCOPY with this value specified receives direction from the USSCOPY on the local system.

:RECEIVE This action requests that files on the remote system be received by the local system, i.e. sent from the remote system to the local system.

:SEND This action requests that files on the local system be sent to the remote system.

Switches

:ascii

Function Use this switch to specify that all data transmitted is to be converted to hex ASCII, transmitted, and then converted back to binary. This option need only be used if the communication port is restricted to 7-bit data. If the port allows 8-bit communications, usscopy allows transfers of binary files successfully without conversion to hex ASCII. Note that the transmission time is greatly reduced if data are not translated to hex ASCII.

Default :noascii

Syntax Type **:ascii**

:baudrate=

Function Use this switch on TTY class devices to set the baud rate of the given device. This alters the baud rate on both systems.

Default Driver dependent unless changed by previous DSTAT command.

Syntax Type **:baudrate=** followed by one of the following: 50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, or 19200.

:before=

Function Use this switch to select only those files that match the file list parameter and were created or modified before the specified date and time.

Default Selects all files that match the file list parameter.

Syntax Type **:before=** followed by a date and/or time in the standard date and time syntax.

:bufsize=

Function Use this switch to specify the maximum number of characters that can be transmitted in each data packet.
Default :bufsize=512
Syntax Type **:bufsize=** followed by a numeral between 128 and 1024.

:builddir

Function Use this switch to create destination directories automatically if they do not exist.
Default :nobuilddir
Syntax Type **:builddir**

:concat

Function Use this switch to cause all of the source files to be concatenated into a single destination file.
Default :noconcat
Syntax Type **:concat**

:confirm

Function Use this switch to confirm or deny the desired operation on each file or device.
Default :noconfirm
Syntax Type **:confirm**

:edit=

Function Use this switch to specify a series of edits to be performed on the destination file designation. The edits are specified as a series of "find and switch" strings. The destination file designation is searched for the first occurrence of the "find" string. If it is found, it is replaced by the "switch" string.
Default No edits are performed.
Syntax Type **:edit=** followed by a list of "find and switch" strings separated by commas. The find string is separated from the switch string by a colon. Either one or both of the strings may be empty.

:exclude=

Function Use this switch to exclude files or devices from the File List parameter.
Default All files or devices that match the File List parameter are selected.
Syntax Type **:exclude=** followed by a list of file or device designations separated by commas. Wildcard characters are allowed.

:hostsync= Function Use this switch on TTY class devices to specify the protocol to be used when the input buffer for the particular port becomes almost full. This switch is used to regulate the rate of transmission of characters.

Default Software host synchronization, i.e. when the buffer becomes almost full, the driver automatically sends a [CTRL] s for each character that is received. When the buffer becomes half full, a [CTRL] q is sent to signal that transmission may resume.

Syntax Type **:hostsync=** followed by one of the keywords **HARDWARE** or **SOFTWARE.**

If HARDWARE is specified, when the buffer becomes almost full the REQUEST TO SEND line is raised. When the buffer becomes half full, the REQUEST TO SEND line is lowered. The receiver's REQUEST TO SEND line must be tied to the transmitter's CLEAR TO SEND line.

:log Function Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

Default :log

Syntax Type **:nolog**

:mod Function Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.

Default :nomod

Syntax Type **:mod**

:preserve= Function Use this switch to preserve certain attributes of the original file when a new copy of it is created. The preservable aspects are: creation date, protection.

Default The original file's attributes are not preserved in the new file, i.e., the new file is created with the current date and time, the owner is the current user, and the protection is the user's default protection mask.

	Syntax	Type **:preserve=** followed by any combination of the keywords **CREATED, PROTECTION,** or **ALL** separated by commas.
:protection=	Function	Use this switch to specify the protection mask to be assigned to the file or files being created.
	Default	The user's default protection mask.
	Syntax	Type **:protection=** followed by a protection mask. Note that only those fields of the protection mask that are specified are altered. Fields not specified keep their original value.
:retries=	Function	Use this switch to specify the number of times USSCOPY should try to send a packet before declaring a transmission error.
	Default	:retries=5
	Syntax	Type **:retries=** followed by a number between 1 and 32.
:since=	Function	Use this switch to select only those files that match the file list parameter and were created or modified since the specified date and time.
	Default	Selects all files that match the file list parameter.
	Syntax	Type **:since=** followed by a date and/or time in the standard syntax.
:stats	Function	Use this switch to request that various statistics be displayed.
	Default	:nostats
	Syntax	Type **:stats**
:timeout=	Function	Use this switch to specify the maximum amount of time (in seconds) to wait for a character.
	Default	:timeout=2
	Syntax	Type **:timeout=** followed by a numeral between 1 and 10.

:uic=	Function	Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.
	Default	Selects all files that match the specified list.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.

Examples

The following procedure for executing USSCOPY assumes that you have a serial cable connected to serial port _TT1 on the local system; that the cable is connected to serial port _TT15 on the remote system; that such port characteristics as baud rate, etc., are the same for both ports (use the DSTAT command to achieve this); and that you want to receive a file from the remote system to the system on which you are working.

Step 1 Type the following on the CIP command line and strike [RETRN]:

```
talkt _tt1
```

The cursor jumps to the left margin of the line below the CIP command line. You are now conversing with the remote system through _TT1 on your system to _TT15 on the remote system.

Step 2 Strike [RETRN], and log on to the remote system.

Step 3 Type the following on the command line of the remote system and strike [RETRN].

```
usscopy _tt15
```

Since no action is specified, :BACKGROUND is assumed. This command line character string initiates the background copy of USSCOPY on the remote system. The background version of USSCOPY watches for instructions to be received on _TT15.

Step 4 Type the following:

```
[CTRL] z [CTRL] e
```

This sequence of characters breaks the connection between your terminal and _TT1. Characters you type after this point are not transmitted to the remote system. When you strike

[CTRL] e, the cursor reappears next to the CIP prompt and you are operating again on your home system.

Step 5

Type the following kind of character string on the CIP command line and strike [RETRN]:

```
usscopy _tt1 :receive _ds0/language/report.doc *
```

This character string indicates you want to copy the file named _DS0/LANGUAGE/REPORT.DOC from the remote system (:RECEIVE) and assign it to your default directory. Note that the source file designation (_DS0/LANGUAGE/REPORT.DOC) corresponds to the names of devices and directories on the remote system. The destination file designation (*) corresponds to the name the file receives on the local system.

The following kind of report appears on your screen:

```
_DS0/LANGUAGE/REPORT.DOC.3 to _DC0/USERS.JOHN/REPORT.DOC copied.
```

Following this report, the cursor appears next to the CIP command prompt for the local system.

The background version of USSCOPY on the remote system continues to wait for instruction. You can leave it waiting as long as you wish, or you can follow these steps to terminate it:

Step 1

Type **talkt _tt1** on the CIP command line.

This command puts you into communication with _TT15 on the remote system through _TT1 on your system.

Step 2

Type **[CTRL] c** to terminate USSCOPY on the remote system.

The cursor appears next to the CIP prompt on the remote system.

Step 3

Type **log** and strike [RETRN] to log off the remote system.

Step 4

Type **[CTRL] z [CTRL] e** to break the connection between _TT1 and _TT15 on the two systems. The cursor now appears next to the CIP prompt on the local system.

> usscopy :send _tt1 plans.doc _ds0/archive/

This example and subsequent examples assume _TT1 on the local system is connected to some unspecified port on a remote system, and a background version of USSCOPY is running on that unspecified port on the remote system.

USSCOPY sends the latest version of the file named PLANS.DOC from the default directory on the local system through the port _TTl on the local system. A file is created on the remote system named _DS0/ARCHIVE/PLANS.DOC.

```
> usscopy :send _ttl plans.doc _ds0/archive/ :hostsync=hardware
        :timeout=5
```

This is the same as the previous example except it assumes hardware host synchronization on both systems. Be aware that hardware host synchronization requires a special cable and possibly hardware modifications to your system.

Also, USSCOPY allows up to 5 seconds between characters before declaring a timeout. A timeout occurs when USSCOPY on either system expects a response from the other system and does not receive it in the expected time. Usually this means one or more characters have been lost.

Using Prompts

```
> usscopy
Serial port   > _ttl5
```

Note that the user is prompted only for the serial port. USSCOPY assumes the default action (:background), which has only one required parameter (Serial Port).

```
> usscopy :receive
Serial port   > _ttl
Source        > _ds0/language/report.doc
Destination   > *
```

This command receives the file REPORT.DOC from the remote system through the port named _TTl on the local system. The file is placed in the default directory on the local system.

Notes on Usage

USSCOPY continuously monitors the integrity of the transmission using checksums on the data packets. Automatic retransmissions occur in the event of a transmission error.

If you have trouble successfully sending or receiving files, you can try one or more of the following techniques:

1. Lower the baud rate. A slower transmission rate usually results in higher reliability. Use DSTAT to adjust the baud rate.

2. Increase the number of retries. USSCOPY tries more times on each erroneous packet. Use the :retries= switch.

3. Reduce the size of the transmission buffer. The smaller the buffer, the less likely an error may occur in any given packet. Use the :bufsize= switch.

4. Increase the length of the timeout. This may help if one or both of the two systems are especially busy and cannot send characters very often. The timeout should never be set longer than 5 seconds. Most of the time, what is reported as a timeout error is really a dropped character. Use the :timeout= switch.

Note that the same procedure used to transmit files between systems that are physically close to one another can also be used to to transmit files through modems and x.25 data links. If you are using modems or x.25 links, do <u>not</u> attempt to change the baud rate with the :baudrate switch. This is because USSCOPY cannot change the speed of the network, only the ports on the system at each end.

Related CIP Commands

copy	Copy files
sscopy	System to system file transfer
talkt	Direct communication with another device

verify

Functional Description

Use this command to compare the checksums of two or more specified files.

Command Line Syntax

Mnemonic	verify		
Required parameter	Checksum File		
Optional parameter	File List		
Switches			
File selection	:edit=	:exclude=	
Output control	:log	:pause	:badonly
Other	:rewind		

Parameters

Checksum File	Function	Required. Use this parameter to specify the name of the file containing the checksum definitions.
	Default	None.
	Syntax	Type a standard file designation. Wildcard symbols are not allowed.

File List

Function: Optional. Use this parameter to specify the names of the files to be checked.

Default: All files listed in the file containing the checksum definitions.

Syntax: Type a standard file designation. Wildcard symbols are allowed.

Switches

:badonly

Function: Use this switch to display only files that have checksum errors or that are not found.

Default: :nobadonly

Syntax: Type **:badonly**

:edit=

Function: Use this switch to specify a series of edits to be performed on the destination file designation. The edits are specified as a series of "find and switch" strings. The destination file designation is searched for the first occurrence of the "find" string. If it is found, it is replaced by the "switch" string.

Default: No edits are performed.

Syntax: Type **:edit=** followed by a list of "find and switch" strings separated by commas. The find string is separated from the switch string by a colon. Either one or both of the strings may be empty.

:exclude=

Function: Use this switch to exclude files or devices from the File List parameter.

Default: All files or devices that match the File List parameter are selected.

Syntax: Type **:exclude=** followed by a list of file or device designations separated by commas. Wildcard characters are allowed.

:log Function Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)

Default The value specified by the OPTION command.

Syntax Type **:nolog** or **:log** to override the default.

:pause Function Use this switch to stop the display after each screen of information. The display resumes when you press any key.

Default The value specified by the OPTION Command.

Syntax Type **:pause** or **:nopause** to override the default.

:rewind Function Use this switch to rewind the tape device prior to searching for files.

Default :norewind

Syntax Type **:rewind**

Examples

```
> verify /xxx/test.cks
```

This command tells VERIFY to compare all files in the default directory to the checksum file /XXX/TEST.CKS. This kind of report appears on the screen:

```
                        Verification Summary
                     File Name                        Verified Size  Sum
_____________________________________________________ ________ ____  ____
_DS0/TMP/A.EXE                                          yes    good  good
_DS0/TMP/A.OUT                                          yes    good  good
_DS0/TMP/I00204A.                                       NO     BAD   BAD
VERIFY    : Opening _DS0/TMP/JU00306A.
```

```
FAILED     : Status = 133.
MESSAGE    : The specified file could not be found.
_DS0/TMP/KILLER.EXE                                        yes    good  good
_DS0/TMP/L.MCR                                             yes    good  good
_DS0/TMP/L.MRL                                             yes    good  good
_DS0/TMP/L.PAS                                             yes    good  good
_DS0/TMP/L.PRN                                             yes    good  good
_DS0/TMP/T.PAS                                             yes    good  good
_DS0/TMP/TRECORD.PAS                                       yes    good  good
_DS0/TMP/TRECORD.TDT                                       yes    good  good
_DS0/TMP/TT.EXE                                            yes    good  good
_DS0/TMP/TT.MCR                                            yes    good  good
_DS0/TMP/TT.MRL                                            yes    good  good
_DS0/TMP/TT.PAS                                            yes    good  good
_DS0/TMP/TT.PRN                                            yes    good  good
_DS0/TMP/U.C                                               yes    good  good
18 files checked.    1 files failed.
```

This report indicates the file I00204A is not the same as the one in the checksum file /XXX/TEST.CKS. The file JU00306A is listed in the checksum file but was not found in the directory listed in the checksum file. All of the other files listed in the checksum file /XXX/TEST.CKS were found and their lengths and checksums are correct.

Using Prompts

```
> verify
Checksum file > /xxx/text.cks
```

This command performs the same function as the previous example.

Notes on Usage

VERIFY uses the checksums prepared by the CHECKSUM command to determine whether the disk or tape copy of a file matches the original.

The checksum is a relatively unique number which is computed by summing the values of the characters in the file. If the checksums for two files match, it is almost certain that they are the same.

Related CIP Commands

checksum	Generate a file containing checksum information

version

Functional Description

Use this command to display version information for WMCS commands.

Command Line Syntax

Mnemonic	version	
Optional parameter	File List	
Switches	:full	:rewind

Parameters

File List	Function	Optional. Use this parameter to specify an optional list of files to be examined by VERSION.
	Default	If nothing is specified, VERSION displays the version information for the operating system.
	Syntax	Type a standard file designation. Wildcard symbols are allowed.

Switches

:full	Function	Use this switch to display all version information in the file.
	Default	:nofull
	Syntax	Type **:full**
:rewind	Function	Use this switch to rewind the tape device.
	Default	:norewind
	Syntax	Type **:rewind**

Examples

> version

This command displays the version information for the operating system:

```
WMCS 6.0  24-Apr-1985
Copyright 1980 WICAT Systems.
```

> version sys$disk/sysexe/a*

This command displays the version information for the files in SYS$DISK/SYSEXE/ that start with the letter A:

```
_SYS$DISK/SYSEXE/ALLOC.EXE.6
        ALLOC     13-Dec-1984 WMCS 6.0
_SYS$DISK/SYSEXE/ARCH.EXE.8
        ARCH      08-Dec-1984 WMCS 6.0
```

Notes on Usage

VERSION operates by searching the file for a particular character string: @(#)| or @(#). A search for the first character string results in a report on the version of the main program. A search for the second character string (specify the :full switch) gives the version of the main program and any libraries and subroutines called by the main program. The characters following this identifying string are printed until a terminator is seen. The valid terminators are the double quotation mark, right angle bracket, backslash, new line, and null.

Since VERSION is searching the file for a character string, it operates with less speed on larger files, particularly if no version string is found in the file.

User-created commands may be processed by VERSION to display their version information if the creator of the program embeds the appropriate constant string in his program.

For example, in the case of a Pascal program, the programmer would declare a string in the constant section as follows:

```
        .
        .
        .

        CONST
              VERSION_STRING          = '@(#)|Revision 4 of my program\';

        .
        .
        .
        .
```

Related CIP Commands

None.

vew

Functional Description

Use this command to create, examine, or edit text files. Read the VEW User's Reference Manual for a complete explanation of this utility.

Command Line Syntax

Mnemonic vew

Optional parameter File List

Switches

:autoindent
:autosavecount=
:autosavetime=
:case
:insert
:worddelimiters=
:memory=
:override
:pagesize=
:preserve=
:readonly
:rightmargin=
:screenlength=
:screenwidth=
:setupin=
:setupout=
:sidescroll=
:silent
:startup
:tabwidth=
:macrofile=

Examples

```
> vew testl.pas
```

This command edits the most recent version of the file TEST1.PAS. If the file does not exist, TEST1.PAS.1 is created in your default directory.

```
> vew dave/prog.asm
```

This command edits the most recent version of PROG.ASM (if the file already exists) or creates file PROG.ASM.1 in directory /DAVE/ on the default device.

```
> vew _dx0/rootdir/report.txt.3
```

This command edits (or creates) the specified file in the main directory on device _DX0, even if _DX0 is not the default device.

```
> vew report1.txt,report2.txt,report3.txt
```

This command opens (or creates) the files REPORT1.TXT, REPORT2.TXT, and REPORT3.TXT in the default directory, and VEW begins editing REPORT1.TXT. When REPORT1.TXT is exited or aborted, VEW switches to REPORT2.TXT. When REPORT2.TXT is exited or aborted, VEW switches to REPORT3.TXT. When REPORT3.TXT is exited or aborted, VEW returns to the CIP command line. You can also switch from one file to another without exiting the files with the VEW function sf.

```
> vew myfile.000 :memory=0 :noinsert :autoindent
```

This command opens (or creates) MYFILE.000 in the default directory. In addition, the switches cause the entire file to be placed in memory, VEW starts in replace mode, and the auto indent feature is turned on.

Using Prompts

If you execute VEW for the first time since logging on without specifying a File List, VEW prompts you for the File List:

```
> vew
File list     > memo.txt
```

Notes on Usage

Read the *VEW User's Reference Manual* for a description of how to use the switches in VEW.

Related CIP Commands

zap	Edit binary files

wait

Functional Description

Use this command in command files to suspend execution of the command file until a specified time. WAIT may also be used alone to display a message at a specified time.

Command Line Syntax

Mnemonic	wait
Required parameter	Date and Time
Optional parameter	Message

Parameters

Date and Time	Function	Required. Use this parameter to specify the amount of time to wait.
	Default	None.
	Syntax	Type a date and time in the standard WMCS format.
Message	Function	Optional. Use this parameter to specify a message to be displayed at the specified time.
	Default	None.
	Syntax	Type a character string enclosed in double quotation marks.

Switches

None.

Examples

```
> wait +00:00:05 "hi there"
```

This command waits 5 seconds then prints the following message:

```
     "hi there"
```

```
> &wait today_17:00:00 "go home"
```

In this command, WAIT runs as a background process until 5 p.m. At that time, it prints the message "go home".

Using Prompts

```
> wait
Date and time > +0.0.5
```

This command performs the same function as the first example.

Notes on Usage

If you do not redirect SYS$INPUT, SYS$OUTPUT, and SYS$ERROR before logging off the system, you won't be able to log back on because WAIT has your terminal open and LOGFLUSH will not poll your terminal for log ons.

When WAIT is run as a background job, it should probably be set to a lower scheduling priority which will reduce conflict with other running processes.

Related CIP Commands

None.

watchdog

Functional Description

Use this command to log off unused processes.

Command Line Syntax

Mnemonic	watchdog
Switches	:logoffdelay=

Parameters

None.

Switches

:logoffdelay=	Funtion	Use this switch to specify how long, in minutes, an interactive process must be idle before logging it off.
	Default	:logoffdelay=30
	Syntax	Type :logoffdelay= followed by the number of minutes.

Examples

```
> &watchdog
```

This command starts up WATCHDOG in background mode with a delay of 30 minutes.

```
> &watchdog :logoffdelay=15
```

This command starts up WATCHDOG in background mode and specifies that unused processes will be logged off after 15 minutes of inactivity.

Using Prompts

None.

Notes on Usage

WATCHDOG automatically kills interactive processes that do not execute for a given period of time. It protects your system by terminating CIPs that users have forgotten to log off. This prevents unauthorized users from getting access to your system through an authorized user's vacant terminal.

Related CIP Commands

None.

who

Functional Description

Use this command to translate a UIC (User Identification Code) to a username or to translate a username to a UIC.

Command Line Syntax

Mnemonic	who			
Optional parameter	UIC or username			
Switches				
Output control	:header	:pause	:log	:full
Process selection	:all	:active	:inactive	

Parameters

UIC or username Function Optional. Use this parameter to specify the UIC you want to translate.

Default All UICs and usernames are displayed.

Syntax Type a UIC enclosed in square brackets, with group and user numbers separated by a comma or a username. Wildcard symbols and lists are allowed.

Switches

:all	Function	Use this switch to display all the users on the system.
	Default	:all
	Syntax	Type **:noall**
:active	Function	Use this switch to display only users that are currently active.
	Default	:all
	Syntax	Type **:active**
:header	Function	Use this switch to suppress the display of labels at the top of each column in the display.
	Default	:header
	Syntax	Type **:noheader**
:inactive	Function	Use this switch to display only users that are currently inactive.
	Default	:all
	Syntax	Type **:inactive**
:pause	Function	Use this switch to stop the display after each screen of information. The display resumes when you press any key.
	Default	The value specified by the OPTION Command.
	Syntax	Type **:pause** or **:nopause** to override the default.
:full	Function	Use this switch to display all information about the UIC or username.
	Default	:nofull
	Syntax	Type **:full**
:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	The value specified by the OPTION command.
	Syntax	Type **:nolog** or **:log** to override the default.

Examples

> who

This command displays all of the usernames and UICs on the system:

```
     Username      Uic
     --------   ----------
     BOB        [0003,0002]
     BRENDA     [0001,0002]
     KARL       [0002,0002]
     SYSTEM     [0001,0001]
     GRACE      [1000,1000]
     BACKUP     [0003,0001]
```

> who [1,1]

This command displays the username belonging to UIC [1,1], which is SYSTEM.

```
     Username      Uic
     --------   ----------
     SYSTEM     [0001,0001]
```

> who [*,2] :full

This command displays full information on all usernames in group number 2.

```
     Username      Uic        Status
     --------   ----------   --------
     BRENDA     [0001,0002]  inactive
     KARL       [0002,0002]  active
     MIKE       [0023,0002]  inactive
```

> who brenda :noheader :full

This command displays the UIC and status for the username BRENDA and suppresses the labels above each column in the display.

```
     BRENDA     [0001,0002] inactive
```

Using Prompts

None.

Notes on Usage

With the protection features in the WMCS, UICs have become very important. However, a UIC does not usually convey much immediate information to a user.

WHO lists the set of UICs and usernames defined on the system. The subset of UICs with processes running can be specifically selected.

Related CIP Commands

None.

wibug

Functional Description

Use this command to debug an executable file. WIBUG is an assembly-language level, symbolic debugger. For details on how to use WIBUG, see the WIBUG Programmer's Reference Manual.

Command Line Syntax

Mnemonic	wibug
Optional parameters	Filename File parameters

Notes on Usage

Read the WIBUG Programmer's Reference Manual.

Related CIP Commands

None.

wscan

Functional Description

Use this command to search files for lines that match a particular wildcard pattern.

Command Line Syntax

Mnemonic	wscan			
Required parameters	File List Pattern			
Switches				
File selection	:before= :mod	:since=	:exclude	:uic
Display control	:context= :log	:number	:header	:pause
Pattern	:exact			
Other	:confirm	:rewind		

Parameters

File List	Function	Required. Use this parameter to specify the names of the files to be scanned.
	Default	None.
	Syntax	Type a list of file designations separated by commas. Wildcard symbols are allowed.

Pattern

Function Required. Use this parameter to specify the pattern to be scanned for. In other words, WSCAN looks for all the lines in the specified files that match what you type as the value for this parameter. If any portion of a line matches the pattern you type for this parameter, the line is displayed. You cannot specify a pattern that spans more than one line.

Default None.

Syntax Type a pattern string.

The same wildcard metacharacters used in filelists may be used to specify the pattern. The following is a summary of these characters:

= The equal sign matches, or signifies any single character.

* The asterisk matches any number of characters and spaces.

[] The square brackets represent any one of a set of characters. Ranges of characters can be specified, e.g. [c-m] represents any of the characters c, d, e, ... , m. Characters can be excluded from the range, e.g. [c-m^f^l] means all characters in the range c through m, but not f or l. Ranges of characters can be excluded, e.g. [c-m^f-l] means all characters in the range c through m, but not f, g, h, i, j, k or l.

() Parentheses match a numeric field.

For instance, (10-15) matches any numeric field with a value in the range 10 through 15. Lists of ranges can be specified. (10-15,30-45) matches any numeric field with a value between 10 and 15 or between 30 and 45, inclusive.

WSCAN does not distinguish between upper- and lowercase characters.

Switches

:before=

Function — Use this switch to select only those files that match the file list parameter and were created or modified before the specified date and time.

Default — Selects all files that match the file list parameter.

Syntax — Type **:before=** followed by a date and/or time in the standard date and time syntax.

:confirm

Function — Use this switch to confirm or deny the desired operation on each file or device.

Default — :noconfirm

Syntax — Type **:confirm**

:context=

Function — Use this switch to specify the number of lines of context to be displayed with each line that matches the specified pattern.

Default — Only the line that matches the pattern is displayed, :context=0.

Syntax — Type **:context=** followed by a numeral between 1 and 10.

:exact

Function — Use this switch to specify that SCAN or WSCAN will match only a line that completely satisfies the pattern. That is, it will not match a line that satisfies the pattern and has additional characters.

Default — :noexact

Syntax — Type **:exact**

:exclude=

Function — Use this switch to exclude files or devices from the File List parameter.

Default — All files or devices that match the File List parameter are selected.

Syntax — Type **:exclude=** followed by a list of file or device designations separated by commas. Wildcard characters are allowed.

:header   Function   Use this switch to display the full file specification as a header.
           Default   :noheader
           Syntax   Type **:header**

:log   Function   Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
           Default   The value specified by the OPTION command.
           Syntax   Type **:nolog** or **:log** to override the default.

:mod   Function   Use this switch to specify that the modification date is to be used in all date and time considerations by the :before= or :since= switches.
           Default   :nomod
           Syntax   Type **:mod**

:number   Function   Use this switch to specify that a line number is to be added at the left margin of each line that is displayed.
           Default   :nonumber
           Syntax   Type **:number**

:pause   Function   Use this switch to stop the display after each screen of information. The display resumes when you press any key.
           Default   The value specified by the OPTION Command.
           Syntax   Type **:pause** or **:nopause** to override the default.

:rewind   Function   Use this switch to rewind the tape device.
           Default   :norewind
           Syntax: Type **:rewind**

:since=   Function   Use this switch to select only those files that match the file list parameter and were created or modified since the specified date and time.
           Default   Selects all files that match the file list parameter.
           Syntax   Type **:since=** followed by a date and/or time in the standard syntax.

:uic=	Function	Use this switch to select only those files or devices that are part of the given list and are owned by the specified user or list of users.
	Default	Selects all files that match the specified list.
	Syntax	Type **:uic=** followed by a list of UICs or usernames.

Examples

```
> wscan *.pas writeln
```

This command searches the most recent version of each file in the default directory that has an extension of .PAS for any occurrence of the string "writeln".

When WSCAN finds a line that matches the pattern, this kind of report appears on your terminal screen:

```
     writeln("Hi mom!");
```

```
> wscan *.doc transi[st]ion :context=3
```

This command searches the latest version of all the files in the default directory with an extension of .DOC. All lines with the word **transision** or **transition** in them are displayed. The :context=3 Switch prints the 3 lines before and the 3 lines after the line with the pattern in it.

```
> wscan huxley.txt *Huxley :exact
```

By default, WSCAN looks for the pattern you specify anywhere on a line. Use the :EXACT Switch to find a line that matches the pattern in its entirety. For instance, the command line above matches only the lines that end with the word **huxley.**

```
> wscan *.txt "macro processors" :header :number
```

This command searches the latest version of all files in the default directory with an extension of .TXT for lines with the phrase "macro processors". The :header switch displays the name of each file scanned.

The :number switch displays the relative line number of each line in the left margin of the report.

The following kind of report appears:

```
Scanning _DC0/USERS.HANSEN/HANDBOOK.TXT.5
7: A study of macro processors is rewarding at two levels. Firstly from a
13: varied. Even the term 'macro processor' is not standard; alternative
27: many different macro processors. Some of them are tied to individual
34: automated using a macro processor. A macro would be defined for each of
```

In the foregoing display, the term **macro processor** appears four times in the file named HANDBOOK.TXT.5. It first appeared on line 7. It also appears on lines 13, 27 and 34.

Using Prompts

```
> wscan
File list    > *.txt
Pattern      > "macro processors" :header :number
```

This command performs the same function as the previous example.

Notes on Usage

WSCAN searches each file specified in the first parameter to find any line(s) containing the pattern specified in the second parameter.

Note that SCAN and WSCAN both have the same function. WSCAN is substantially faster than SCAN, but it doesn't allow complex patterns with alternation. If you are searching for a simple pattern, WSCAN will do the job. For complicated patterns, use SCAN.

WSCAN can be used with pipes as a filter of the output of other utilities. For instance, the following command line will display all of the processes whose process name begins with CIP_. (CIP_V is excluded to prevent display of CIP_VEW.)

```
> pstat | wscan sys$input cip_[^v]
```

To exaggerate the point a little, you could even pipe the output of the previous command to the count program to count the number of CIPs. The following command line and report show how to do this:

```
> pstat | wscan sys$input cip_[^v] | count sys$input
```

The file name _C17E6$04 is the name of the pipe between WSCAN and COUNT. This report shows that 5 CIPs are running on the system, not including CIP_VEWs.

Related CIP Commands

scan	Search files for a pattern

wsort

Functional Description

Use this command to sort files according to specified keys. (Read the *WSORT User's Reference Manual* for a complete explanation of the WSORT utility.)

Command Line Syntax

Mnemonic	wsort			
Required parameters	File List Destination			
Switches				
File selection	:before=	:exclude=	:mod	:since=
	:uic=			
File sorting	:ascending	:delimiter=	:descending	:duplicates
	:ignoreleading=	:keyN=	:skip=	:sort=
Key modifiers	allowerrors	ascending	ascii	asciidecimal
	asciifloatingpoint	asciihexadecimal		asciireal
	binary	bitfield	caseless	delimiter=
	descending	dictionary	endat=	field=
	floatingpoint	ignoreleading=	length=	location
	offset=	real	skip=	startat=
	width			
Position modifiers	(number)	bcddigits=	bits=	bytes=
	characters=	doublewords=	nibbles=	words=
Others	:errorfile=	:errorhandling=	:log	:maxrecordlen=
	:memory=	:recordlen=	:stats	:text

Examples

```
> wsort names.txt sorted.txt
```

This command sorts the file NAMES.TXT in ascending order and puts the result in the file SORTED.TXT.

Using Prompts

```
> wsort
File list      > names.txt
Destination    >  sorted.txt
```

This command performs the same function as the example above.

Notes on Usage

Read the WSORT User's Reference Manual for a complete explanation of the switches and sorting options.

If more than one position modifier is used, they must be separated by commas and enclosed in double quotation marks. Example: "bytes=12,bits=7".

Position modifiers specify a relative position, i.e., 1 means the first, 2 means the second, etc. If several modifiers are used, they are relative to each other, with the first being relative to the beginning of the record. For example, "byte=1,bit=1" specifies the first bit in the first byte.

Length modifiers specify a sort key length and are additive, not relative. Therefore, the example "words=2,nibbles=2,bits=16" is the same as "bytes=7", not "bytes=5,bits=4" as would be the case for a position modifier. If more than one length modifier is used, they must be separated by commas and enclosed in double quotation marks. Example: "bytes=12,bits=7".

Related CIP Commands

sort	Sort the contents of one or more files

xfer

Functional Description

Use this command to copy an entire disk. Unlike COPY, which is used to copy files that exist on a disk, XFER copies from disk to disk, file to disk, and disk to file.

Command Line Syntax

Mnemonic	xfer			
Required parameters	Source Destination			
Switches	:verify	:log	:rewind	:sectors=

Parameters

Source	Function	Required. Use this parameter to specify the name of a file or mounted disk device.
	Default	None.
	Syntax	Type either a devicename beginning with an underscore or a filename. Wildcard symbols and lists are not allowed.
Destination	Function	Required. Use this parameter to specify the name of a file or a mounted ":SPECIAL" disk device.
	Default	None.
	Syntax	Type either a devicename beginning with an underscore or a filename. Wildcard symbols and lists are not allowed.

Switches

:log	Function	Use this switch to specify whether log messages are displayed. (Log messages are informational displays that indicate what the utility is doing.)
	Default	:log
	Syntax	Type **:nolog**
:rewind	Function	Use this switch to rewind the tape device.
	Default	:norewind
	Syntax	Type **:rewind**
:sectors=	Function	Use this switch to specify a range of sectors to be copied.
	Default	All sectors are copied.
	Syntax	Type **:sectors=** followed by a range of sector numbers.
:verify	Function	Use this switch to perform a VERIFY instead of a COPY function.
	Default	:noverify
	Syntax	Type **:verify**

Examples

```
> xfer _dx0 _dxl
```

This command copies all of _dx0 to _dxl, sector by sector.

```
> xfer sys$disk/systemp/diskimage.tmp _dx0 :verify
```

This command verifies that the disk image contained in the file SYS$DISK/SYSTEMP/DISKIMAGE.TMP matches the disk which is mounted in _DX0.

Using Prompts

```
> xfer
Source      > _dx0
Destination > tempfile.tmp
```

This command makes a copy of _DX0 and puts it in the file TEMPFILE.TMP in the default directory.

Notes on Usage

The user must have READPHYS privilege to read from a disk device with XFER. To write to a disk device, the user must have WRITEPHYS privilege and the disk must be mounted with the :SPECIAL switch. Disk devices must be mounted prior to using XFER.

XFER is useful for making a backup copy of a floppy disk. Assuming that there are two disk drives on the system, mount (see the command description for MNT) the source disk in one drive and the properly formatted (see the command description for DINIT) destination disk in the other drive. Use XFER to copy the source disk to the destination disk. After copying, dismount (see DMNT) the disks and remove them from the drives. The destination disk is now an exact image of the source disk.

Here is the command sequence for this operation:

Example:

```
> ! -------- Format the destination disk
> ! Insert blank disk in second drive
> DINIT _dx1 temp :format
> !
> ! -------- Make the destination disk same as source disk
> ! Insert source disk in first drive
> MNT _dx0
> MNT _dx1 :special
> XFER _dx0 _dx1
> DMNT _dx*
> ! Remove disks
```

Many systems have only one floppy disk drive. In this case, the source disk is mounted first. XFER is used to copy an image of the source disk into a file on the system hard disk. The source disk is dismounted and removed from the drive. Then the properly formatted destination disk is inserted and mounted. XFER is now used to copy the source disk image from the temporary file onto the destination disk. The destination disk is dismounted and removed from the system.

Here is the command sequence for this operation:

Example:

```
> ! ------- Make a copy of the source disk
> ! Insert source disk
> MNT _dx0
> XFER _dx0 sys$disk/systemp/diskimage.tmp
> DMNT _dx0
> ! Remove source disk
> ! ------- Format the destination disk
> ! Insert blank disk
> DINIT _dx0 temp :format
> !
> ! ------- Make the destination disk same as source disk
> MNT _dx0 :special
> XFER sys$disk/systemp/diskimage.tmp _dx0
> DMNT _dx0
> ! Remove destination disk
> !
> DEL sys$disk/systemp/diskimage.tmp
> ! Done
```

XFER can also be used to verify that one disk is the same as another. It is not required to mount the second disk with the SPECIAL Switch since only read operations are performed during the verify.

Here is the command sequence for this operation:

Example:

```
> ! ------- Verify that two disks are the same
> ! Insert disk 1
> MNT _dx0
> XFER _dx0 sys$disk/systemp/diskimage.tmp
> DMNT _dx0 :auto
> ! Remove disk 1
> !
> ! Insert disk 2
> MNT _dx0
> XFER sys$disk/systemp/diskimage.tmp _dx0 :verify
> DMNT _dx0 :auto
> ! Remove disk 2
> !
> DEL sys$disk/systemp/diskimage.tmp
> ! Done
```

The use of XFER to read a disk requires READPHYS privilege. The use of XFER to write a disk requires WRITEPHYS and OPERATOR privilege (to mount the disk with the :SPECIAL Switch).

Related CIP Commands

mnt	Mount a device
copy	Copy files
dmnt	Dismount a device

zap

Functional Description

Use this command to edit files that contain binary data, such as executable files, directory files, and certain system files. For information on how to execute the ZAP program, read the ZAP User's Reference Manual.

Command Line Syntax

Mnemonic	zap		
Required parameter	Filename		
Switches	:mask8bit	:memory=	:readonly
	:rewind	:setupin=	:setupout=

Notes on Usage

Read the ZAP User's Reference Manual.

Related CIP Commands

dump	Display file contents in hexadecimal format
vew	Edit a text file

Chapter 7

Using CIP Commands in Files

A command file is a text file (that you can create and maintain with the VEW text editing program) that contains lists of commands to be executed by the CIP.

Because it is frequently necessary to perform functions that require the execution of a series of CIP commands, a command file can be used to group commands so that you can execute the command file rather than typing each command in the series.

For example, suppose that before you log off each day you wish to perform the following:

1. Purge your user-account default directory of unwanted copies of files that you have worked on that day.

2. Place a copy of each file modified that day in an archive file on a diskette.

The following command-line character strings in a command file would allow you to perform the foregoing operations:

```
cd _dc0/usrs.al/ :perm
dir
pu
prompt "INSERT DISKETTE AND PRESS [RETRN]"
mnt _dx0
cd _dx0/rootdir/ :perm
arch _dx0/archive.000 _dc0/usrs.al/* :new :mod :since=today
arch _dx0/archive.000 *.*.* :list
```

In other words, were the name of this file DAYEND.COM, you could type @DAYEND onto the CIP command line, strike [RETRN], and the commands in the command file would be executed, one line at a time, in the order in which they appear in the file.

How to Create a Command File

Use the VEW Command to create or modify a command file.

The default file extension for a command file is .COM.

Each line of a command file must be treated as a CIP command line, i.e., the CIP command line character limit, command limit, and other syntactic rules apply to each line in a command file.

Finally, strike [RETRN] at the end of every line in a command file. The Return Key is the command line terminator for the CIP Program. Therefore, if a line in a command file is not terminated by striking [RETRN], the CIP does not execute the line.

The CIP's other capabilities regarding command files are discussed in the following sections.

Comments in Command Files

In large command files it may be difficult for you to follow the steps that the CIP is being instructed to perform. Therefore, the CIP provides for the use of the exclamation point, !, to indicate to the CIP that what follows is to be ignored. In other words, whatever follows the exclamation point, !, will not be executed.

This allows you to insert comments within the file for your own reference.

The exclamation point can be inserted anywhere on a line.

The following example illustrates the use of comments:

```
     option :noverify                  ! Suppress display of command file
                                       !  contents
     cd _dc0/usrs.al/ :perm            ! Assign /USRS.AL/ as the default
     dir *.*.*                         ! List all versions of all files
                                       !  in /USRS.AL/
     pu                                ! Purge /USRS.AL/
     prompt "INSERT DISKETTE AND PRESS [RETRN]"
     mnt _dx0                          ! Mount a diskette
     cd _dx0/rootdir/ :perm            ! Assign _DX0/ROOTDIR/ as default
     arch _dx0/archive.000 _dc0/usrs.al/* :new :mod :since=today
                                       ! Include all files modified today
                                       !  in _DX0/ARCHIVE.000
     arch _dx0/archive.000 *.*.* :list! List contents of ARCHIVE.000
```

Parameter Substitution in Command Files

The following sample command file allows you to archive files in a particular directory. If you want to perform such a function for several directories, you can prepare a similar command file for each directory, or you can modify the command file so that you could specify command file parameters at the time you execute the command file.

In the following example, USRS.TOM becomes the first parameter value in the execution of DAYEND.COM. To use DAYEND.COM in this manner, you need to type a substitution marker such as 'pl' at the locations indicated.

When you type @DAYEND onto the CIP command line, you can then strike the spacebar and specify the name of the directory whose modified files you want archived. For example:

```
     > @dayend usrs.tom
```

Wherever you type 'pl' in the command file, the CIP substitutes the first command file parameter typed on the CIP command line when the command file is executed. Consequently, when you type @DAYEND USRS.TOM on the CIP command line, you indicate to the CIP that you want USRS.TOM substituted for 'pl' in the text of DAYEND.COM. Were you to type @DAYEND USRS.AL onto the CIP command line, USRS.AL would be substituted as the first parameter value.

You can specify up to eight command file parameters, using 'pl', 'p2', 'p3', etc., as substitution markers within the text of the command file to indicate where the parameter values are to be substituted. Each substitution marker can appear any number of times within the command file.

```
option :noverify                    ! Suppress display of command file
                                    !  contents
cd _dc0/'pl'/ :perm                 ! Assign 'pl' as the default
dir *.*.*                           ! List all version of all files
                                    !  in 'pl'
pu                                  ! Purge 'pl'
prompt "INSERT DISKETTE AND PRESS [RETRN]"
mnt _dx0                            ! Mount a diskette
cd _dx0/rootdir/ :perm              ! Assign _DX0/ROOTDIR/ as default
arch _dx0/archive.000 _dc0/'pl'/* :new :mod :since=today
                                    ! Include all files modified today
                                    !  in _DX0/ARCHIVE.000
arch _dx0/archive.000 *.*.* :list ! List contents of ARCHIVE.000
```

NOTE: You must include the single quotation marks. If you do not specify the value to be substituted within a command file, e.g., you have 'pl' in a command file and do not type the value (onto the command line) that the CIP is to put in place of 'pl,' the CIP executes the command file and 'pl' is ignored. Thus, if 'pl' represents a required parameter, the parameter will be prompted for. If 'pl' represents an optional parameter, the command file is executed as though 'pl' were not part of the file.

Logical Names in Command Files

Logical names can be included in command files.

For example, suppose that you have the following logical name assignment for mounting a diskette in _DX0:

```
mf :== "mnt _dx0"
```

Furthermore, suppose that you create a command file in which you want to instruct the CIP to mount a diskette in _DX0. The foregoing logical name assignment could be used as follows:

```
prompt "INSERT THE FIRST VOLUME IN _DX0.  PRESS [RETRN]."
mf
cd _dx0/rootdir/ :perm
pu
```

Note that the operator you use for the logical name assignment, e.g., :=, :==, :===, determines whether the assignment pertains only to the command file or to the user process. For example, if you use := to specify a logical name assignment within a command file, that assignment is valid only for the duration of the execution of the command file. On the other hand, :== assigns the logical name assignment to the entire process, while :=== assigns logical names to the system logical name tables, making them immediately accessible to all users.

Read the PROMPT Command description for information on how to use RESPONSE as a logical name within command files.

Nested Command Files

A nested command file is a command file that is executed by a command line character string in another command file. For example, AMROUTINE.COM might contain the following command line character string:

```
@USRS.TOM/CHECK
```

This string executes CHECK.COM in /USRS.TOM/. CHECK.COM is therefore a nested command file. CHECK.COM can also contain strings that execute other command files.

The amount of available memory is the only limit on the number of nested command files you can have.

Calling Parameter Files from Command Files

All utilities use the same procedure for interpreting command lines and handling all types of parameters.

The CIP reads through the command file and receives the command-line parameters as if they were given on the command line.

The order for parameters is required parameters, optional parameters, and then switches.

All items enclosed in double quotation marks are considered one parameter.

```
cd _dc0/users.tom/ :perm
dir
pu
prompt "INSERT DISKETTE AND PRESS [RETRN]"
mnt _dx0
cd _dx0/rootdir/ :perm
arch _dx0/archive.000 _dc0/users.tom/* :new :mod :since=today
arch _dx0/archive.000 *.*.* :list
```

In the foregoing example, the prompt INSERT DISKETTE AND PRESS [RETRN] is taken as a single parameter because the characters are enclosed in double quotation marks.

Furthermore, the parameters for the ARCH command in the last two lines of the command are required and are taken before the switches that follow, i.e., :new, :mod, :since=.

Search Paths

The search path is the list of places the CIP is to look to find executable image files. See the :path= switch of the OPTION Command description in Chapter 6 of this manual, the Dictionary of CIP Commands.

For example, this is the default value for the :path= switch for the OPTION Command:

```
/sysexe/,/sysexe.users/
```

This tells the CIP that when the user types a mnemonic for a CIP command (or the filename of a file containing an executable image) the CIP is to look first in his current default directory. If it is not there, then it is to look in /SYSEXE/ and then /SYSEXE.USERS/.

You can specify any path or paths you wish (the list of paths can contain up to 93 characters).

If a slash is the first character of a path, the CIP assumes that the directories on the path are on the system disk.

USERUP.COM

Your system manager should create a USERUP.COM file in your user-account default directory.

However, any user can create such a file as long as USERUP.COM is the filename and extension assigned to the file.

Use the VEW Command to create or edit the contents of USERUP.COM.

USERUP.COM is executed automatically whenever you log on. Consequently, if there are logical name assignments, special commands, command files, etc., that you want available whenever you log on, use this file to make them part of your working environment.

The following is an example of the contents of a USERUP.COM file:

```
     cdi :== "@_dc0/usrs.al.ecomcon/cdi"
      al :== "cdi _dc0/usrs.al/"
    elan :== "cdi _dc0/usrs.elan/"
    flop :== "cdi _dx0/rootdir/"
printlog :== "vew _dc0/usrs.al/printlog.000"
     tom :== "cdi _dc0/usrs.tom"
       v :== vew
     win :== "cdi _dc0/rootdir/"
```

USEROFF.COM

Your system manager creates a USEROFF.COM file in your user-account default directory.

However, any user can use the VEW Command to create such a file (as long as USEROFF is the filename and .COM is the file extension assigned to the file).

For example, if you want to type a message such as GOOD NIGHT on your terminal screen when you log off, put the following command line in your USEROFF.COM file:

```
typeb GOOD NIGHT
```

USEROFF.COM is executed whenever you log off. Consequently, if there are special commands, command files, etc., that you want executed whenever you log off, use this file to do so.

Chapter 8

Parameter Files

A parameter file can be created when the command line for a frequently used CIP command becomes to long for you to type it conveniently on the CIP command line.

This is an example of the contents of a parameter file to be used with ARCH.

```
archive.001
*.doc.*,usrs.al.report/*.*,usrs.tom.memoranda/jan.*.*,
usrs.elan.doc/*.*.*
:insert
:confirm
```

Note the following in the foregoing example:

1. The command mnemonic is not included in the parameter file.

2. Because a parameter value can be continued on another line, type a comma at the end of the line and continue the parameter value on the next line. However, do not split a file designation. In other words, do not put part of a file designation on one line and the remainder of the designation on the next line.

3. For convenience, type each parameter value on a separate line.

Were the foregoing parameters and switches in a file named SUMMARY.PRM, you could type the following onto the CIP command line to use them with ARCH:

```
> arch @summary
```

NOTE: The parameter file (the file containing the parameter values and switches) must have a .PRM file extension.

Parameter Substitution

Parameter substitution in parameter files is the same as parameter substitution in command files (read chapter 10, Parameter Substitution in Command Files).

Comments in Parameter Files

Insert comments in parameter files just as you would insert comments in a command file (read Chapter 7).

Chapter 9

Background Processing

Each CIP command is generally executed in what is called interactive or foreground mode. That is, when you type a command mnemonic and strike [RETRN], control of the terminal is given to that command until its execution is either completed or killed, making the CIP inactive. For example, when you type DIR onto the CIP command line and strike [RETRN], you must wait for the completion of that command (or its termination) before you can execute another CIP command.

Nevertheless, the execution of some CIP commands requires considerable time. Furthermore, such commands do not require that you type anything once they are executed.Therefore, it is best to initiate the command in such a way that it runs in the background, freeing up the terminal for interactive processing.

For example, a compiler frequently takes several minutes to compile a large program, yet requires no user input. It is therefore desirable that the compiler run in the background. CIP provides this capability.

To initiate a process in background mode, type an ampersand, &, followed by the command mnemonic. In other words, the syntax following the ampersand is identical to what you type when executing the command in foreground mode, that is the kind of execution described under Command Line Syntax in each command description in this manual. For example, the following command line character string would tie up your terminal for some time:

```
     > link alpha,beta,check,doublechk,sum :stack=$2000 :symbol=_HEAP
```

However, a background execution of the same command line character string allows you to use your terminal while the string is executed:

```
     > &link alpha,beta,check,doublechk,sum :stack==$2000 :symbol_HEAP
```

When you strike [RETRN], the cursor reappears on a new CIP command line.

Inasmuch as programs executed in background mode write messages to SYS$OUTPUT, you may want to temporarily redirect the output for a background process to the _NULL Port so that output would not appear on the terminal screen.

You can execute command files in background mode by typing ampersand, at sign (&@) immediately before the file designation for the command file. It is generally advisable to redirect the output of such a background execution so that you retain the use of your terminal during the execution.

If you have several processes that you would like to run in background mode, use the SUBMIT Command. This command allows you to put the processes into queues for execution at a later time in background mode. See the command description for SUBMIT in Chapter 6 for more information.

Chapter 10

Input/output redirection

The information in this chapter is not requisite to using the CIP and, therefore, is designed primarily for the experienced user. However, this information will allow the experienced user to take advantage of certain features of the CIP.

Default Input/Output Files

All WMCS processes have three standard files assigned to them. These three files are:

1. sys$input

2. sys$output

3. sys$error

These assignments appear in the table of logical name assignments that pertain to the user process, i.e., the value assigned to SYS$INPUT in the table of logical name assignments is the input file for the process. The Equivalence assigned to SYS$OUTPUT is the output file for the process, and SYS$ERROR is the error file.

When you log on, the name of the serial port to which your terminal is connected is assigned as the Equivalence for each of these names. Thus, your terminal keyboard becomes the file from which your user process receives input, and your terminal screen becomes the file to which output from your user process as well as diagnostic messages pertaining to your user process are written. For example, were you to log onto a terminal connected to serial port _TT3, the following assignments would appear in the list of logical name assignments belonging to your user process:

```
SYS$ERROR = _TT3
SYS$INPUT = _TT3
SYS$OUTPUT = _TT3
```

Sys$input is used by the process to receive command input. It is not used as the data source for filters. For example, COPY does not copy its sys$input to sys$output. Data to be filtered by a program is identified as a parameter to the program. When using a utility with a pipe, the value of the parameter representing the input file can be sys$input.

Using sys$input as the command source and not as the data source allows the process to prompt for parameters, to operate on lists of data source files, and to do its own file name wildcarding.

Sys$output is used to display prompts to the operator, and for some utilities it is used to write the data resulting from the operation (for example, dir writes its report to sys$output). Sys$output is not used for displaying log messages. Prompts for more parameters or for operator commands are written to sys$output. Typically, the filtered data is not written to it either, but to a file identified by a parameter. When output is directed to a pipe, the operator can have the filtered data written to the pipe by specifying sys$output as the name of the output file.

NOTE: Prompting and pipes are mutually exclusive. You cannot use pipes and expect to be prompted for parameters or command input. When using pipes, all parameters must be specified on the command line.

Sys$error is used to report errors to the operator and to report log messages. All errors detected by the program are reported to this file. Log messages (keeping the operator apprised of progress) are also written to this file. If log messages were written to sys$output, there would be conflicts in the use of pipes, that is, the data stream would contain both data and log messages.

All lines of text written to sys$output or sys$error are written assuming that the cursor is at the left margin. The line is terminated with a line feed, leaving the cursor at the beginning of the next line.

Sys$input is opened with the shared mode enabled. This means that if a parent process has the file assigned to sys$input open, the current process will use that same file, positioned at the same place.

Sys$output and sys$error are created with the shared mode enabled. If a parent process has the file assigned to sys$output or sys$error open, the current process will use the same file, and will begin writing where the parent process left off. Using this feature allows a set of utilities to create a common log file, as opposed to a set of log files, one for each process. If the parent does not have the file assigned to sys$output or sys$error open, the current process will create a new output file.

NOTE: There is a fourth filename that is not intrinsic to the utilities, but can be used by the operator to force communication with the user. Sys$prompt is always assigned to the name of the terminal associated with the current process. None of the utilities will open this file to get commands. The operator must explicitly redirect input to come from sys$prompt.

Any process created by your user process not only has input, output, and error files assigned to it, but (unless you specify otherwise) has the same Equivalences assigned for these files as your user process. For example, were you to execute the DIR Program while logged onto _TT3, the input, output, and error files for DIR would be the same as for your user process. As a result, the output from DIR, as well as any diagnostic messages associated with DIR, are written to _TT3.

The assignment of these files when you log on is permanent, i.e., valid for the duration of your user process (use the ASSIGN Command to change the Equivalence assigned to SYS$INPUT, SYS$OUTPUT, or SYS$ERROR for your user process).

Nevertheless, there are times when a temporary reassignment of the input, output, or error file for a process is desirable. For example, if you wish to have a copy of the report generated by the DIR Program, you must be able to redirect the output for that program to a file other than your terminal screen. This temporary reassignment is called I/O Redirection (Input/Output Redirection).

You can make a temporary reassignment for any one of these files without affecting either of the other files, or you can temporarily reassign a value to each.

I/O Redirection is performed when you type a command line character string (as illustrated in the following sections), and is valid only for the duration of the process generated by that character string. Consequently, the files assigned to your user process are unaffected by I/O Redirection.

NOTE: When you type parameter values by prompt, I/O Redirection must be specified when you type the command mnemonic, i.e., you cannot redirect input, etc., after you are prompted for a parameter value.

Temporarily Redirecting Input

Use the left angle bracket, <, to reassign the input file for a process.

The following syntactic rules apply to redirecting input:

1. The left angle bracket can be located anywhere following the command mnemonic on the command line.

2. Put at least one space immediately before the left angle bracket.

3. Spacing between the left angle bracket and the name of the input file is allowed.

4. At least one space must follow the name of the input file if anything follows the input filename in the character string.

Redirecting input has a number of applications, including the preparation of a script, for a program. For example, having prepared the script file, you can invoke the program and, as part of the CIP command line character string by which you do so, specify the script file as the default input file. The program then reads the data from the script file as though you were typing in the data at that moment. In other words, the utility is able to receive command input which is embedded in command files.

Temporarily Redirecting Output

Use the right angle bracket, >, to reassign the output file for a process.

The following syntactic rules apply to redirecting output:

1. The right angle bracket can be located anywhere following the command mnemonic on the command line.

2. Put at least one space immediately before the right angle bracket.

3. Spacing between the right angle bracket and the name of the output file is allowed.

4. At least one space must follow the name of the output file if anything follows the output filename in the character string.

For example, you can type the right angle bracket anywhere on the CIP command line and follow it with the name of the temporary output file:

```
> dir >dir.lst
```

This command string calls for the directory listing to be written to the file DIR.LST (located in the default directory) rather than to the terminal screen.

Redirecting and Appending Output

Use two right angle brackets, >>, to redirect the output of a process and append it to a file.

For example, the following command writes the output from DIR to the end of the file REPORT.TXT:

```
> dir >>report.txt
```

Temporarily Redirecting Diagnostic Messages

Use the circumflex, ^, to temporarily reassign the error file, i.e., the file to which diagnostic messages are written.

For example, you can use the following kind of command line character string to keep diagnostic messages, generated by the execution of the command line character string, from appearing on your terminal screen:

```
> alpha ^_null
```

The foregoing character string directs all diagnostic messages, generated when ALPHA.EXE is executed, to the _NULL port.

Note that such a character string redirects errors from ALPHA.EXE, not from the CIP. For example, if you type the foregoing character string and ALPHA.EXE is not in the default searchpaths, the CIP writes a diagnostic message to your terminal screen.

Input/output redirection

Use the following kind of command line character string if you want a text-file copy of diagnostic messages generated when you execute a program:

```
> alpha ^alpha.err
```

Use such an error file to examine diagnostic messages after the execution of the program, or to separate the diagnostic messages from other output.

The following syntactic rules apply to the temporary redirection of errors:

1. The circumflex, ^, can be located anywhere following the command mnemonic on the command line.

2. Put at least one space immediately before the circumflex.

3. Spacing between the circumflex and the name of the error file is allowed.

4. At least one space must follow the name of the error file if anything follows the error file name in the character string.

Redirecting and Appending Diagnostic Messages

Use two circumflexes, ^^, to redirect the diagnostic message and append it to a file.

For example, the following command writes all diagnostic messages generated by the CHECKSUM Command to the end of the file CHECKSUM.ERR:

```
> checksum report.txt ^^ checksum.err
```

Multiple Temporary Redirection

You can temporarily reassign a process's input, output, and error files to a single destination by combining the significant characters on the command line.

For example, you can initiate a program (even CIP itself) on another terminal:

```
> &cip >_tt2 <_tt2 ^_tt2
```

The foregoing command line character string assigns the input, output, and error files, for this copy of CIP, to serial port _TT2.

In cases where you have multiple I/O redirection to the same file, you can combine the redirection symbols in front of the filename. The following command line performs the same function as the previous example:

```
> &cip <^> _tt2
```

The following command temporarily redirects both the output (a directory list) and the error files (sys$error) of DIR to the same file (JUNK.DAT):

```
> dir >^ junk.dat
```

The standard error output can be piped with the standard output by combining the vertical bar and circumflex characters. For example, the following command pipes both the error files and the output generated by VERIFY to the file MYFILES.DAT:

```
> verify myfiles.cks |^ dispatch myfiles.dat
```

I/O Redirection and the _NULL Port

The _NULL Port is the system's black hole, or bit bucket. Data can be written to the _Null Port, but no data can be read from that port.

Redirect output to the _NULL Port when you want that output to disappear. For example, the following kind of command line character string keeps output from the execution of a command file from appearing on your terminal screen:

```
> @index >_null
```

Pipes

A pipe is a named sequential file kept in memory, and used for interprocess communication.

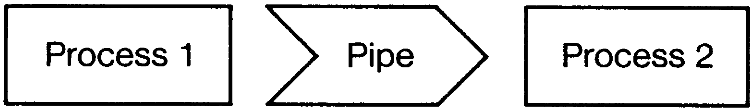
Read the WMCS Programmer's Reference Manual for details on the mechanism to use in writing programs that use pipes. That mechanism usually involves redirecting a program's output to a pipe, whereupon the data written to the pipe can be assigned as input to a second process.

Input/output redirection

Implementing filters is one of the most important uses of pipes. A filter is a program that reads input; manipulates it, e.g., the data are sorted into alphabetical order, all lowercase characters are transformed to uppercase, all trailing space characters are removed, or the source code is translated into object code; and writes the results.

Filters are commonly strung together in such a way that the data are significantly altered after they have passed through several filters.

You can direct the CIP to establish pipes, and then connect programs, using those pipes to obtain the desired effect. For example:



In other words, I/O Redirection makes it possible for you to designate the pipe as the output for process 1, and the input for process 2.

This is the syntax, recognized by the CIP, that makes this possible:

```
> dir | dispatch snoop.dat
```

The foregoing command line character string tells the CIP to write the output from DIR to a pipe. DISPATCH then reads the data from the pipe (represented by the vertical bar) and writes them to SNOOP.DAT (a text file), as well as the screen.

CIP creates the required pipes; then dismounts them upon completion of the process.

You can use as many pipes as will fit on the command line as shown in the following command line character string:

```
> cmd1 | cmd2 | cmd3 | cmd4 ...
```

There are two ways in which you can mount a pipe:

1. Use the vertical bar as part of a command line character string.

2. Use the MNT Command.

When you use the vertical bar, the pipe is mounted automatically. For example, suppose that you type the following command line character string:

```
> @parse | dispatch job.log
```

This character string indicates that you want to send the output from the execution of PARSE.COM to a pipe, and that DISPATCH will read the data from the pipe and send it to JOB.LOG and SYS$OUTPUT.

When you specify a pipe in this manner, the WMCS mounts it automatically and then dismounts it when the execution of the command line is completed. This is the kind of designation assigned to such a pipe while it is listed among the system's mounted devices:

```
    _C2078$01
```

The four characters, the dollar sign ($), and the next two characters of the foregoing devicename are the equivalent of the PID assigned to the CIP by which the pipe was mounted, and a sequence number. The designation always begins with _C.

This method of naming pipes mounted automatically ensures that each pipe receives a unique name.

When you use the MNT Command to mount a pipe, follow the syntax for devicenames explained earlier in this manual.

The WMCS includes a device driver for pipes, PIPE$nnn.DSR (the nnn in the file designation is a variable corresponding to your system type, e.g., 156 for a System 150-6, etc.).

This device driver can be assigned to a pipe mounted with the MNT Command. The :pipesize= switch can be usesd to specify the capacity of the pipe. The default pipe size is 256 bytes and the maximum value you can specify is 32767.

When the vertical bar is typed on the CIP command line, CIP uses the logical name SYS$PIPE to decide which driver to mount when creating pipes.

Also, SYS$PIPE is, by default, assigned to the name PIPE$nnn.DSR.

Once a pipe is mounted, data can be read from it when:

1. A process has the pipe open for writing.

2. The pipe contains data (whether or not the pipe is open, for writing, by a process).

Chapter 11

Logical Name Assignment

Understanding this chapter is not requisite to using the CIP. However, the knowledgable user can take advantage of the features discussed in this chapter to more fully utilize the capabilities of the CIP. Therefore, this chapter is designed primarily for the experienced user.

Details on logical name assignments are provided in the descriptions of the ASSIGN and SHLOG commands. This chapter deals with logical name assignment generally.

The Utility of Logical Name Assignments

Logical name assignments can be used for the following:

1. Customizing your working environment.

2. Device and file independence.

3. Flags.

Each of these uses is discussed in the following sections.

Customizing Your Working Environment

You can use logical name assignments to tailor your working environment to your needs.

Logical name assignment

For example, if you are used to typing LIST to perform the function of the TYPE Command, you could place the following logical name assignment in your USERUP.COM file:

```
     list := type
```

Then, whenever you type LIST onto the CIP command line and strike [RETRN], the CIP recognizes LIST as a synonym for TYPE, and executes the TYPE Command.

In addition to redefining the names of existing commands, you can use logical name assignment for special-purpose commands that you create. For example, suppose that _DC0/INVENTORY.PARTS/ is one of the subdirectories in which you do much of your work. To avoid typing CD _DC0/INVENTORY.PARTS/ each time you want to go to that directory, you could type the following logical name assignment in your USERUP.COM file:

```
     home := "cd _dc0/inventory.parts/"
```

With such an assignment, you would be able to type HOME onto the CIP command line and strike [RETRN] each time you want to return to your user-account default directory.

For more complicated functions, you can even create a command file that defines a function, and then assign a Name to the character string that executes the command file.

For example, you can create a command file that executes the PASCAL compiler, and, having created the command file, you could make the following logical name assignment (in USERUP.COM if you wish):

```
     pas := @_dc0/pascal/pas.com
```

Thus PAS is all you need to type whenever you want to execute PAS.COM.

Program Independence from Devices and Filenames

Logical names allow a program to refer to a file or device without using the actual name of the file or device.

For example, when a program needs to know the name of the device on which a user is logged on, the name SYS$PROMPT allows that program to refer to the logical name table to find the actual devicename. Thus, the user can log on to any device on the system and execute that program.

Logical Names Used as Flags

Logical names can also be used as flags. For example, you might wish to establish a flag indicating that all programs should map any lowercase input characters to uppercase. The Name UL_MAP could be equated to the Equivalence YES or NO, and all programs could translate UL_MAP and then decide whether or not to map characters. Inasmuch as such a logical name can be placed in the table of logical name assignments pertaining to your user process, the assignment would be accessible to all your programs.

For example, the PROMPT Command generates a logical name assignment wherein the Name RESPONSE is equated to whatever was typed in response to the prompt. This logical name assignment remains in the table of logical name assignments for that process until the assignment is erased or replaced. Thus, other programs with access to that table can use RESPONSE as a flag.

A Note on Translation

The CIP reviews the command mnemonic, and all parameter values enclosed in single quotation marks (typed on the CIP command line), for logical name assignments, before passing the command line character string to the process.

In the case that the Name L has been assigned to the Equivalence LINK, you cannot use L as a command mnemonic for a program other than LINK.

Chapter 12

Diagnostic Messages

The diagnostic message is the means whereby the WMCS:

1. Tells you that something has gone awry.

2. Gives you as much information as possible so that you can correct the problem.

A diagnostic message is generated when:

1. The manner in which a request is made, e.g., the syntax of a command line character string, is unacceptable.

2. A user process, command, etc., requests that the WMCS do something in behalf of the process, and the WMCS is unable to do so.

3. A volume's file system, or the hardware itself, develops a problem.

4. A problem occurs within the WMCS itself.

The Format of the Messages

The following diagnostic message is typical of the format of most diagnostic messages:

```
CIP       : Creating process ddir.exe
Failed    : Status = 133.
Message   : The specified file could not be found.
```

The CIP in the foregoing diagnostic message is the name of the program to which the WMCS reports the problem.

What the display has indicated is:

: What the utility was doing when the error occurred.
: Diagnostic number
: A statement describing the error.

The first line of the diagnostic message tells you what the program was doing when the problem occurred. In other words, the CIP was endeavoring to create a process and was unable to do so.

Failed tells you the status of the request made of the WMCS. The number which follows status, in this case 133, is the diagnostic number indicating that some aspect of the process was incorrect.

The third part of the diagnostic message is the message itself, explaining why the process could not be completed. In this case, there is no file named ddir.

Appendix A of this manual contains a complete list of WMCS diagnostic messages and an explanation of each. Turn to Appendix A whenever you do not understand the 1-line explanation of a particular problem.

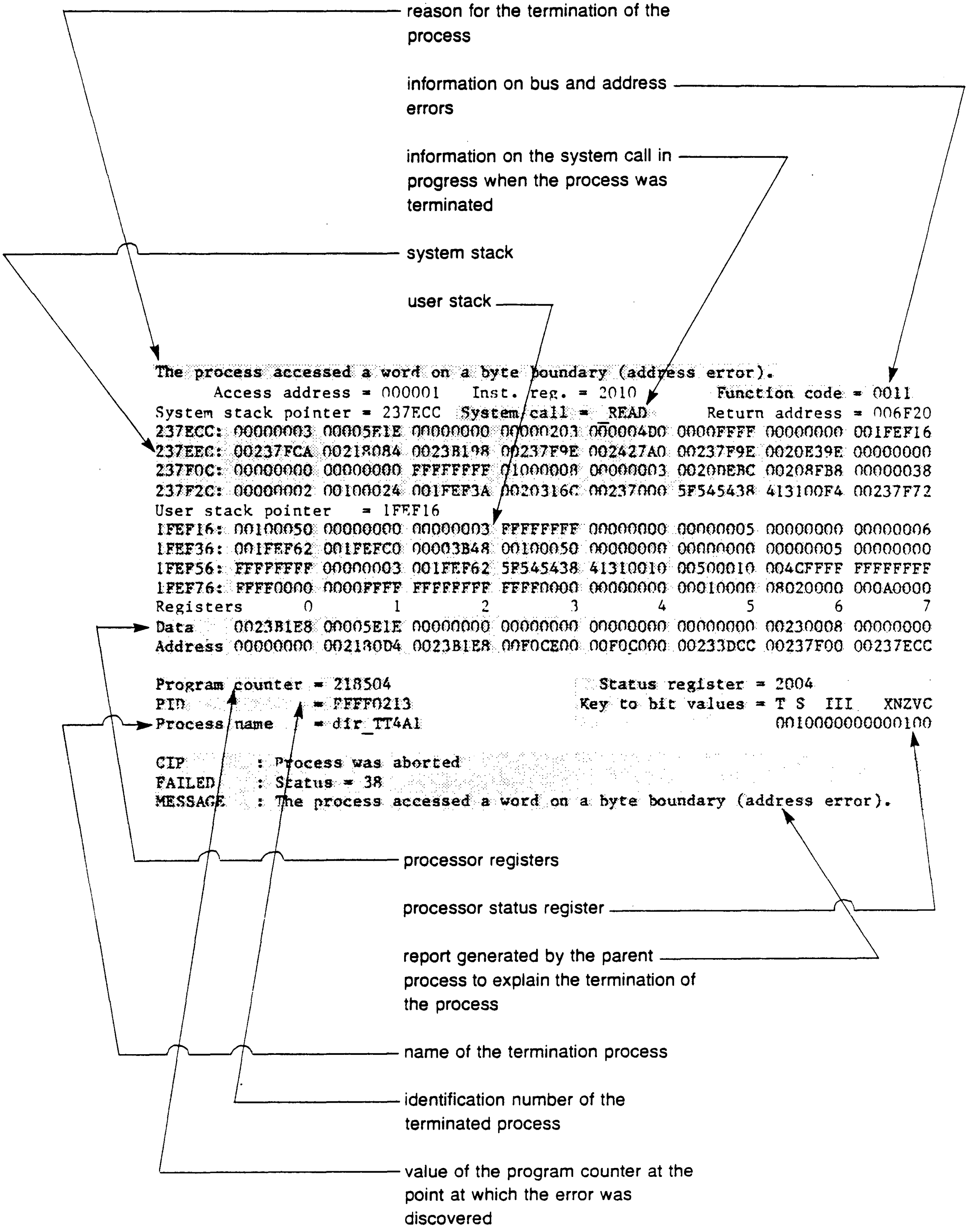
WMCS diagnostic messages are divided into four groups. The numbers in the following list are the numbers that appear on the second line of the diagnostic message, indicating the reason the command failed:

0 - 127	WMCS kernel, e.g., scheduling, creating, and controlling processes within the system.
128 - 255	File system (including KSAM), i.e., the inventory and manipulation of data.
256 - 383	Device driver errors.
384 - 4095	WMCS utility errors.

Stack Dumps

Some diagnostic messages give you more than a 1-line explanation of the problem. Errors 28 - 42 frequently display a stack dump on the terminal screen (errors 29 - 41 always result in a stack dump). Errors 28 - 42 result in termination of the process, and errors 37 - 41 are memory errors.

The following items are sample stack dumps. The information that appears on the screen as part of a stack dump is intended to help the programmer identify what went wrong. Inasmuch as the nature of the error determines the amount of information available, not every stack dump fills the screen.



reason for the termination of the process

information on bus and address errors

information on the system call in progress when the process was terminated

system stack

user stack

```
The process accessed a word on a byte boundary (address error).
      Access address = 000001    Inst. reg. = 2010        Function code = 0011
System stack pointer = 237ECC  System call = _READ      Return address = 006F20
237ECC: 00000003 00005E1E 00000000 00000203 000004D0 0000FFFF 00000000 001FEF16
237EEC: 00237FCA 00218084 0023B198 00237F9E 002427A0 00237F9E 0020E39E 00000000
237F0C: 00000000 00000000 FFFFFFFF 01000008 00000003 0020DEBC 00208FB8 00000038
237F2C: 00000002 00100024 001FEF3A 0020316C 00237000 5F545438 413100F4 00237F72
User stack pointer    = 1FEF16
1FEF16: 00100050 00000000 00000003 FFFFFFFF 00000000 00000005 00000000 00000006
1FEF36: 001FEF62 001FEFC0 00003B48 00100050 00000000 00000000 00000005 00000000
1FEF56: FFFFFFFF 00000003 001FEF62 5F545438 41310010 00500010 004CFFFF FFFFFFFF
1FEF76: FFFF0000 0000FFFF FFFFFFFF FFFF0000 00000000 00010000 08020000 000A0000
Registers       0        1        2        3        4        5        6        7
Data     0023B1E8 00005E1E 00000000 00000000 00000000 00000000 00230008 00000000
Address  00000000 002180D4 0023B1E8 00F0CE00 00F0C000 00233DCC 00237F00 00237ECC

Program counter = 218504                     Status register = 2004
PID             = FFFF0213                Key to bit values = T S  III   XNZVC
Process name    = dir_TT4A1                                   0010000000000100

CIP           : Process was aborted
FAILED        : Status = 38
MESSAGE       : The process accessed a word on a byte boundary (address error).
```

processor registers

processor status register

report generated by the parent process to explain the termination of the process

name of the termination process

identification number of the terminated process

value of the program counter at the point at which the error was discovered

Fig. 12.1 Full Stack Dump

```
The process attempted to write to a write-protected page in memory.
System stack pointer = 20E000
User stack pointer   = 1FEED0
1FEED0: 0000000A 00000001 00000000 00000000 00100006 00000000 20202020 20202020
1FEEF0: 20202020 20202020 20203933 001FEF46 000058C2 FFFF0000 00930000 00000000
1FEF10: 000A0000 00010000 00000000 00000000 00000000 00000000 00000000 00000010
1FEF30: 00060000 58870000 00000000 0000001F EFCE0000 0000001F EF5E0000 5B340010
Registers       0        1        2        3        4        5        6        7
Data     00000000 00000002 00000000 00000002 00000000 00000000 00000000 00000000
Address  00000000 001FEEFB 00000000 00000000 001FEFCE 00000000 001FEEFC 001FEED0

Program counter = 005A76                     Status register = 0000
PID             = FFFF015E                 Key to bit values = T S  III   XNZVC
Process name    = DIR_TT0A0                                    0000000000000000

CIP     : Process was aborted.
FAILED  : Status = 41
MESSAGE: The process attempted to write to a write-protected page in memory.
```

Fig 12.2 Partial Stack Dump

Appendix A

Glossary of WMCS Diagnostic Messages

0 The specified operation was performed successfully (i.e., the request made of the WMCS was successfully completed).

Diagnostic name:

KERNEL Diagnostic Messages

1 The process lacks the privileges required to perform the operation.

Each process is assigned a set of privileges, i.e., rights or prerogatives within the WMCS. When a process asks the WMCS to do something, the process must have the right or prerogative to make such a request. Otherwise, the requested operation is not performed.

Diagnostic name: **errinsufpriv**

2 The specified process is not in the system process table.

The WMCS assigns a PID (process identification number) to each process. Some requests require that the PID of the target process be specified as part of the request. This status is assigned to a request when the specified PID does not match the PID of any of the current processes, i.e., the PID could not be found in the system's list, or table, of current processes.

Diagnostic name: **errprcsnotfnd**

3 The process's buffer does not begin on a word boundary.

Several of the system calls require that buffers provided by the calling process be word aligned. This diagnostic message is sent when the WMCS expects a word aligned buffer and the buffer provided by the calling process is not word aligned.

Diagnostic name: **erroddbufaddr**

4 The logical address, for the memory requested, is invalid.

The WMCS routines that deal with the allocation and deallocation of memory require that a page address be designated. The page address must be within the logical address space of the calling process and must begin on a 4 Kbyte boundary. Otherwise, this message is specified.

Diagnostic name: **errinvadr**

5 The process requested a logical page that was already allocated.

WICAT hardware allows processes to assign pages of physical memory to addresses within the logical address space of a process. This status is assigned when a process specifies a logical address to which memory is already assigned.

Diagnostic name: **errmemalloc**

6 The process tried to affect a page in memory it did not own.

A process may not own all the pages of memory in its logical address space. For example, the pure code for a process can be shared with another process (this sharing is effected when the latter process is created). When pages are shared, neither process owns the page, and attempts to deallocate or modify the protection assigned to these pages results in this status.

Diagnostic name: **errnonowned**

7 All available memory has been allocated.

Memory is allocated for processes (process creation, explicit calls for memory allocation, automatic stack growth), as well as for the system itself (process tables, disk caches, device drivers, open files, etc.). This status is assigned to a process whose request requires more memory than is available.

Diagnostic name: **errnomemavail**

8 The specified site ID does not exist.

Each WICAT system has a site identification number that identifies the system. Several system calls allow a process to specify the ID numbers of the sites that will be affected by a request. This status is assigned when the WMCS does not recognize the specified site ID number.

Diagnostic name: **errinvsiteid**

9 The process attempted to release memory that does not exist.

The logical address space assigned to the process does not include the address of the page that the process asked the WMCS to release, i.e., the process does not have a page of memory at the address specified.

Diagnostic name: **errmemdeall**

10 An arithmetic operation produced a number longer than 32 bits.

An arithmetic overflow occurred when the WMCS tried to calculate the position (within a file) of a particular record. In other words, the calculation resulted in a number that could not be contained in 32 bits.

Diagnostic name: **errartiovflw**

11 No number was found during a search or scan for a number.

The WMCS had to convert an ASCII string to a binary number, e.g., a file version number.

This status is never returned to a user process.

Diagnostic name: **errnonumber**

12 The file type is inappropriate for the given operation.

The specified operation requires that the target file be a particular kind of file, e.g., a process asks the WMCS to install a file that is neither an image file nor a system file.

Diagnostic name: **errinapft**

13 The specified process already exists.

This diagnostic is returned to the process that tries to create an additional copy of a system process (e.g., logflush).

Diagnostic name: **errprcsfnd**

14 A negative number is not allowed for this parameter.

A process specified a negative number for a parameter where a negative number is invalid.

Diagnostic name: **errnegnumber**

15 Trap number (during _SETTRP) exceeds range of specifiable numbers.

The process issued a _SETTRP system call and asked the WMCS to define a trap whose number does not correspond to a user-defineable trap.

Diagnostic name: **errbadtrapnum**

16 The specified device is not allocated.

This diagnostic is returned when a process requests that a device be deallocated, and it is not currently allocated.

Diagnostic name: **errnotalloc**

17 Insufficient memory to automatically extend the users stack.

The user process has exceeded its user stack space. The WMCS has attempted to extend the stack space on behalf of the process, but there is no unallocated memory available. The process is terminated.

Diagnostic name: **errautostackexp**

18 The specified rotor list is empty.

This diagnostic is returned when the process attempts to allocate a device using a rotor name as the name of the device, and there are no available devices in the list associated with that rotor.

Diagnostic name: **erremptyrtrlst**

19 The process was terminated because the remote connection was lost.

This diagnostic is generated if a process is terminated because the connection from a port to a remote port is lost, and the local port is configured to delete processes associated with that port. This diagnostic is reported to the process that is terminated, and, if the process does not have an exit handler defined, it is reported to the parent process in the ccode parameter of the _CRPROC system call.

Diagnostic name: **errremoteabort**

20 No interprocess mail, in system message table, for the process.

The process issued a _GMAIL system call when no interprocess mail had been sent to the process.

Diagnostic name: **errnomail**

21 The specified file is not an image file.

Some system functions (create, mount, allocate, etc.,) are defined only for image files. For example, a process might attempt to create a process by specifying a file that is not an image file.

Diagnostic name: **errnotimfle**

22 The queue control file is being deleted at the users request.

This diagnostic is returned as a warning when the user specifies the :norestart switch on the MNT command to mount a queue, and the queue manager deleted the queue control file. The queue control file contains the list of queue entries.

Diagnostic name: **errqctdeluser**

23 The queue control file is being deleted. It may be corrupted.

This diagnostic is returned as a warning when a process mounts a queue, and the queue manager discovers that the queue control file contains erroneous data. The queue control file is deleted, losing any entries that may have been there.

Diagnostic name: **errqctdelupdat**

24 The process has already allocated floating point hardware.

This diagnostic is returned when the process attempts to map floating point hardware into its address space after it has already done so. That is, a process cannot map floating point hardware more than once.

Diagnostic name: **errhavemath**

25 The process has an undefined trap: floating point.

A floating point exception has occurred. Note that if the process had set up a floating point trap handler (using _SETTRP), the floating point exception would have caused a transfer to the floating point trap handler rather than the termination of the process.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errflpointtrap**

26 The process abort status was forced to a normal exit status.

A process can specify this diagnostic code in the result parameter of the _exproc system call to force a normal exit status (0) overriding the current process abort reason code.

Diagnostic name: **errnormalexit**

27 The process was killed by another process.

This status results when a process is terminated by another process, e.g., when a process is terminated by the KILL Command.

If the terminated process was spawned (and is terminated by a third process that called _EXPROC) this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

Diagnostic name: **errforcedexit**

28 The system clock reached the value specified for _ALARM.

A process set a duration period, i.e., an alarm, and the duration expired before the process was completed.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

Diagnostic name: **erralarmexit**

29 The process has an undefined trap: divide-by-zero.

The process attempted to divide by zero. Note that if the process had set up a divide-by-zero trap handler (using _SETTRP), the attempt to divide by zero would have caused a transfer to the trap handler rather than the termination of the process.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errzerodivtrap**

30 The process has an undefined trap: CHK Instruction.

This status results when a process is terminated because it generated a CHK trap. Note that if the process had set up a CHK trap handler (using _SETTRP), the CHK trap would have caused a transfer to the trap handler rather than the termination of the process.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errchktrap**

31 The process has an undefined trap: TRAPV Instruction.

This status results when a process is terminated because it generated a TRAPV trap. Note that if the process had set up a TRAPV trap handler (using _SETTRP), the TRAPV would have caused a transfer to the trap handler rather than the termination of the process.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errtrapvtrap**

32 The process has an undefined trap: TRACE.

The process generated a TRACE trap and was terminated. Note that if the process had set up a TRACE trap handler (using _SETTRP), the TRACE trap would have caused a transfer to the trap handler rather than the termination of the process.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errtracetrap**

33 The process has an undefined trap: 1010 Instruction.

The process was terminated because it attempted to execute a 1010 instruction. Note that if the process had set up a 1010 trap handler (using _SETTRP), the 1010 instruction would have caused a transfer to the trap handler rather than the termination of the process.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **err1010trap**

34 The process has an undefined trap: 1111 Instruction.

The process was terminated because it attempted to execute an 1111 instruction. Note that if the process had set up an 1111 trap handler (using _SETTRP), the 1111 instruction would have caused a transfer to the trap handler rather than the termination of the process.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **err1111trap**

35 The process attempted to execute a privileged instruction.

The process attempted to execute a privileged instruction while the process was in user mode. These are the privileged instructions:

```
STOP
MOVE xx,SR
AND  xx,SR
OR   xx,SR
XOR  xx,SR
MOVE Ax,USP
MOVE USP,Ax
RESET
RTE
```

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **erprivintrap**

36 The process attempted to execute an illegal instruction.

The process attempted to execute an illegal instruction, i.e., an instruction that is not recognized by the MC68000 microprocessor.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errillintrap**

37 The process accessed nonexistent physical memory (bus error).

The process generated a bus error. A bus error pertains to the system's hardware, and is any access to an address at which there is no memory or memory mapped device.

The process is also terminated, unless the bus error results from a call to _WTPMEM or _RDPMEM.

If the process was terminated (and was a spawned process), this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is written to the SYS$ERROR file assigned to the process (if the process is terminated).

Diagnostic name: **errbustrap**

38 The process accessed a word on a byte boundary (address error).

The process caused an address error. An address error is generated by the MC68000 microprocessor when when the address of a word or long word operand is odd rather than even.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **erradrtrap**

39 The process accessed nonexistent logical memory (memory violation).

The process caused a memory violation. A memory violation occurs when a process attempts to access an address in logical memory ($000000 - $1FFFFF) to which no memory has been assigned, or when a process in user mode accesses any address greater than $1FFFFF.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errnonexmem**

40 The process has a memory parity-error.

The process was terminated because it attempted to access a location in memory whose contents had been altered because some aspect of the system's hardware malfunctioned.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errmemparity**

41 The process attempted to write to a write-protected page in memory.

The process attempted to write to an address on a write-protected page of logical memory. Write-protected portions of logical memory include the following:

Pure code shared by processes.

Pages write-protected by the process that owns them.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errwriteprot**

42 A handler was not defined before a TRAP instruction was executed.

The process attempted to execute a TRAP instruction for which it had not used the _SETTRP system call to define a trap handler.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errundeftrap**

43 The WMCS does not recognize the SVC number used by the process.

The process specified an undefined SVC number in an attempt to execute a system call.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errundefsvc**

44 The process has lost Data Set Ready on a tty line it controlled.

This diagnostic is generated if a process is terminated because the connection from a port to a remote port is lost, and the local port is configured to delete processes associated with that port. This diagnostic is reported to the process that is terminated, and, if the process does not have an exit handler defined, it is reported to the parent process in the ccode parameter of the _CRPROC system call.

Diagnostic name: **errdsrloss**

45 This item is not implemented yet.

This function has not yet been implemented, but will be implemented in a future release of WMCS.

Diagnostic name: **errnotimp**

46 The spawned child has terminated.

A child process created by a surrogate process has terminated. Note that when a process is spawned by a surrogate process, the surrogate process does not go to sleep in a child wait, but continues execution. When the spawned child process terminates, rather that waking up the parent process, the O.S. kills the surrogate process with this error code, and the surrogate process's exit handler then makes a reply packet to the remote parent process.

Diagnostic name: **errspawndone**

47 The process was not allowed to log on to the remote system.

The user ID from the given site ID of the process is not allowed to log on to the desired remote system.

Diagnostic name: **errremotelogon**

48 (WMCS error) Nondelete, or critical, count is too large (overflow).

This status represents an error within the WMCS, i.e., the user process is not at fault.

Diagnostic name: **errsetdelovfl**

49 (WMCS error) Nondelete, or critical, count is less than 0 (underflow).

This status represents an error within the WMCS, i.e., the user process is not at fault.

Diagnostic name: **errclrdelovfl**

50 Supervisor's stack does not contain enough parameters (underflow).

Two stacks are associated with each process:

user - used while the process is in user mode

system - used when the process calls the WMCS

This status results when the process expects data from the system stack and the system stack is empty.

This occurs when a process is running in supervisor mode and calls the WMCS without having pushed the requisite number of parameters.

Diagnostic name: **errprevalloc**

51 User's stack does not contain enough parameters (underflow).

Two stacks are associated with each process:

user - used while the process is in user mode

. system - used when the process calls the WMCS

This status results when the process is running in user mode and calls the WMCS without having pushed the requisite number of parameters.

Diagnostic name: **errustckunfl**

52 No network virtual circuits are available for this operation.

All of the network virtual circuits have already been allocated, leaving insufficient network virtual circuits available to make a connection to a remote machine in order to complete this operation.

Diagnostic name: **errnodevavail**

53 The specified node could not be found.

The system network tables do not contain an entry corresponding to the specified node name.

Diagnostic name: **errnonodefnd**

54 The originator process has been aborted.

The remote originator of a surrogate process has been terminated. (This error will never be received by a user process. It will only be sent by the OS to a surrogate process.)

Diagnostic name: **errorigterm**

55 Remote process creation is not allowed by the remote system.

The process does not have the privilege required on the remote system in order to create a process on the remote system.

Diagnostic name: **errnoremcrproc**

56 The table ends before the specified occurrence.

The following system calls enable a process to request an item from a system table:

_GETDNAM, _GETLOG, _GETINST

When a process uses one of these system calls, the process must specify the position in the list, or table, of the item.

This status results when the list is not long enough to include the ordinal number specified, e.g., a process requests the name of the fifth device in a particular list, and the list contains only four devicenames.

Diagnostic name: **erridxrange**

57 The siteid verification failed for the specified network node.

The site ID verification process for the specified network node failed.

Diagnostic name: **errsiteinvalid**

58 The priority ratio for the scheduler is less than or equal to zero.

The priority ratio is the number of processes, at any level of priority, that will be scheduled (for processing) for each process at the next lower level of priority.

This status results when the ratio assigned (using _PRIRAT) to the scheduler is less than or equal to zero.

Diagnostic name: **errpriorratio**

59 The address, sent to an SVC, exceeds user's logical address space.

The WMCS requires that all addresses that it receives from a user process be within the logical address space assigned to the user process ($000000 through $1FFFFF). This keeps processes from affecting the WMCS and other processes.

This status results when an address exceeds the logical address space for the user process.

Note that a process operating in supervisor mode can access addresses grater than $1FFFFF.

Diagnostic name: **erraddovfl**

60 The size, sent to an SVC, is out of range.

The size parameter associated with this system call contains an unacceptable value. For instance, this diagnostic is returned to the process attempting to share a named shared memory segment, and the size specified is zero or, the size specified is not large enough to accommodate the entire memory segment.

Diagnostic name: **errsizovfl**

61 An invalid value was specified.

The value specified for the parameter is not valid.

Diagnostic name: **errinvvalue**

62 The process was killed because of a SWAPPER I/O error.

The swapper was unable to read the process from the swap file to restore it to memory and allow it to run.

Diagnostic name: **errswapio**

63 (Floating point diagnostic) Illegal instruction given to FFP board.

An illegal operation code was sent to the FFP hardware floating point board.

Diagnostic name: **errffpillinst**

64 An invalid character appears in a decimal string.

This status results when a process asks the WMCS to convert a decimal ASCII string to a binary string (for example, a file version number) and the WMCS encounters a nonnumeric character in the string.

Diagnostic name: **errinvnumchar**

65 (Floating point diagnostic) Device does not respond.

This status indicates a malfunction in the floating point hardware.

A timer is set whenever the floating point processor is assigned a task. This status is assigned when the processor does not complete the task in the allotted time, and the process is terminated. All processes in the middle of a floating point operation when this status is assigned are also terminated.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

Diagnostic name: **errmathbrdfail**

66 (Floating point diagnostic) Divide-by-zero error.

The process was terminated because it attempted to use the floating point software or hardware to divide by zero.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errdiv0**

67 (Floating point diagnostic) Number is too small.

The process was terminated because the result of the floating point operation was too small to be represented in the floating point format. In other words, the exponent that results from the normalization of a floating point operation does not fit in the exponent field of the floating point format.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errunderflo**

68 (Floating point diagnostic) Number is too large.

The process was terminated because the result of the floating point operation was too large to be represented in the floating point format. In other words, the exponent that results from the normalization of a floating point operation does not fit in the exponent field of the floating point format.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **erroverflo**

69 (Floating point diagnostic) Illegal operation.

The process was terminated because it asked the floating point hardware to perform an undefined operation, e.g., the square root of a negative number, the log of a negative number, or raising a negative number to a power.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errillegalop**

70 (Floating point diagnostic) Denormalized operand.

The process was terminated because it performed a floating point operation that produced a denormalized result.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

A process crash display, or stack dump, is also written to the SYS$ERROR file assigned to the terminated process.

Diagnostic name: **errdenormop**

71 This operation is not allowed on a SURROGATE process.

The operation attempted cannot be done to a surrogate process.

Diagnostic name: **errsurrogate**

72 A connect packet was received after the connection was made.

After a valid connection has been made by the network, a connect packet was received. (This error should not be received by a user process.)

Diagnostic name: **errdupconnect**

73 An SVC packet was received before the connect packet was received.

A connection must be established with a remote node via a connect packet before any SVC packets or special packets may be sent.

Diagnostic name: **errnoconnect**

74 The disconnect packet was not from the originator process.

Only the process that initiated a connection may terminate the connection via a disconnect packet.

Diagnostic name: **errwrngorigpid**

75 A packet was received for a local-execution-only SVC.

Certain SVCs may not be executed remotely, but only on the local machine. If a local-execution-only SVC packet type is received by a surrogate process, it will generate a reply packet with this error.

Diagnostic name: **errnotremotesvc**

76 The actual packet size is not the same as the size in the header.

The number of data bytes read by the network software for the current packet does not agree with the number of bytes that the packet header indicated were to follow.

Diagnostic name: **errbadpktsize**

77 The reply packet SVC is not the same as the request packet SVC.

The SVC number in the reply packet is not the SVC number that was sent to the remote system.

Diagnostic name: **errbadpktsvcno**

78 All available memory has been allocated on the remote system.

There is no memory available on the remote system to perform the requested operation.

Diagnostic name: **errnoremotemem**

79 The process is incompatible with the current operating system version.

Certain utility programs contain intimate knowledge about a particular version of WMCS. If a program that requires a specific version of WMCS to operate correctly detects that the version of WMCS is different than the required version, this error message is returned.

Diagnostic name: **errwrongos**

80 The specified name must not be null.

This diagnostic is returned to the process attempting to define a named shared memory segment with a null name. Null names are not allowed for named shared memory segments.

Diagnostic name: **errnamenull**

81 The specified name already exists.

This diagnostic is returned to the process attempting to define a named shared memory segment using a name that is already defined as a named shared memory segment.

Diagnostic name: **errnameexists**

82 The specified name does not exist.

This diagnostic is returned to the process attempting to share a named shared memory segment, and a segment with the specified name cannot be found.

Diagnostic name: **errnoname**

83 Process killed because of a queue restart request.

The Queue Manager was requested to restart a process, so it killed the currently running version of the process. This error is to enable special restart handling or cleanup of jobs run in a queue.

Diagnostic name: **errquerestart**

84-127 No error assigned.

Class Handler Diagnostic Messages

128 A request was not completed within the specified time.

The process specified a timeout value, i.e., a duration, for the completion of a request, and the duration expired before the requested operation was completed.

For example, a _READ request was not completed in the specified time.

Diagnostic name: **errtimeout**

129 A file's version number cannot be greater than 65535.

The process attempted to use the _CREATE or _CREATS system call to perform one or the other of these operations:

Create a file with a version number greater than 65535.

Create a file with a version number of zero in a directory that contains a file whose filename and extension match that of the file to be created, and whose version number is 65535.

Diagnostic name: **errinvvernum**

130 The specified devicename is syntactically incorrect.

At least one element in the devicename field of the specified file designation is disallowed.

Diagnostic name: **errinvdevnam**

131 The WMCS does not recognize the devicename. Is the device mounted?

The system device table, i.e., the list of mounted devices, does not contain the specified devicename.

Diagnostic name: **errundevnam**

132 The logical unit number does not correspond to an open file.

No open file has a logical unit number matching the unit number specified.

Diagnostic name: **errinvlfn**

133 The specified file could not be found.

The specified directory does not contain a file whose filename, file extension, etc., match the file designation specified.

Diagnostic name: **errfilnotfnd**

134 The specified version of the file already exists.

A file already exists whose file designation matches that of the file specified.

Diagnostic name: **errfilexists**

135 The specified file is read-locked.

The specified file is open and read-locked, i.e., it cannot be opened for read access.

Diagnostic name: **errreadlock**

136 The specified file is write-locked.

The specified file is open and write-locked, i.e., it cannot be opened for write access.

Diagnostic name: **errwritelock**

137 The specified queue does not have a default definition.

This diagnostic is returned when a process attempts to open (using the _open or the _create system call) a queue class device, and there is no default create process definition for the specified queue. (see the :printtype switch of the DSTAT command, and/or the _physop system call).

Diagnostic name: **errquenodef**

138 This edit mode requires that the record length be set to one.

The process opened, or created, a file with a record length other than one, and then (in an edit mode requiring that the file have a record length of one) attempted to read (_READ) or write to (_WRITE) the file.

Diagnostic name: **errinvreclen**

139 The specified file type is reserved for the WMCS.

The process attempted to create a kind of file whose file type is reserved for the WMCS.

Diagnostic name: **errinvfiletype**

140 The process tried to read past the logical end of a file.

The process attempted to use the _READ system call to read records that are beyond the end of the file, i.e., they do not exist.

Note that if some of the specified records exist they are read and no diagnostic message is reported, e.g., a process requests 10 records and the file contains 5; the five are read and no diagnostics message is reported.

Diagnostic name: **errreadleof**

141 The process does not have read-access to the specified file.

The process opened a file, did not request read access to that file, and then issued a system call that requires read access, e.g., _READ.

Diagnostic name: **errnoreadacc**

142 The process does not have write-access to the specified file.

The process opened a file, did not request write access, and then issued a system call that requires write access, e.g., _WRITE.

Diagnostic name: **errnowriteacc**

143 The process does not have Execute Privilege for the file.

The process cannot execute the specified file because the process does not have execute privilege for the file. Note that the owner of a file grants execute privilege to the various classes of users. The privileges granted to the class of users to which the process belongs is indicated in the file's protection mask.

This status also results when a process attempts to access a file, e.g., use the _OPEN system call, when the process does not have execute privilege for one or more of the directories along the directory path to the file. Thus, a process must have the appropriate privilege for the file, and execute privilege for all directories belonging to the file's designation.

Diagnostic name: **errnoexecpriv**

144 The process does not have Read Privilege for the file.

The process cannot open the file for read access because the process does not have read privilege for the file. Note that the owner of a file grants read privilege to the various classes of users. The privileges granted to the class of users to which the process belongs is indicated in the file's protection mask.

This status also results when a process attempts to access a file, e.g., use the _OPEN system call, when the process does not have read privilege for the directory or device containing the file.

Diagnostic name: **errnoreadpriv**

145 The process does not have Write Privilege for the file.

The process cannot open the file for write access because the process does not have write privilege for the file. Note that the owner of a file grants write privilege to the various classes of users. The privileges granted to the class of users to which the process belongs is indicated in the file's protection mask.

This status also results when a process attempts to access a file, e.g., use the _DELETE system call, when the process does not have write privilege for the directory or the device containing the file.

Diagnostic name: **errnowritepriv**

146 The process does not have Delete Privilege for the file.

The process does not have delete privilege for the specified file, and attempted one of the following operations:

Delete the file.

Open the file (with the delete-upon-closing mode-bit set).

Dismount the device.

Close the file (with the delete mode-bit set).

Note that the owner of a file grants delete privilege to the various classes of users. The privileges granted to the class of users to which the process belongs is indicated in the file's protection mask.

Diagnostic name: **errnodelpriv**

147 The specified filename is syntactically incorrect.

The filename field of the specified file designation is syntactically incorrect. For example, it may contain characters disallowed in the filename field, or it may contain too many characters, etc.

Diagnostic name: **errinvfnstr**

148 The specified directory is not a directory-type file.

While searching the directory path (the directories specified in the directory name filed of the specified file designation), the WMCS encountered a file that is not a directory.

Diagnostic name: **errinvdirfle**

149 The specified directory name is syntactically incorrect.

The directory name field of the specified file designation violates the syntax for that portion of the file designation. For example, it mnay contain characters disallowed in file designations.

Diagnostic name: **errinvdirstr**

150 The specified entry is already active.

Once an entry in the queue has begun execution, it cannot be modified, put on hold, or awakened. This diagnostic is returned to the process that attempts to perform one of those operations on an active queue entry. (see the _physop system call)

Diagnostic name: **errentryactive**

151 The WMCS cannot allocate more than 65535 sectors at a time.

The process asked the WMCS to allocate more than 65535 sectors (a very large _WRITE request).

Diagnostic name: **errinvsecreq**

152 The FCB (or the TFCB) does not correspond to its checksum.

Each File Control Block (FCB), or Tape File Control Block (TFCB) has a checksum associated with it, i.e., a sum of each of the values appearing in each field of the control block.

This status results when a process refers the WMCS to an FCB or TFCB and the WMCS finds that the checksum and the data in the control block do not correspond.

This helps the WMCS maintain the integrity of the file system on the device.

Diagnostic name: **errinvfcbcksum**

153 The specified file is open, has been marked for deletion.

The process asked the WMCS to delete an open file, e.g., a file that another process has open.

This status results as a WARNING, and when the process that has the file open closes the file, the file will be deleted.

Diagnostic name: **erropendel**

154 All available disk space has been allocated.

The process asked the WMCS to allocate space on a volume on which all available space has been allocated.

Diagnostic name: **errnospace**

155 The specified queue is closed.

This diagnostic is returned to a process which attempts to insert a new entry into a queue that is closed. A closed queue will not accept any new entries, but will continue to process all existing entries.

Diagnostic name: **errqueclosed**

156 The specified sector/block size is not supported on this device.

This status results when the sector size on a disk neither 512 nor 1024, or when the block size on a tape is greater than 4096.

Diagnostic name: **errinvsectorsz**

157 The specified entry was not found.

This diagnostic is returned to the process which attempts to access or modify an entry in a queue, and the specified entry does not exist.

Diagnostic name: **errentrynotfnd**

158 System files cannot be deleted.

The process attempted to delete one of the following system files:

/ROOTDIR/BITMAP.SYS /ROOTDIR/FCB.SYS /ROOTDIR/FCBBITMAP.SYS / ROOTDIR/ROOTDIR.DIR

Diagnostic name: **errdelfile**

159 System files cannot be renamed.

The process attempted to rename one of the following system files:

/ROOTDIR/BITMAP.SYS /ROOTDIR/FCB.SYS /ROOTDIR/FCBBITMAP.SYS / ROOTDIR/ROOTDIR.DIR

Diagnostic name: **errrenfile**

160 The device cannot be dismounted because files are still open on it.

The process attempted to dismount a device on which files are still open.

Diagnostic name: **errfilesopen**

161 The usage field in the file's FCB contains an unexpected value.

The usage field of a File Control Block (FCB) indicates the use begin made of that FCB, i.e., whether the FCB is available (unassigned), a primary FCB, or a continuation FCB.

This status results when the WMCS refers to an FCB and finds that the usage field does not indicate the usage the WMCS expected, e.g., the WMCS anticipates that the specified FCB is a primary FCB and finds that the FCB's usage field indicates that the FCB is either available or that it is a continuation FCB.

Diagnostic name: **errinvusagid**

162 The specified device was not properly configured.

This diagnostic is returned when a process attempts to mount a disk whose boot block does not contain a media description block. For disks mounted special, this diagnostic is returned as a warning. This diagnostic is not returned to the user process as an error.

Diagnostic name: **errdrnotconfig**

163 The request cannot cross machine boundaries.

The operation requested is not able to be done remotely. It must be performed on a local machine.

Diagnostic name: **errdiffmachine**

164 This device was improperly dismounted.

While mounting the specified device, the WMCS found that the device had not been properly dismounted. This can happen when the SHUTDOWN Command is not used to reboot the system.

Diagnostic name: **errimprdmnt**

165 The read request is invalid.

The process asked the WMCS to perform a task either too large or too small for the system to handle.

Diagnostic name: **errinvreadreq**

166 The request crosses a physical page boundary in memory.

Some file system operations, e.g., a fast read, require that the data buffer exist entirely within a single page of physical memory. This status results when such an operation is specified and the data buffer crosses a physical page boundary:

Diagnostic name: **errpagebndry**

167 A file cannot be renamed to another device.

The REN Command cannot be used to move a file to a device other than the device on which the file is located.

Diagnostic name: **errrendiffdev**

168 The boot block has changed since the device was mounted.

This status indicates that when the specified disk was dismounted, the data in the disk's boot block do not match the data read into memory when the disk was mounted.

Diagnostic name: **errdiffbtblk**

169 A sector(s) in the disk cache could not be written to the disk.

The WMCS has made several unsuccessful attempts to write a sector from the disk cache to the disk.

Diagnostic name: **errnowritesec**

170 Operator privilege is required in order to change a network window size.

The process must have operator privilege in order to change a network window size.

Diagnostic name: **errwindwszpriv**

171 The operation is inappropriate for physical devices in the network class.

The operation requested may only be performed by logical devices (virtual circuits), and not by physical network devices.

Diagnostic name: **errinvnetoper**

172 An error occurred in doing Huffman decompression on the network data.

Network data were compressed using the Huffman compression algorithms, and during the decompression phase an error was detected, invalidating the data that was transferred.

Diagnostic name: **errdecompress**

173 The operation is inappropriate for the device class.

Some operations can be performed only on devices in a particular class(es). For example, a process can rename a disk file, but not a TTY-class file or a tape file (the rename operation is not defined for TTY-class files and tape files).

Diagnostic name: **errinvcloper**

174 Directories do not exist on the specified device.

The target device for the specified operation does not contain directories, i.e., it is not a directory oriented device, and the specified operation requires that the target device have directories.

Diagnostic name: **errinvdirdev**

175 The specified device driver function code is disallowed.

The specified function code (_PHYSOP) is not recognized by the device driver.

Diagnostic name: **errunknowncmd**

176 The process buffer is too small for the specified operation.

The process used edit mode 2 or 4 to request a _READ, and the line that was read does not fit in the specified buffer for the user process.

Diagnostic name: **errbuftosmall**

177 The specified directory does not exist.

At least one of the directories in the directory name field of the file designation could not be found.

Diagnostic name: **errdirnotfnd**

178 The FCB.SEQ number for the file does not match the specified FCB.

The sequence number appearing on the specified File Control Block (FCB) does not correspond with one of the following (depending upon the manner in which the FCB was requested):

1. The FCB.SEQ number used to open the file, e.g., the process requests that the WMCS open the file by FCB.SEQ number.
2. The FCB.SEQ number in the directory file that contains information on the files in that directory does not match the FCB (in FCB.SYS) for the specified file. In other words, each directory file (a file with a .DIR file extension) contains a list of the files in that directory. Part of the information on that list is the FCB.SEQ number for each file in the directory. When a file designation is used to specify a file, the WMCS searches the list in the directory file to obtain the FCB.SEQ number for the specified file and then searches FCB.SYS to find the FCB

itself so that the WMCS can then find the data constituting the file on the disk. This status indicates that the FCB.SEQ number in the list in the directory file, and the FCB.SEQ number on the specified FCB in FCB.SYS, do not match.

Diagnostic name: **errinvseqnum**

179 The specified device is already mounted.

The process attempted to mount a device that is already mounted.

Diagnostic name: **errdevnamexs**

180 The WMCS does not recognize the specified device class.

The device class, specified in the request that a device be mounted, is not a recognized device class.

Diagnostic name: **errinvclass**

181 The specified volume has no valid boot block.

When the WMCS attempts to mount a disk, the WMCS reads sectors 0 and 32 on the disk (each of these sectors contains a boot block).

The WMCS determines the validity of a disk's boot block by comparing the checksum (that is part of the boot block) with the sum of the values that appear in each field of the boot block. This status is assigned when the checksum and the data in the boot block do not correspond.

When the WMCS attempts to mount a tape, the WMCS reads the first block from the tape and determines whether the data in that block correspond to the checksum. This status indicates that the data and the checksum do not correspond.

Diagnostic name: **errnobbfound**

182 The user's write request is too large to fit in the system buffers.

There is insufficient system buffer space to hold the user's write request.

Diagnostic name: **errwritetoobig**

183 The process requested more than 3964 bytes of dynamic memory.

One of the system tables is too large. This diagnostic is internal to the WMCS and is never assigned to a user process.

Diagnostic name: **errinvdmreq**

184 Not enough network buffers are available for a remote connection.

There are insufficient network buffers available to complete a remote connection.

Diagnostic name: **errnonetbufs**

185 The device class handler was not loaded when the system was booted.

The process asked the WMCS to mount a device, and the class handler for the class to which the device belongs, has not been loaded.

This status is also assigned when a process attempts to use the KSAM routines, and those routines have not been loaded.

Diagnostic name: **errnoclass**

186 The process tried to rename a directory as its own subdirectory.

The process attempted to rename a directory as a subdirectory of itself.

Diagnostic name: **errinvdirren**

187 The WMCS cannot extend the FCB file.

There is not enough free disk space to be able to extend the FCB.SYS file.

Diagnostic name: **errnoextfcbfil**

188 The specified device is already mounted, and has another name.

The process asked the WMCS to mount a device that is already mounted and has been assigned another name.

Diagnostic name: **errprevinit**

189 The WMCS does not recognize the specified edit mode.

The WMCS does not recognize the edit mode specified by the process.

Diagnostic name: **errinveditmd**

190 The specified device has already been mounted for synchronous use.

The process asked that a device be mounted as an asynchronous communication line, and the WMCS found that the device is already mounted as a synchronous communication device.

Diagnostic name: **errmntasync**

191 The specified device has already been mounted for asynchronous use.

The process asked that a device be mounted as a synchronous communication device, and the WMCS found that the device is already mounted as an asynchronous communication device.

Diagnostic name: **errmntsync**

192 The specified tape speed is not 12, 25, 30, 50, 90, 100, or 125 ips.

The tape speed parameter of the device status block is not one of the recognized speeds.

Diagnostic name: **errinvtpspeed**

193 The specified tape density is not 800, 1600, 3200, 6250, or 6400 bpi.

The only valid values for tape density are 800, 1600, 3200, 6250 or 6400 bpi. None of the values is valid for all types of tapes. Use the value that is appropriate for the type of tape being used.

Diagnostic name: **errinvtpdensity**

194 The network site ID on this machine is uninitialized.

The system variable containing the site ID must be initialized before any network operations can be performed.

Diagnostic name: **errnositeid**

195 The network nodename on this machine is uninitialized.

The system variable containing the nodename of the machine must be initialized before any network operations can be performed.

Diagnostic name: **errnonodename**

196 No error assigned.

197 The process tried to access a record (on a tape) out of sequence.

Random access is not possible on tape devices. Therefore, this status indicates that the process requested a record that is not the next record (relative to the position of the tape head).

Diagnostic name: **errbadpos**

198-199 No error assigned.

200 A directory file cannot have a version number greater than one.

The process asked the WMCS to perform an operation that would have resulted in a directory file (a file with a .DIR file extension) with a version number greater than 1. The WMCS requires that all directory files have a version number of 1.

Diagnostic name: **errdirinvver**

201 No error assigned.

202 The operation cannot be performed because a tape file is open.

The process made a request that would have resulted in more than one file being open on a tape simultaneously; this is disallowed because random file access is not supported on tapes.

Diagnostic name: **errfilopen**

203-205 No error assigned.

206 The specified skip or erase tape-function is undefined.

The process missed an unrecognized skip tape function (see _SKIP and _PHYSOP/_PHYSIO).

Diagnostic name: **errinvskpcmd**

207-209 No error assigned.

210 The specified directory cannot be deleted; it contains files.

The DEL Program assigns this status when a process attempts to delete a directory that contains files.

Diagnostic name: **errdelnempdir**

211-214 No error assigned.

215 The specified device driver is unsuitable for this device class.

An attempt was made to mount a device and assign it a device driver that is incompatible with the class to which the device belongs, e.g., mounting a disk drive and assigning it a TTY-class device driver.

Diagnostic name: **errinvdrvclass**

216 The specified file does not contain a device driver.

The file designated as the device driver does not contain a device driver, or is not a system file (the WMCS checks several fields in a file to determine whether the file contains a device driver).

Diagnostic name: **errinvdriver**

217 The value specified for a KSAM key type is undefined.

KSAM allows for multiples of signed and unsigned bytes, words, and long words as key types. This status is assigned when the specified type is not one of the types allowed (see _KCREAT).

Diagnostic name: **errbadkeytype**

218-220 No error assigned.

221 One or more of the KSAM keys is not contained in the record.

When KSAM reads a record, it checks to make sure that all key fields within that record contain the values identified in the key file. This status is assigned when one or more of those key fields contains incorrect values.

Diagnostic name: **errkeynotinrec**

222 The KSAM key definition table is larger than 3500 bytes.

The KSAM key definition table (used in _KCREAT) cannot be larger than 3500 bytes.

Diagnostic name: **errkeytablelen**

223 The specified file is not a KSAM data file.

The file, specified in a call to _KOPEN, is not a KSAM data file. Note that a file's type is determined when the file is created, and that a file created with _KCREAT is of the correct type.

Diagnostic name: **errnodatafile**

224 The specified file is not a KSAM key file.

The file, specified in a call to _KOPEN, is not a KSAM key file. Note that a file's type is determined when the file is created, and that a file created with _KCREAT is of the correct type.

Diagnostic name: **errnokeyfile**

225 The specified number of keys is less than or equal to zero.

The first word in the ktable parameter of the _KCREAT system call specifies how many keys are defined in this KSAM file. This status indicates that the value assigned to the ktable parameter is either zero or a negative number.

Diagnostic name: **errnumofkeys**

226 The specified number of segments is less than or equal to zero.

Each key in a KSAM file can consist of from 1 to 15 segments. This status indicates that, during the creation of a KSAM file) by means of _KCREAT), the number of segments (for one or more keys) is less than or equal to zero.

Diagnostic name: **errnumofsegs**

227 The record size is less than 4 bytes or greater than 65534 bytes.

KSAM allows record sizes of from 4 - 64434 bytes. This status indicates that the record size, specified in the reclen parameter of the _KCREAT system call, is not in the foregoing range.

Diagnostic name: **errrecsz**

228 A KSAM key for a word or longword key type is not word aligned.

All signed or unsigned word or long word keys in a KSAM file must be word aligned within a record. This status indicates one or more of the word or long word keys specified in a call to _KCREAT is not properly aligned.

Diagnostic name: **errsegalign**

229 The specified key length is not a multiple of the key-type length.

The _KCREAT system call assigns this status to indicate that the length of a word key-field is not a multiple of two, or that the length of a longword key-field is not a multiple of four.

Diagnostic name: **errseglen**

230 Key number is greater than or equal to the number of defined keys.

When a process creates a KSAM file, the process specifies the number of keys that each record will contain. This status indicates that, subsequent to the file's creation, a process refers to a key whose number is greater than or equal to the number of keys in each record in the file, e.g., each record in a file contains five keys (numbered 0 - 4) and a process attempts to refer to a key whose number is 5 or more (the 5 is equal to the total number of keys in the file, but does not correspond to a key number).

Diagnostic name: **errkeynotfnd**

231 This operation requires that the current key be defined.

This status indicates that the required definition was not given.

Diagnostic name: **errkeynotdef**

232 Duplicate key was attempted in a field disallowing duplicate keys.

One attribute of each key field in a KSAM file is whether or not duplicate values are allowed for the key. This status indicates that, for a key disallowing duplicate values, a process attempted to write to that key a value that is not unique (within the file) to that key.

Diagnostic name: **errnodupkey**

233 (WMCS error) A discrepancy in the KSAM code has been detected.

This status indicates that an error occurred within KSAM. The process is not directly responsible for the error.

Diagnostic name: **errksamint**

234 The specified record cannot be locked without causing a deadlock.

The WMCS cannot lock the specified records without causing a deadlock, i.e., a situation wherein each of two processes has a record the other process wants to lock of access.

Diagnostic name: **errdeadlock**

235 The specified record(s) are locked by another process.

The process attempted to access a record locked by another process.

Diagnostic name: **errreclocked**

236 This operation requires that the current record be defined.

Some KSAM operations require that the current record be defined. This status indicates that the required definition was not provided.

Diagnostic name: **errrecnotdef**

237 The process attempted to unlock a record(s) it had not locked.

The process attempted to unlock on or more records that are not locked by the calling process.

Diagnostic name: **errrlocknotl**

238 (WMCS error) A discrepancy in the KPFD linkage has been detected.

An error occurred within KSAM, an error for which the process is not directly responsible.

Diagnostic name: **errkpfdlink**

239 The key does not point to the beginning of an active data record.

A KSAM key pointer points to a record that has been deleted.

Diagnostic name: **errkeylink**

240 (WMCS error) A KSAM data-structure linkage error has been detected.

An error occurred within KSAM, an error for which the process is not directly responsible.

Diagnostic name: **errnoprocess**

241 An exact match for the specified key value was not found.

The specified key value (in a call to _KFIND) was not found in the file.

Diagnostic name: **errsrchnotfnd**

242 (WMCS error) A KSAM buffer flushing error was detected.

An error occurred within KSAM, an error for which the process is not directly responsible.

Diagnostic name: **errksamflush**

243 Key- and data-file values for a record's key do not agree.

The value of key field in the data record, and the value of the key in the key file do not match.

Diagnostic name: **errbadkeycmp**

244 (WMCS error) An error was detected during deletion of a leaf key.

An error occurred within KSAM, an error for which the process is not directly responsible.

Diagnostic name: **errdellink**

245 No error assigned.

246 One of the parameters specifies an unrecognized option.

Several KSAM system calls have an option parameter. This status indicates that the value assigned to an option parameter does not correspond with a recognized function.

Diagnostic name: **errknooption**

247 (WMCS error) A discrepancy in the KFCB linkage has been detected.

An error occurred within KSAM, an error for which the process is not directly responsible.

Diagnostic name: **errkfcblink**

248-253 No error assigned.

254 (WMCS error) A discrepancy in the Record Locking code has been detected.

An internal recording-locking error occurred. The process is not directly responsible for the error.

Diagnostic name: **errrlockint**

255 [CTRL] c terminated the process.

The process was terminated because it was the last process to access a serial port on which a [CTRL] c was received.

If the process was spawned, this status is sent to the ccode parameter of the _CRPROC system call to the parent process.

Diagnostic name: **errcontccode**

Device Driver Diagnostic Messages

256 The sector header on the disk cannot be read.

The device was unable to read the header for a disk sector.

Diagnostic name: **errbadheader**

257 The seek or rewind took too long.

This error is returned when the device reports an error during a seek or a rewind operation.

Diagnostic name: **errseekto**

258 The device cannot perform a seek.

A device check occurred, and a disk seek cannot be performed.

Diagnostic name: **errseekflt**

259 A seek did not reach the proper cylinder.

This error is reported when a disk device fails during a search operation and does not reach the specified cylinder.

Diagnostic name: **errseekwrng**

260 The data in a sector header do not match the CRC or ECC.

This error is returned when a the data in a sector header on a disk does not match the check word (CRC or ECC) that is used to verify that data.

Diagnostic name: **errheadcrc**

261 The device cannot perform a recalibration.

This error is returned when a device fault has occurred during a recalibration operation.

Diagnostic name: **errreorgflt**

262 A recalibration took too long.

This error is returned when a device fault has occurred during a recalibration operation.

Diagnostic name: **errreorgto**

263 The specified device is either off-line, or is not responding.

The device driver attempted to access the specified device and that device did not respond.

Diagnostic name: **erroffline**

264 A device error occurred during a write to the volume write fault).

This error is returned when a write fault occurs on a device.

Diagnostic name: **errwriteflt**

265 The specified device is format-protected, and cannot be formatted.

The device driver's attempt to format the device failed because the device is format-protected (this is not the same as write protection).

Diagnostic name: **errformatprot**

266 A device error occurred during a read from the volume (read fault).

This error is returned when a read fault occurs on a device.

Diagnostic name: **errreaddataflt**

267 The data on the volume do not match the CRC, ECC, or checksum.

Each sector or block on a disk or tape contains a check word (CRC, ECC or checksum) used to guarantee the validity of the data. If a sector or block is read, and the contents of the sector or block do not match the check word, this error is returned.

Diagnostic name: **errdatacrcerr**

268 The specified sector was not found on the current track.

The controller could not find the specified sector.

Diagnostic name: **errseclocate**

269 The specified device is write-protected.

The process attempted to write to a write-protected device.

This status is also assigned as a warning when a write-protected device is mounted.

Diagnostic name: **errdevwrtprot**

270 The specified sector number is too large.

If the specified sector existed, it would be beyond the end of the volume.

Diagnostic name: **errsectolarge**

271 The device received a command the device did not recognize.

This error is returned when the device controller is sent a command that it did not recognize.

Diagnostic name: **errbadcmd**

272 The device is not functioning properly (device check).

A general device check has occurred; the cause is unknown.

Diagnostic name: **errdevcheck**

273 Data were lost; the driver could not read them quickly enough.

This error is returned when a data overrun error occurs.

Diagnostic name: **errportovrun**

274 Sector headers could not be found. Is the volume formatted?

This error is returned when the disk controller did not find sector headers on the volume.

Diagnostic name: **errnotformat**

275 The specified device did not respond in the allotted time.

This error is returned when the device did not respond in the expected time. A command was sent to the device, and was not completed in a reasonable amount of time.

Diagnostic name: **errdevtimeout**

276 A read-after-write shows a discrepancy in the data.

A verification of a read-after-write was performed and the data read do not match the data written.

Diagnostic name: **errrawfailure**

277 The tape is positioned at the end of the data on the tape.

The process attempted to read data that, if they existed, would be located beyond the end of the data that do exist on the tape.

Diagnostic name: **errlogicaleot**

278 The tape is positioned at the physical end of the volume.

The process attempted to write data beyond the physical end of the tape.

Diagnostic name: **errphysicaleot**

279 The tape is positioned at the physical beginning of the volume.

The process attempted to position the tape beyond the physical beginning of the tape. For example, the process may have attempted to skip five files toward the beginning of the tape when only four files were located between the tape's position and the beginning of the tape.,

Diagnostic name: **errphysicalbot**

280 The size of the block read from the tape is larger than requested.

The process asked that a block be read from a tape, and the number of bytes in the block exceeds the number of bytes requested by the process.

Diagnostic name: **errtpoverflow**

281 A parity error was detected in the data on the tape.

This error is returned when a parity error was detected in the data being read from the tape.

Diagnostic name: **errtpparityl**

282 The device wasn't granted access to the bus in the allotted time.

DMA-type devices request access to the system bus. This status indicates that access was not granted within a reasonable time.

Diagnostic name: **errbusto**

283 A parity error was detected in the tape controller.

A parity error was detected in the dual port memory on the tape controller board. Note that this is not a parity error in the data on the tape.

Diagnostic name: **errtpparity2**

284 The specified device was improperly set up.

The device driver received a function request for a device that has not been initialized, e.g., a device driver receives a read function request before receiving a startup request.

Diagnostic name: **errnotinit**

285 The device being read was written at a different density.

The density with which the data were written on the tape does not match the data density expected by the controller.

Diagnostic name: **errinvdensity**

286 Connection to a remote computer has not been established.

This diagnostic is returned to a process which attempts to read or write to a modem control port (e.g. X.25) which has no connection established to a remote port.

Diagnostic name: **errnocallestb**

287 Connection to a remote computer has already been established.

This diagnostic is returned to a process which attempts to establish a connection on a modem control port which already has a connection.

Diagnostic name: **errcallestb**

288 The specified device was improperly set up.

This diagnostic is returned to a process which attempts to mount a port using hardware which is being used by a different driver.

Diagnostic name: **errdevinuse**

289 A deadlock error has been detected on the device.

This diagnostic is returned when a process attempts to write to a port to which no writes are possible because there are no more output buffers available, and the input buffers are full. That is, "nobody can write until somebody reads".

Diagnostic name: **erriodeadlock**

290 The X.25 channel has been reset by the network, possible data loss.

This diagnostic is returned to a process attempting to read or write to an X.25 channel which has been reset by the network. This is a warning that data may have been lost.

Diagnostic name: **errlinereset**

291 The dial request failed.

This diagnostic is returned to a process attempting to do a dial _phsop system call when the dial fails. The process can use _getdst to obtain the device status block which contains the reason code why the dial failed.

Diagnostic name: **errdialfailed**

292 The state of the BSC line disallows the specified function.

The process specified a function that, given the state of the BSC line, cannot be performed.

Diagnostic name: **errsyinvlnst**

293 The modem is not ready for communication.

The modem, attached to a synchronous line, is not in a ready state.

Diagnostic name: **errmodemnotrdy**

294 A bid was received in response to a BSC bid.

The process issued a bid request to the BSC driver and received a different bid on the response line.

Diagnostic name: **errsylnbidrcvd**

295 A NAK was received in response to a BSC bid, poll, or select.

A NAK was received on the synchronous line in response to a bid, poll, or select.

Diagnostic name: **errsynakrcvd**

296 An EOT was received on a BSC line.

An unexpected EOT was received on the synchronous line.

Diagnostic name: **errsyeotrcvd**

297 An RVI was received in response to a write on a BSC line.

The process sent a write on a BSC line and received an RVI in response.

Diagnostic name: **errsyrvircvd**

298 A disconnect sequence was received on a BSC line.

Diagnostic name: **errsydiscrcvd**

299 None of the devices, on a BSC polling list, responded.

The BSC driver polled each device specified on the polling list and none responded.

Diagnostic name: **errsypollstemp**

300 _BSCLOG's Transfer Log was invoked before Begin Logging.

The process asked the BSC driver to transfer the contents of the log buffer, but did not first use the the Begin Logging Function to initiate the log buffer.

Diagnostic name: **errsynodblog**

301 The driver transferred unverified data to the process.

The process requested data and the BSC driver sent unverified data to the process. Data are unverified when the driver has not compared the data to the check word.

Diagnostic name: **errsynoverread**

302 A conversational reply was recevied in response to a BSC write.

The BSC driver received an unexpected conversational reply, e.g., bid, in response to a write.

Diagnostic name: **errsyconvreply**

303 The last (no-verify) read did not succeed.

The BSC driver transferred unverified data to a process and then found the data to be erroneous.

Diagnostic name: **errsyprevread**

304 The last (no-wait) write did not succeed.

The calling process did not wait for the requested transmission from the BSC driver.

Diagnostic name: **errsyprevwrite**

305 Only part of the driver's transmission block was transferred.

The BSC driver assigns this status as a WARNING.

Diagnostic name: **errsypartread**

306 The BSC transmission block is larger than the driver's buffer.

The transmission block is too large for the BSC driver.

Diagnostic name: **errsybufovflw**

307 A WAK was received in response to a BSC bid, poll, or select.

The BSC driver received an unexpected WAK.

Diagnostic name: **errsywackrcvd**

308 The size of the device driver does not match its expected size.

At the front of each device driver is a length. When a device is mounted the device driver for that device is loaded into the system. If the OS gets an error reading the driver, this error is returned.

Diagnostic name: **errcantreaddsr**

309 A BSC line is no longer synchronized.

The BSC driver assigns this status when a synchronous line drops out of synchronization.

Diagnostic name: **errsydropsync**

310 _BSCPOL's parameter block is incorrect.

A syntactical error was discovered in the parameter block.

Diagnostic name: **errsyinvprmblk**

311 A value in at least one field of the devicename is disallowed.

The system's hardware does not include anything corresponding to the value appearing in the drive number field of the specified devicename, e.g., an attempt to mount _TT99 when a 99th serial port does not exist.

Diagnostic name: **errinvdrvnum**

312 The PC board for the specified device is not installed.

The driver received a bus error while attempting to access the specified device.

Diagnostic name: **errnohardware**

313 The hangup cannot take place, files are still open on the device.

No longer used.

Diagnostic name: **errdiscfilsopn**

314 The device driver does not contain the code to be downloaded.

This diagnostic is returned when a process attempts to mount a port that ιequires download code, and the download code is not present in the driver file.

Diagnostic name: **errnodwnldcode**

315 The WICOM board has been restarted and all calls were cleared.

This diagnostic is returned to each process that attempts to access a port without first closing and reopening the port following a hardware failure on the Wicom board which clears all calls.

Diagnostic name: **errboardreset**

316 The contents of the dial buffer are missing or invalid.

This diagnostic is returned to the process making a dial _physop call if the length of the dial buffer is zero or if the contents of the buffer are not appropriate for the line type.

Diagnostic name: **errinvdialbuf**

317 The driver cannot use this version of the drive type table.

There is a drive type table stored in the boot block of each disk device. One field of the drive type table is a version number. The disk device drivers are keyed to specific versions of this drive type table. If the driver reads the boot block and discovers that the version number is not a version that the driver recognizes, this diagnostic is returned.

Diagnostic name: **errbaddrtptbl**

318 The SCSI port is already busy on select.

When the hardware device port was sent a select command, it was already busy.

Diagnostic name: **errscsibusy**

319 No SCSI request after select.

After issuing a select command to the hardware device, it did not respond with a request.

Diagnostic name: **errnoscsireq**

320 The SCSI controller is in the wrong phase.

The device controller is in the wrong phase to receive the type of command received.

Diagnostic name: **errscsiphase**

321 Error detected while requesting SCSI error status.

An error occurred during the process of requesting extended error status from the device.

Diagnostic name: **errreqscsistat**

322 SCSI port hardware error.

An error occurred in the hardware port of the device.

Diagnostic name: **errscsiporthd**

323 SCSI error detected with no error status.

An error was detected in the device controller, but no error status followed to define the error.

Diagnostic name: **errscsinostat**

324 No index signal.

No index signal was detected on the device to synchronize with.

Diagnostic name: **errscsinostat**

325 No track zero.

The disk controller was unable to detect where track zero should begin.

Diagnostic name: **errnotrkzero**

326 Multiple Winchester drives selected.

More than one Winchester driver has been selected at the same time.

Diagnostic name: **errscsimanydev**

327-383 No error assigned.

Utility Diagnostic Messages

384 A character in the specified accept sequence is disallowed.

The specified accept sequence (a character string beginning with a backslash, _\) is syntactically incorrect.

Diagnostic name: **errinvacceptsq**

385 No more file designations match the specified wildcard pattern.

The list of files, matching the specified wildcard pattern, has been exhausted.

Diagnostic name: **errnomorefiles**

386 No file designations match the specified wildcard pattern.

The specified wildcard pattern does not match any files in the directories searched.

Diagnostic name: **errnofilesfnd**

387 One or more parameter value(s) is longer than 255 characters.

Neither the command line parser nor the WMCS utilities can accommodate a parameter containing more than 255 characters.

Diagnostic name: **errclparamlong**

388 There are more than nine parameters to the parameter file.

No more than nine parameters can be assigned to a parameter file. Note that this restriction does not pertain to the number of parameters to the WMCS utilities.

Diagnostic name: **errtomanyprm**

389 Too many parameter values were specified.

The number of values specified for the required and optional parameters to a particular WMCS utility does not correspond to the number of parameters expected. This status indicates that too many parameter values were specified for a particular CIP command (on the command line, in a command file, or in a parameter file).

Diagnostic name: **errtomanyreq**

390 The specified switch is not recognized.

At least one of the specified switches is not recognized, e.g., the spelling or abbreviation is incorrect.

Diagnostic name: **errclunknownsw**

391 An unacceptable value was specified for this switch.

The value assigned to a value switch is either unacceptable or unrecognizable.

Diagnostic name: **errclinvswval**

392 The abbreviation of the specified switch is ambiguous.

The command line parser could not determine which of two or more switches was intended, e.g., if a utility has :ABCD and :ABCE as switches and you type :ABC on the CIP command line, the command line parser cannot determine which of the two switches is intended.

Diagnostic name: **errclnonunqsw**

393 This switch was specified twice; the first occurrence is used.

The same switch is specified more than once on a single command line, or in a single parameter file.

Diagnostic name: **errclmultsw**

394 A required parameter was not specified in the parameter file.

There are fewer parameters in the parameter file than there are required parameters to the command.

Diagnostic name: **errclmissprm**

395 An error occurred when the process attempted to create SYS$ERROR.

The process received an error while attempting to create SYS$ERROR.

Diagnostic name: **erropensyserr**

396 The operation cannot be performed on a file of this type.

Some WMCS utilities operate on certain kinds of files, e.g., image files, directory files, data files, etc. The specified file is not of the kind required by the utility.

Diagnostic name: **errinvftype**

397 The specified directory cannot be deleted; it contains files.

The specified file is a directory file that has files in it; only empty directories can be deleted.

Diagnostic name: **errnodeldir**

398 Multiple command lines are not allowed for this operation.

This status results when the command line character string or the parameter file contains multiple command lines, and the utility does not support multiple command lines.

Diagnostic name: **errmultcmdln**

399 No such command is defined for this operation.

Diagnostic name: **errinvcmd**

400 The specified switch is not of the expected type.

Either a value switch was specified and no value was assigned, or a boolean switch has a value assigned to it.

Diagnostic name: **errinvswtype**

401 The syntax of the specified date and time is incorrect.

Either the date, time, or both the date and the time were incorrectly specified, e.g., a keyword is unrecognizable, an unacceptable delimiter was used, etc.

Diagnostic name: **errinvdate**

402 Conflicting function switches were specified.

For utilities that have function switches, at most one of the switches may be specified in any given command. This diagnostic is reported when more than one function switch was specified.

Diagnostic name: **errinvsetsw**

403 There is not enough space on the volume to accommodate the request.

This diagnostic is reported by the dinit utility when it cannot write the file system to the volume with the parameters specified. For instance, if the operator specifies a value for the :fcbsize= switch which will not fit on the volume this diagnostic is reported.

Diagnostic name: **errinsufspace**

404 The :edit= switch syntax did not match strl:str2,str3:str4, etc.

This diagnostic is reported when the value of the :edit= switch is syntactically incorrect.

Diagnostic name: **errinvedit**

405 The :protection= switch syntax did not match S:DWRE,P:DWRE, etc.

This diagnostic is reported when the value of the :protection= switch is syntactically incorrect.

Diagnostic name: **errinvprotect**

406 The UIC syntax did not match [xxxx,xxxx].

This diagnostic is reported when the specified UIC is syntactically incorrect.

Diagnostic name: **errinvuic**

407 The range specification syntax did not match n or n-m or n- .

This diagnostic is reported when the syntax of a range of numbers is syntactically incorrect.

Diagnostic name: **errinvrange**

408 The data received do not match the original data transmitted.

The USSCOPY utility reports this diagnostic when it cannot successfully send or receive a packet after the specified number of retries.

Diagnostic name: **errtransmit**

409 The remote station's response does not relate to the transmitted data.

The USSCOPY utility reports this diagnostic when it receives a packet that does not correspond to the current context. The two USSCOPY processes are not synchronized.

Diagnostic name: **errsynchronize**

410 The remote station did not respond in a reasonable amount of time.

The USSCOPY utility will report this diagnostic when it does not receive a packet from the opposite station within the specified timeout. It waits for a packet with the specified timeout for the specified number of retries before this error is reported.

Diagnostic name: **errnoresponse**

411 The specified switch is disallowed in this context.

Some utilities have combinations of switches and parameters that are mutually exclusive. When an invalid combination of switches is specified, this diagnostic is reported.

Diagnostic name: **errinapprsw**

412 The specified username does not exist.

The specified username was not found in the SHORTUAF.DAT file. Use the WHO command to obtain a list of authorized usernames.

Diagnostic name: **errinvuser**

413 Fixed-length records cannot be converted to a different length.

This diagnostic is displayed by the TCOPY utility when there is a numeric value assigned to both the :srceform= switch and the :destform= switch, and the two values are not equal. TCOPY is not capable of "reblocking" the file. That is, you cannot use TCOPY to convert fixed-length records of one record size to fixed-length records of another record size.

Diagnostic name: **errinvconvert**

414 The record size must divide evenly into the block size.

The block size specified in the TCOPY utility must be an even multiple of the specified record size. If it is not, this diagnostic appears. For instance, if the record size is 132, the block size can be any multiple of that record size (132, 264, 396, 528, 660, ...).

Diagnostic name: **errinvrecsize**

415 The :privilege= switch syntax did not match SYSTEM,SETPRIV, etc.

The :privilege= switch may be followed by any combination of the following keywords (or unique abbreviations thereof) separated by commas. If any of these syntax rules are violated, this error is reported.

setpriv	nosetpriv	system	nosystem
readphys	noreadphys	writephys	nowritephys
setprior	nosetprior	chngsuper	nochngsuper
bypass	nobypass	operator	nooperator
altuic	noaltuic	world	noworld
group	nogroup	all	none
network	nonetwork		

Diagnostic name: **errinvpriv**

416 A parameter contains a wildcard character where they are not allowed.

A wildcard character was found in a parameter where wildcards are disallowed. Possible wildcards are asterisk, *, equal sign, =, square brackets, [], and parentheses, ().

Diagnostic name: **errinvwild**

417 The specified pipe command is invalid.

This diagnostic is displayed when the command line contains an invalid usage of the pipe character, e.g., > dir |

Diagnostic name: **errinvpipecmd**

418 The syntax of the specified pattern is incorrect.

The pattern specified as a parameter to the SCAN utility is syntactically incorrect.

Diagnostic name: **errinvpattern**

419 There is not enough space in the file to accommodate the request.

This diagnostic is reported when the user requests that a file be inserted or replaced within an existing archive file and if either the specified archive file cannot contain any more files (the archive directory is full) or there is not enough space within the archive file to accommodate the new file.

Diagnostic name: **errarchfull**

420 The values in the setup file are invalid or out of range.

This diagnostic is reported when a process reads a setup file and discovers that the contents are inappropriate.

Diagnostic name: **errsetupfileinv**

421 The specified drive type was not found in the drive type file.

The utility searched the DISK.CFG file for a record corresponding to the specified drive type, and did not find one.

Diagnostic name: **errdrtypnotfnd**

422 The specified device had no drive type listed for it.

The utility checked the record in the DEVCONFIG file for the specified device. The record was found, but the drive type field was empty (unspecified).

Diagnostic name: **errnodrtype**

423 The process was terminated with an error.

This abort status code is returned by a process to its parent process to signal that the child was not successful. For example, the COMPILE utility creates a child process to perform the syntax check. If errors are found by the child process in the syntax check, then the child process returns this error to the COMPILE utility.

Diagnostic name: **errprcsfailure**

424 The lower bound of the range is greater than the upper bound.

In specifying a range, the lower bound may not be greater than the upper bound.

Diagnostic name: **errbadrange**

425 The specified range falls outside the allowable range.

The user-specified range is not within the allowable range for this operation.

Diagnostic name: **erroutrange**

426 The keys are not consecutive; a :keyN= switch has been skipped.

When specifying multiple keys for sorting, the key numbers must be consecutive.

Diagnostic name: **errskipkeysw**

427 The FIELD= modifier cannot be used with binary type fields.

The FIELD= key modifier for sorting cannot be used with binary type fields.

Diagnostic name: **errsrtfldbin**

428 The IGNORELEADING= modifier cannot be used with binary type fields.

The IGNORELEADING= key modifier for sorting cannot be used with binary type fields.

Diagnostic name: **errsrtignbin**

429 The STARTAT= modifier must be on a byte boundary.

The STARTAT= key modifier for sorting must begin on a byte boundary.

Diagnostic name: **errsrtsatbyte**

430 The ENDAT= modifier must be on a byte boundary.

The ENDAT= key modifier for sorting must begin on a byte boundary, rather than within a byte (bit field).

Diagnostic name: **errsrteatbyte**

431 The OFFSET= modifier must be on a byte boundary.

The OFFSET= key modifier for sorting must begin on a byte boundary, rather than within a byte (bit field).

Diagnostic name: **errsrtoffbyte**

432 The sort key requires the field to start on a byte boundary.

The specified sort key requires the field to begin on a byte boundary, rather than within a byte (bit field).

Diagnostic name: **errsrtposbyte**

433 The sort key requires the length to be a multiple of bytes.

The specified sort key requires the field to be a multiple of bytes, rather than bits (bit field).

Diagnostic name: **errsrtlenbyte**

434 The sum of STARTAT= + OFFSET= modifiers must be positive.

The sum of the STARTAT= and OFFSET= key modifiers for sorting must be positive.

Diagnostic name: **errsrtsatoffng**

435 The STARTAT= modifier must be a positive integer.

The value specified for the STARTAT= key modifier for sorting must be a positive integer.

Diagnostic name: **errsrtsatpos**

436 The OFFSET= modifier must be a positive integer.

The value specified for the OFFSET= key modifier for sorting must be a positive integer.

Diagnostic name: **errsrtoffpos**

437 The ENDAT= modifier must be a positive integer.

The value specified for the ENDAT= key modifier for sorting must be a positive integer.

Diagnostic name: **errsrteatpos**

438 The LENGTH= modifier must be a positive integer.

The value specified for the LENGTH= key modifier for sorting must be a positive integer.

Diagnostic name: **errsrtlenpos**

439 The FIELD= modifier must be a positive integer.

The value specified for the FIELD= key modifier for sorting must be a positive integer.

Diagnostic name: **errsrtfldpos**

440 The :RECORDLEN= switch must be a positive integer.

The value specified for the :RECORDLEN= switch for sorting must be a positive integer.

Diagnostic name: **errsrtrecpos**

441 The :MEMORY= switch must be a positive integer.

The value specified for the :MEMORY= switch for sorting must be a positive integer.

Diagnostic name: **errsrtmempos**

442 The :MAXRECORDLEN= switch must be a positive integer.

The value specified for the :MAXRECORDLEN= switch for sorting must be a positive integer.

Diagnostic name: **errsrtmaxpos**

443 A field must be at least one bit wide (STARTAT= + LENGTH= > ENDAT=).

The field for sorting defined by the STARTAT=, LENGTH=, and ENDAT= key modifiers must be at least one bit wide.

Diagnostic name: **errsrtsatoffgt**

444 The field is not big enough for the given length (LENGTH= > ENDAT=).

The field for sorting defined by the STARTAT=, LENGTH=, and ENDAT= key modifiers is not big enough.

Diagnostic name: **errsrtlengt**

445 A length must be specified.

A length must be specified for the sort field. No defaults are allowed.

Diagnostic name: **errsrtlen0**

446 The key length must be <= 32 bits for BINARY or BIT.

The length of the key for BIT or BINARY sort fields must be less than or equal to 32.

Diagnostic name: **errsrtlen32**

447 The key length must be <= 64 bits for FLOATINGPOINT or REAL.

The length of the key for FLOATINGPOINT or REAL sort fields must be less than or equal to 64.

Diagnostic name: **errsrtlen64**

448 FLOATINGPOINT or REAL must have a length of 32 or 64 bits.

The length of the key for FLOATINGPOINT or REAL sort fields must be 32 or 64

Diagnostic name: **errsrtlen3264**

449 A text file cannot have a record length greater than one (1) byte.

A text file for sorting must have a record length of one byte.

Diagnostic name: **errsrttxtlen**

450 The delimiter= modifier is required when field= is specified.

If the field= key modifier is specified, then the delimiter= key modifier must also be specified.

Diagnostic name: **errsrtdelreq**

451 The pattern is too complex.

The sort pattern specified is too complex for the SORT program.

Diagnostic name: **errscnpatcmplx**

452 The extension is not recognized.

The extension of the file is not recognized by the program. The program assumes certain file formats and/or contents based on the file extension.

Diagnostic name: **errunknownext**

453 The :attribute= switch did not match SWAPPABLE,DESENCRYPT,...

Valid values for the :attribute= switch are: SWAPPABLE,PREZEROMEM, POSTZEROMEM,DESENCRYPT,FASTENCRYPT,WATCHDOG,USER1,USER2,USER3, and USER4. No other values may be given for this switch.

Diagnostic name: **errinvattr**

454 The username/password cannot be validated.

To successfully access the public/private key file, a valid username and password must be supplied.

Diagnostic name: **errinvuserpass**

455 The data checksum is not valid.

The checksum calculated for the data does not agree with the checksum that was stored with the data.

Diagnostic name: **errinvdatacksum**

456 Error(s) occurred during assembly.

One or more errors were detected in the source files processed by WIMAC.

Diagnostic name: **errasmerr**

457 The terminal type is unsupported by this utility.

The terminal type that is assigned to this terminal is not supported by this utility.

Diagnostic name: **errunsupportterm**

458 The data read is inconsistent, invalid, or has missing bytes.

The data that was read by the utility had either inconsistent or invalid values in it, or it was detected to be missing some required data.

Diagnostic name: **errmissinvdata**

459-4095 No error assigned.

Appendix B

System Logical Name Assignments

These are the system logical name assignments that are part of your system:

1. SYS$BATCH
2. SYS$CIP
3. SYS$CLKRATE
4. SYS$DISK
5. SYS$ERROR
6. SYS$HOME
7. SYS$INPUT
8. SYS$MATH
9. SYS$MODEL
10. SYS$NODENAME
11 SYS$OUTPUT
12. SYS$PIPE
13. SYS$PRINT
14. SYS$PROMPT
15. SYS$RESULT
16. SYS$SYSNAME
17. SYS$TMP
18. SYS$USERNAME
19. WICAT$MODEL

In most cases, the WMCS assigns values to these names to give the user information about the configuration of his system. In other cases, the system manager assigns values to them in command files.

SYS$BATCH

This name is assigned to the default batch job queue in the DEVICEUP command file.

SYS$CIP

This name is assigned by the LOGON process (based on information in the UAF.DAT file) to the file containing the Command Interpreter Program.

SYS$CLKRATE

This name is assigned by the WMCS initialization routines during system boot to the speed of the clock that is driving the system.

SYS$DISK

This name is initially assigned at boot time to the device from which your system booted. It can be reassigned by the system manager if he wishes to change the source disk for WMCS executable and data files. The device designated SYS$DISK is the device on which the CIP looks for the executable image files for its programs. Diagnostic messages, terminal setup files, etc., should also be located on the designated device.

SYS$ERROR

This name is assigned when a process is created to the standard error for that process.

SYS$HOME

This name is assigned by the LOGON process (based on information in the UAF.DAT file) to the user-account default directory, or home directory.

SYS$INPUT

This name is assigned during process creation to the file from which input to the process is received.

SYS$MATH

This name is assigned by the WMCS initialization routines during system boot. When the floating point PC board is installed and operational, this name is assigned to HARDWARE. If the board is not installed, SYS$MATH is assigned to SOFTWARE.

SYS$MODEL

This name is' assigned by the WMCS initialization routines during system boot to the kind of system you have, e.g., SYSTEM156, etc.

SYS$NODENAME

This name is assigned in the LOCALUP command file to the name of the local node.

SYS$OUTPUT

This name is assigned during process creation to the file to which process output is written.

SYS$PIPE

This name is assigned in the DEVICEUP command file to the device driver to be used when mounting pipes.

System Logical Name Assignments

SYS$PRINT

This name is assigned in the DEVICEUP command file to the default printer queue.

SYS$PROMPT

This name is automatically assigned to the port at which the user logs on during the LOGON process.

SYS$RESULT

This name is assigned by the CIP to the error code returned by the last process that was executed.

SYS$SYSNAME

This name is assigned to your system in the LOCALUP command file.

SYS$TMP

This name is assigned in the DEVICEUP command file to the device where temporary files will be placed.

SYS$USERNAME

This name is assigned by the LOGON process to the username that is entered by the user.

WICAT$MODEL

This name is assigned in the LOCALUP command file to the model of your system, e.g., System 150, System 200, System 300, etc.

Appendix C

System Time

Each computer system has a system clock and a calendar clock that the system uses to produce dates and times. Whenever you log on, the system time and date appear just above the command line. For example:

```
13-dec-1983_20:30:32.89
```

There are also several places where dates and times can be specified as parameters or switch values.

There are two ways to specify system time:

1. Relative Dates

2. Absolute Dates

Relative Dates

Relative dates represent a time relative to the absolute date and time. Because these dates are relative, each begins with a plus sign or minus sign, indicating whether it is before (-) or after (+) the absolute date and time.

A good example of this type of dating is a space launch. The time of lift-off is T. This is equivalent to current time on the computer. The countdown to lift-off is T minus (-) that time. So, T minus 30 seconds is 30 seconds before the launch. This is minus time in the same sense that a relative date is minus time. A positive (+) relative time refers to the absolute date (the time of lift-off), plus the number of minutes, hours, days, etc., since lift-off.

System Time

The following format is used for relative dates:

```
+dd_hh()mm()ss()tt
```

or

```
-dd_hh()mm()ss()tt
```

In the foregoing format examples, the parentheses represent a syntactical element. The syntax for relative dates includes the minus sign, -, slash, /, colon, :, comma, and period. The following items also explain the foregoing formats:

dd is the number of days before or after the current date and time. Its value can be from 0 to 32767.

hh is the number of hours from the absolute date and time. Its value can also be from 0 to 32767.

mm is the number of minutes before or after the absolute date. Its value can be from 0 to 32767.

ss is the number of seconds before or after the absolute date and time. Its value can also be from 0 to 32767.

tt is the fraction of a second, or tick. Its value can only be from 0 to 99 inclusive.

Following are examples of relative times written in the specified format and syntax:

```
+31_12:15:30/89
-31_12/15/30:89
+31_12:15:30-89
```

The date and time are always separated by an underscore. The day (31), hour (12), minutes (15), seconds (30), and ticks (89) are all separated by either a hyphen, -, slash, /, colon, :, comma, or period.

Defaults are allowed for every field. If the year is omitted, this year is assumed. If the month and year are omitted, this month and year are assumed. And, if the day, month, and year are omitted, today, the current month, and the current year are all assumed. If the ticks are omitted they are assumed to be zero. If the seconds and ticks are omitted they are both assumed to be zero. If the ticks, seconds, and minutes are omitted, all are assumed to be zero, and so on.

Following are some examples of relative times. Each is followed by an explanation:

+0:0:30 -Represents 30 seconds from now.

-12_2/11/30/13 -Represents 12 days, 2 hours, 11 minutes, and 30.13 seconds ago (before the current time).

-1_ -Represents yesterday at this same time (one day ago). The underscore following the -1 indicates that it is a day rather than an hour.

+1_5,20/45 -Represents now plus one day, five hours, twenty minutes, and forty-five seconds.

+2 -Represents two hours from now. The +2 represents two hours rather than 2 days because there is no underscore following it.

Absolute Dates

An absolute date is not relative to another date or time. Absolute dates are composed of a date and time separated by an underscore. No spaces are allowed in the display.

The date can consist of a keyword (for example: today, yesterday, tomorrow) or the date itself, displayed in a specific syntax. Following is an example of the date syntax:

dd()mm()yy

In the foregoing format example, the parentheses represent a syntactical element. The syntax for absolute dates includes hyphen, -, slash, /, colon, :, comma, and period. The following items also explain the foregoing format:

dd is a numeral indicating the day of the month. It can be any value between 1 and 31 inclusive.

mm represents the month, which can be specified as a numeral having a value between 1 and 12, or any unique abbreviation of the month name, such as ja, jan, janu, etc.

yy represents the year as specified by a numeral. If the value of the year displayed is less than 100, it is assumed to be 1900 plus the specified year (83 refers to 1983). However, if the value of the year is greater than or equal to 100, it is taken literally.

The following are examples of an absolute date written in the specified syntax:

18/11/82
18-11:82
18:11,82
18,11,82

Notice that the separator does not need to be consistent within a given date. Each of these dates correctly represents November 18, 1982.

The day (18), month (11), and year (82) are all separated by either a hyphen, -, slash, /, colon, :, comma, or period.

NOTE: The day is always written first, not the month.

Other examples of the same date in appropriate syntax are:

18:nov:1982
18,no,82
18/11/1982

Defaults are allowed for every field. If the year is omitted, the current year is assumed. If the month and year are omitted, the current month and year are assumed. And, if the day, month, and year are omitted, the current day, month, and year are all assumed.

Following are several examples of absolute dates. Some are simply keywords and some are written in the specified syntax. Each is followed by an explanation:

today	-The current day, month, and year
yesterday	-Yesterday's day, month, and year
tomorrow	-The day, month, and year for tomorrow
2:febr:82	-February 2, 1982
3,n,83	-November 3, 1983
31/12/81	-December 31, 1981
2	-The second day of the current month and year
5-jan	-January 5th of the current year
10:7	-July 10th of the current year

The time part of the absolute date can also be either a keyword, in this case "current", or the following syntax and format:

```
hh()mm()ss()tt
```

As in the date portion, the parentheses represents a syntactical element, which can be a hyphen, -, slash, /, colon, :, comma, and period. The following items also explain the foregoing format:

hh represents a numeral for the hour of the day in 24-hour format. Its value can be from 1 to 23 inclusive.

mm represents the numeral for the minute of the hour. As such its value is from 0 to 59 inclusive.

ss stands for the numeral for seconds of the minutes. Its value is also 0 to 59 inclusive.

tt represents the numeral for a fraction of a second (tick). Its value is from 0 to 99 inclusive.

Defaults are allowed for time of absolute dates in the same way they are allowed for dates. If the ticks are omitted, zero is assumed. If the seconds and ticks are omitted, zero is assumed for each, and so on.

The following are examples of time written in the specified syntax:

```
23:59:59:99
23/59/59/99
23-59-59-99
23,59,59,99
```

Each of these correctly represents the 23rd hour, 59th minute, 59th second, and 99/100th of a second. The hour, minute, second, and ticks are all separated by a hyphen, slash, colon, comma, or period. As mentioned in relation to relative dates, the separator need not be consistent within the time. The time would be 1/100th of a second before 12:00 midnight.

Following are some sample times with explanations. These sample times are followed by examples of dates and times together, also with explanations:

current	-The current hour, minute, second, and tick
9:14:3:50	-9:14 and 3.5 seconds am
10-15/9,23	-9.23 seconds after 10:15 am
21,10	-9:10pm
6:apr:83_22,15	-10:15pm, April 6, 1983
5/1_2:3	-The fifth day of January of the current year at 2:03am
_12/12	-The current date and time at 12:12pm

NOTE: The date and time are separated by an underscore. In the last example, the 12/12 is preceded by an underscore, indicating that it is the current date, at 12:12pm. If the 12/12 began the line, was not preceded by the underscore, it would represent December 12 of the current year at time zero In each of the examples, time is preceded by an underscore, whether it also follows an enumerated date or not.

WICAT systems

Documentation Comment Form

We are constantly improving our documentation and welcome specific comments on this manual. (Please refer to page numbers, where applicable.)

Document Title: ____

Part Number: ____

Your Position:
□ Applications programmer
□ EDP manager
□ Systems analyst
□ Novice programmer
□ Operator/user
□ Technician

Questions and Comments

Briefly describe examples, illustrations, or information that you think should be added to this manual.

____

____

____

What would you delete from the manual and why?

____

____

____

What areas need greater emphasis?

____

____

____

List any terms or symbols used incorrectly.

____

____

____

First Fold

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 00028 OREM, UTAH

POSTAGE WILL BE PAID BY ADDRESSEE

WICATsystems
Attn: Technical Publications
21 South 1500 West
Orem, Utah 84058

Second Fold