AOS
Console User's Handbook

093-000150-01

For the latest enhancements, cautions, documentation changes, and other information on this product, please see the Release Notice (085-series) supplied with the software.

Ordering No. 093-000150

Revision 01, June 1979

AOS Console User's Handbook 093-000150

Revision History:

Original Release - July 1978
First Revision - June 1979

A vertical bar or an asterisk in the margin of a page indicates a substantive change or deletion, respectively, from revision 00.

Preface

The *AOS Console User's Handbook* is a capsulized version of the *AOS CLI User's Manual.* To make this handbook even handier, aside from the CLI commands that comprise the major portion of this book, we've included a summary of LINEDIT and SPEED commands, CLI exceptional condition messages, and DEBUG/DEDIT command formats.

How to Use This Handbook

By definition, a handbook should be a quick, concise, easy-to-use capsulization of a parent manual. This handbook is *not* designed to teach you all of the wonders and intricacies of the CLI. It *is* a console reference -- a reminder to complement the parent manual(s).

Before you use this console reference book, you should first familiarize yourself with the appropriate manuals describing the AOS CLI, compilers, utilities, and subsystems operating under AOS. Once you have done so, keep this book close by. It should prove to be an invaluable aid in those fretful, forgetful times.

Referrals

In writing this handbook, we have compiled information from the following manuals. If you have specific questions about any topic, you'll find detailed information in the appropriate book. If you need information about other AOS manuals, consult the *AOS Software Documentation Guide.*

AOS Command Line Interpreter User's Manual (093-000122)
AOS LINEDIT Text Editor User's Manual (093-000218)
AOS SPEED Text Editor User's Manual (093-000197)
Learning to Use Your AOS (069-000018)
How to Load and Generate Your AOS (093-000217)
AOS System Manager's Guide (093-000193)
AOS Programmer's Manual (093-000120)
AOS Operator's Guide (093-000194)

Command Conventions

We use these conventions for command formats in this manual:

COMMAND required *[optional]* ...

Where	**Means**
COMMAND	You must enter the command (or its accepted abbreviation) as shown. You may abbreviate commands and switches to the fewest number of characters required to identify them uniquely.
required	You must enter some argument (such as a filename). Sometimes, we use: $\left\{\begin{matrix}\text{required}_1\\\text{required}_2\end{matrix}\right\}$ which means you must enter *one* of the arguments. Don't enter the braces; they only set off the choice.
[optional]	You have the option of entering some argument. Don't enter the brackets; they only set off what's optional.
...	You may repeat the preceding entry or entries. The explanation will tell you exactly what you may repeat.

Additionally, we use certain symbols in special ways:

Symbol	Means
)	Press the NEW LINE or RETURN key on your terminal's keyboard.
□	Be sure to put a space here. (We use this only when we must; normally, you can see where to put spaces.)

All numbers are decimal unless we indicate otherwise; e.g., 35_8.

Finally, we usually show all examples of entries and system responses in THIS TYPEFACE. But, where we *must* clearly differentiate your entries from system responses in a dialog, we will use

THIS TYPEFACE TO SHOW YOUR ENTRY)
THIS TYPEFACE FOR THE SYSTEM RESPONSE

Continuation Lines

You can continue a command line to another input line by typing an ampersand (&) before the NEW LINE character. The CLI issues the prompt &) on each continuation line. There is no limit to the number of continuation lines that the CLI will accept.

NOTE: The ampersand is not a delimiter; therefore, you must precede the ampersand or begin the continuation line with a delimiter if one is required.

For example, the command line you would use to bind a FORTRAN IV main program and several subroutine modules is:

```
) XEQ BIND/L=PROG.LS/P=PROG.PR MAIN& )
&),SUBR1,SUBR2,SUBR3,FSYS.LB,FORT0& )
&).LB,FORT1.LB,FORT2.LB,FORT.LB,& )
&)IMPYD.LB )
```

Note that on the second input line the leading comma delimits the arguments MAIN and SUBR1, and on the third input line the comma preceding the & delimits FORT3.LB and IMPYD.LB. However, a single argument spans the second and third lines because a delimiter was not typed before & on line 2 or .LB on line 3.

CLI Templates

Certain CLI commands permit you to use templates to specify a set of filenames. These commands include DELETE, DUMP, FILESTATUS, LOAD, and MOVE. The following table defines the available CLI templates.

Character	Meaning
-	Matches any character string that does not contain a period, including the null string.
+	Matches every character string, including those containing periods and the null string.
*	Matches any single character except a period.
#	Used in place of a filename in a pathname. The number sign represents the directory immediately before it in the pathname, all inferior directories in the tree, and all contents of the inferior directories.
\	Restricts the set of filenames matched by a template. The CLI will match files except those which match the filename template following the backslash.

For a detailed explanation of CLI templates, see the *AOS Command Line Interpreter User's Manual.*

Control Characters

..OS provides control characters which give you additional control over your console. To issue a control character instruction, simultaneously depress the key labeled CTRL and one or more of the control characters listed in the ..llowing table:

Control Character(s)	Operation Performed
CTRL-O	Cancel the display of whatever information is appearing on your console. You can restart the display by typing CTRL-Q.
CTRL-S	Postpone the display of information on your console until you decide to restart it by typing CTRL-Q.
CTRL-Q	Restart the display of information on your console.
CTRL-C CTRL-A	Stop the current CLI, SPEED, or BASIC operation so that you can enter another CLI, SPEED, or BASIC command. For programs other than the CLI, SPEED, and BASIC this control sequence is ignored.
CTRL-C CTRL-B	Abort whatever program is running and return to its parent process.
CTRL-U	Erase the current CLI command line. (CTRL-U does not appear on your screen.)

For a detailed explanation of control characters see *The AOS Command Line Interpreter User's Manual.*

End of Preface

Contents

DEBUG/DEDIT Command Formats

The first section lists breakpoint commands which apply to DEBUG only. The next section lists DEBUG and DEDIT commands.

DEBUG Commands

B *[address] [;breakpoint-condition] [;breakpoint-count]*

Set a breakpoint.

?B

Display existing breakpoints.

DB address-1 *[;...address-n]*

Delete one or more breakpoints.

NOBRK

Delete all breakpoints.

?A

Display contents of accumulators 0-3.

?F *[number]*

Display contents of a floating point accumulator.

SFP expression 1; expression 2

Set floating point accumulator.

P *[breakpoint-count]*

Start user program execution.

STAB)
FILENAME? symbol-table-name)

Append a symbol table.

MODE mode-character-1 *[;...mode-character-4]*

Change display and address mode.

CLOSE

Close dialog log file.

expression *[mode character-1;...mode character-4]*=

Compute expression and display result.

SHARE)
FILENAME? library-name)

DEBUG/DEDIT a shared library.

DSTR byte-address *[;length]*

Display an ASCII string.

[address]:

Display contents of a location.

?M

Display current display modes.

mode-character-1 *[...mode-character-4]* (ESC)

Display last item with different display modes.

LLIST address-of-1st-element; *[link-offset]* *;[display-start-address]; [display-stop-address]* *;[display-condition; [terminator]* *;[maximum-chain-length]*

Display linked elements.

(CR/LF)

Display next data item.

(SHIFT N)

Display previous data item.

DISP address 1; address 2 *[;increment] [;condition]*

Display a range of data items.

MES error-code

Interpret error code.

[address;] expression

Modify contents of a location.

LOG)
FILENAME? log-filename)

Save dialog in a file.

SET variablename; expression

Set the value of a temporary variable.

NOSYM expression 1; expression 2

Suppress new symbols.

BYE

Terminate debugging/editing.

Reader, Please Note:

Throughout this handbook, we refer to this page for information about the following AOS CLI command switches. You can append these switches to any CLI command (but not necessarily to pseudo-macros or system utilities).

Command Switches

/1= $\left\{\begin{array}{l}\text{IGNORE}\\ \text{WARNING}\\ \text{ERROR}\\ \text{ABORT}\end{array}\right\}$	Set CLASS1 to the specified severity level for this command.
/2= $\left\{\begin{array}{l}\text{IGNORE}\\ \text{WARNING}\\ \text{ERROR}\\ \text{ABORT}\end{array}\right\}$	Set CLASS2 to the specified severity level for this command.
/L	Write CLI output to the current LISTFILE instead of @OUTPUT.
/L=pathname	Write CLI output to the file specified by pathname instead of @OUTPUT.
/Q	Set SQUEEZE to ON for this command.

Format

ACL pathname *[user,access]...*

Possible access types:

O	Owner access
W	Write access
A	Append access
R	Read access
E	Execute access

Purpose

Set or display the Access Control List (ACL) for a file.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/V
Display the filename with the ACL.

/K
Delete the ACL. (This denies everyone, except those with the SUPERUSER privilege, access to the file until the ACL is changed again.)

Argument Switches

None.

Examples

```
)ACL TEST.PR,JONES,WARE PROJ.-,RE)
)ACL/V TEST.PR)
```
TEST.PR JONES, WARE PROJ.-,RE

Set a new Access Control List for file TEST.PR, then display the new ACL preceded by the filename.

!ACL *Pseudo-Macro*

Format

[!ACL pathname]

Purpose

Expand to a file's Access Control List (ACL).

Macroname Switches

None.

Argument Switches

None.

Examples

```
)WRITE THE ACL FOR MYFILE IS [!ACL MYFILE])
THE ACL FOR MYFILE IS ME OWARE
)
```

The CLI evaluates the pseudo-macro [!ACL MYFILE], and displays the resulting argument list on your console.

```
)WRITE FILE SECRETS ACL IS [!ACL SECRETS])
WARNING: READ ACCESS DENIED, FILE SECRETS
FILE SECRETS ACL is
)
```

The CLI does not display the ACL for the file SECRETS because you do not have read access to the file.

Purpose

Generate a new operating system.

Referral

For a complete description of AOSGEN, see *How to Load and Generate Your Advanced Operating System.*

!ASCII *Pseudo-Macro*

Format

```
[!ASCII octal-number...]
```

The octal-number must be a positive octal integer from 1 to 377.

Purpose

Expand to characters corresponding to octal arguments.

Macroname Switches

None.

Argument Switches

None.

Example

```
)WRITE [!ASCII 207])
```

Include the ASCII code for a bell character in a WRITE command. The bell character is CTRL-G, but you cannot include CTRL-G in the WRITE command. The CLI does not permit you to use the CTRL-G key in the WRITE command. However, you can specify any possible command using the command's ASCII equivalents.

Format

ASSIGN character-device...

The character-devices include the line printer, card reader, paper tape punch, etc. If you're logged on under EXEC, you cannot ASSIGN a spooled device.

Purpose

Assign a character device for your exclusive use.

After you ASSIGN a device, you control it until you DEASSIGN it or log off the system.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Examples

```
)ASSIGN @TRA)
```

You have exclusive use of the device.

```
)DEASSIGN @TRA)
```

Format

XEQ BASIC

Purpose

Invoke the BASIC interpreter.

For more information about the BASIC utility, refer to the following manuals:

basic BASIC (093-000088)
Extended BASIC User's Manual (093-000065)

BASIC Switches

None.

Argument Switches

None.

Example

```
)XEQ BASIC)
*
        .
        .
        .
(Enter BASIC commands)
*BYE)
(Return to CLI)
)
```

Format

BIAS *[minimum number□ [maximum number]]*

Purpose

Set or display the system's bias factors.

Any process can display the bias factors, but only the initial operator process (PID2) can set them.

For more information on bias factors, see the *AOS System Manager's Guide.*

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)BIAS)
MIN: 0, MAX: 1
)
```

Displays the system's bias factor.

Format

```
XEQ BIND  { objectmodule [argument]...       }
          { objectmodule(s) commandfile/C    }
```

In the first format, objectmodule is the name of the first object that you want bound. Unless you include the /P switch, the program file will be called objectmodule.PR.

[argument]... can be any of the following:

1. Other object module
2. Shared library name
3. Unshared library name

In the second format, objectmodule(s) is as above, but commandfile/C specifies objects which are to be bound as overlays and their corresponding switches. These will be placed in the overlay file objectmodule.OL to correspond with the program file objectmodule.PR.

To bind overlays, you must build (CREATE) the command file from the binaries that you want to be overlays. For more information, see the *AOS Binder User's Manual* (093-000190).

Purpose

Build an executable program file from object files.

BIND Switches

/B

Produce an alphabetically and numerically ordered symbol file listing.

/E

Output the load map to the @OUTPUT file even though a listing is specified.

/H

Print all numbers in hexadecimal.

/I

Build a nonexecutable program file without a user status table, task control blocks, and all other system tables. Don't scan the user runtime library (URT.LB). You should use this switch to check for BIND errors, such as multiply-defined or undefined .ENT symbols.

/K=xx

Where xx is the number of tasks. This number overrides any .TSK pseudo-op statement included in the source file.

/L

Produce a listing file using current @LIST file.

/L=name

Send listing to name instead of current @LIST file.

/M=xxx

Save xxx 1K word pages of memory for shared library use.

/N

Don't scan the user runtime library (URT.LB).

BIND Switches (continued)

/O
Allow load overwrites to occur.

/P=name
Produce a program file, name.PR. Default is the first object module name.

/S
Produce a shared routine. Include this switch if you will eventually build a shared library.

/T=xxx
Decimal xxx specifies the top of the shared area. The BIND command rounds out this area to an even 1K boundary.

/Z=xxx
Decimal xxx specifies stack size for program. By default, BIND allocates 30 words (decimal).

BIND Argument Switches

/AM=xxx
This switch applies only to a right bracket in an overlay specification. It sets a total overlay area to xxx basic areas.

/B
Bind the shared library into the shared area. This switch applies only to a shared library.

/C
This file contains the objects to be bound as overlays.

/D
Bind nonshared code into the nonshared data area.

BIND Argument Switches (continued)

/H
Bind nonshared code in this module into the shared code area. If you append this switch to the name of a nonshared library, records extracted from the library will be bound into the shared area.

/R
Issue a warning if any code in this module is not position-independent.

/U
Write local symbols from this module to the symbol file. You cannot use this switch unless you also specified /U to the Macroassembler (assuming, of course, that this .OB was produced by the Macroassembler).

name/V=xxx
Assign value xxx to the accumulating symbol, name, which was defined by pseudo-op .ASYM.

name/X
Used in conjunction with the /B argument switch. Do not bind the shared library routine from a library specified by argument switch /B into the program.

xxx/Z
Set the ZREL base to octal xxx. If the current ZREL base exceeds xxx, then the system ignores this switch.

Example

```
)XEQ BIND/L=LFILE/100/Z MYPROG MYLIB)
)
```

Bind two objects, MYPROG and MYLIB, into program file MYPROG.PR. Page zero (ZREL) code will start at location 100_8. The listing goes to disk file LFILE.

BLOCK *Command*

Format

$$\text{BLOCK}\begin{Bmatrix}\text{username:procname}\\ \text{process ID}\end{Bmatrix}$$

You must supply either the procname or the process ID. The process you want to be blocked must be an inferior process unless you have the SUPERPROCESS privilege. The procname must be a full process name. (See the *AOS Programmer's Manual* for a description of process names.)

Purpose

Block an inferior process.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Examples

```
)BLOCK 19)
)
```

Block the process with PID 19.

```
)BLOCK SMITH: PROG1)
)
```

Block the process named PROG1.

Format

BYE *[arguments]*

Purpose

Terminate the CLI at your console.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

If you include any of the following switches, the CLI will return and signal the specified severity level. If you include any arguments to the command, the arguments will be returned in a string to the calling process.

/WARNING
/ERROR
/ABORT

Furthermore, if you issue the BYE command when you have sons, the CLI outputs the message:

YOU HAVE SONS. DO YOU WANT TO TERMINATE?

and waits for a YES answer before terminating. If you answer NO, the CLI does not terminate.

Argument Switches

None.

Example

```
)BYE)
AOS CLI TERMINATION 01-OCT-77 13:00:19
```

Terminate the CLI. If the CLI's father was EXEC, you are logged off the system.

Format

CHAIN pathname *[arguments-to-new-program]*

Purpose

Overwrite your CLI with the program named in pathname, and transfer CPU control to its entry point.

WARNING: Chaining overwrites your CLI in main memory. The CLI will not return unless the chained program invokes it using the system call, ?CHAIN.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/D
Enter the Debugger.

Argument Switches

As needed by the new program.

Example

```
)CHAIN MYPROG)
```

Load MYPROG into main memory and begin execution at its entry point. Do not create a new process, simply change the process's program.

Format

CHARACTERISTICS *[device]...*

Purpose

Set or display the device characteristics for a character device.

Device characteristics control the way a device interprets input or sends output. Once you set the device characteristics, they will remain in effect *until you either change them or log off the system.* You can issue successive CHARACTERISTICS commands.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Use one of the first five switches to identify your console.

/HARDCOPY
Hardcopy terminals.

/4010I
DGC Model 4010I.

/6012
DGC Model 6012.

/605x
DGC Model 6052 or 6053.

/CRT4
Other CRTs.

Command Switches (continued)

/LPP=n
Lines per page, in decimal.

/CPL=n
Characters per line, in decimal.

/ON
Set the bit in the device characteristics words for each of the command switches that follow. This bit remains set until you issue a /OFF switch or a delimiter. (NOTE: This bit is automatically set unless you include the /OFF switch. Therefore, this switch is optional.)

/OFF
Clears the bit in the device characteristics words for each of the command switches that follow. To reset the bit for any command switch, you must include the /ON switch.

/EBO
If you want echoing to occur on your console, you must set /EBO or /EB1. This will cause the system to echo control characters such as ↑A, ↑B, etc. and echo ESC as $. For more information see ?GCHR in the *AOS Programmer's Manual.*

/EPI
Accept only even parity on input; if this switch is off, accept any parity on input.

/EOL
Do not output a NEW LINE if CPL length is exceeded on output.

/ESC
ESC character produces ↑C↑A interrupt.

/FF
Output a form feed on open.

CHARACTERISTICS (continued)

Command Switches (continued)

/LT
Output 60 (decimal) nulls on open and close.

/MOD
Device is on a modem interface.

/NAS
Set non-ANSI standard bit.

/NRM
Do not allow this console to receive SEND messages.

/OTT
On input, convert octal 175 and 176 to octal 33.

/PBN
(Card readers only)
Packed format on binary read, 4 columns are put in 3 words. If you don't include this switch, columns are right-justified in memory.

/PM
Page mode: On output, write LPP lines per page, then suspend output until you type CTRL-Q.

/RAC
Send two rubouts after each NEW LINE and carriage return.

/RAF
Send 21 (decimal) rubouts after each form feed.

Command Switches (continued)

/RAT
Send two rubouts after each tab (CTRL-I).

/SFF
Simulate form feed.

/SPO
Output characters in even parity; if this switch is off, output characters as sent by program.

/ST
Simulate tab stop every eighth column.

/TO
Enable time-outs.

/TSP
(Card readers only)
Include trailing blanks; if this switch is off, trailing blanks are suppressed.

/UCO
On output, convert lowercase to uppercase.

/ULC
On input, accept both upper- and lowercase; if this switch is off, lowercase input is converted to uppercase.

/WRP
Hardware generates NEW LINE on line-too-long.

Argument Switches

None.

Examples

```
)CHARACTERISTICS)
/HARDCOPY/LPP=66/CPL=80
/ON/ST/EB0/SFF/RAF/RAT/RAC/NAS
/OFF/EPI/SPO/OTT/FOL/EBI/ULC/MOD/PM
/NRM/TO/TSP/PBN/ESC/WPP)
```

Display the characteristics of the console, which in this case happens to be a hardcopy terminal.

```
)CHARACTERISTICS/LPP=24)
```

Set the number of lines-per-page to 24.

```
)CHARACTERISTICS/PM/OFF/EPI)
```

Set page mode ON and accept both even and odd parity on subsequent input to the console.

Format

CHECKTERMS

Purpose

Check for the termination of a son process.

If the process has terminated abnormally (console interrupt, trap, etc.), the CLI outputs an appropriate message.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Examples

```
)PROCESS PROG1)
PID: 14
)TERMINATE 14)
)CHECKTERMS)
PROCESS TERMINATION,PID:14
*ABORT*
TERMINATED BY A SUPERIOR PROCESS
)
```

The first command creates a subordinate swappable process with program PROG1. The second command terminates process 14. The CHECKTERMS command checks for PID 14's termination.

Format

CLASS1 reaction-level

Purpose

Set or display the current reaction level for a CLASS1 exceptional condition.

The reaction levels are:

IGNORE
No message is displayed. The CLI continues to process your input.

WARNING
A warning message is displayed and the CLI continues to process your input.

ERROR
The CLI displays an error message and discards the input that is in the command buffer at the time the error is encountered.

ABORT
Your process terminates at once.

NOTE: When you log on, the default setting for a CLASS1 error is ERROR. In batch, CLASS1 is set to ABORT by default.

Command Switches

/L, /L=pathname
See CLI Commands tab.

Argument Switches

None.

Example

```
)CLASS1)
ERROR
)CLASS1 ABORT)
```

First, display the current CLASS1 setting, then change it to ABORT.

Format

CLASS2 reaction-level

Purpose

Set or display the current reaction level for a CLASS2 exceptional condition.

The reaction levels are:

IGNORE
No message is displayed. The CLI continues to process your input.

WARNING
A warning message is displayed and the CLI continues to process your input.

ERROR
The CLI displays an error message and discards the input that is in the command buffer at the time the error is encountered.

ABORT
Your process terminates at once.

NOTE: When you log on, the default setting for CLASS2 is WARNING.

Command Switches

/L,/L=pathname
See CLI Commands tab.

Argument Switches

None.

Example

```
)CLASS2)
WARNING
)CLASS2 IGNORE)
```

First, display the current CLASS2 setting, then change it to IGNORE.

Format

COBOL *[switches]* filename *[listingfile/L] [objectfile/R]*

Where:

filename	is a filename that specifies the source program file you want compiled.
listingfile	specifies the file or device to which you want the listing file output. It may be the console (@OUTPUT), the line printer (@LIST), or a disk or tape file.
objectfile	is a filename that specifies the name you want assigned to the object file the compiler produces. If you do not supply a name, the compiler uses filename with the extension .OB.

Purpose

Compile a COBOL source file.

For a complete discussion of the COBOL programming language and the CLI COBOL command line, see the *COBOL Reference Manual* (093-000180).

Compiler Switches

Global switches give the compiler information about the nature of your source file, and instruct it about the kind of output you want it to produce. The following list contains all the options available for a COBOL compilation.

Compiler Switches (continued)

/A	Produce an address map of the relative locations of the Procedure Division lines.
/C	Source is in card format. If you omit this switch, the compiler assumes the source is in text format.
/D	Compile debug lines and load the interactive debugger. Use this switch if your file includes debug lines, and if you want COBOL to load the code for the debugger along with your source file code.
/E	Compile language extensions. Use this switch if you want octal values produced for alphanumeric literals (this conflicts with ANSI Standard COBOL features).
/G	List the generated machine code. (This switch overrides the /A switch.)
/M	List a map of data and procedure storage in the object file.
/P	Do not generate an object file.
/Q	Do not compile; simply scan the source file code and produce a cross-reference table.
/S	List compilation statistics (the number of lines, the speed of compilation, etc.).
/W	Suppress warning messages.
/X	Include a cross-reference table in the listing.

Argument Switches

If you designate a *listingfile* or an *objectfile,* you must append the appropriate local switches (/L or /R). The filenames may appear in any order. Append any global switches you use to the command word and any local switches to the appropriate filename. These switches may also appear in any order.

/L List the source code at *listingfile.* If you did not name a list file, @LIST is assumed.

/R List the object file at *objectfile.* If you did not name an object file, the compiler uses the source filename with the extension .OB.

Example

The following command line calls the COBOL compiler to compile a file named FILE1:

```
)COBOL/L/X/W FILE1 FILE1.LS/L)
```

This command will compile the source file FILE1, and produce an object file named FILE1.OB (default name). The listing file FILE1.LS will contain source listing (/L), a cross-reference table (/X), and error messages (automatically). The compiler will suppress warning messages (/W).

Format

CONTROL ipcname argument...

Purpose

Send a control message to a process.

The arguments are a message string sent as an Interprocess Communication to the process being controlled. Although this command is normally used by system operators, you can use it to control your own programs if you've written the program to receive IPCs.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/I
Messages follow on successive lines from @INPUT. Each line is sent as a separate IPC. The messages end when a line containing a single) is typed.

/M
The messages are contained in this macro file. Each line of the macro is sent as a separate IPC. The macro file ends on a line containing a single).

Argument Switches

None.

Examples

```
)CONTROL @SPOOL RESTART @LPA)
)
```

The operator directs the spooler to restart output on the line printer.

```
)CONTROL @EXEC ENABLE @CON1)
)
```

The operator directs the EXEC process to enable @CON1.

Format

XEQ CONVERT pathname

Purpose

Convert an RDOS .RB file to an AOS .OB file.

RDOS is another Data General operating system which supports two different binary modules - an .OB type which is compatible with AOS, and an .RB type, which is not.

The CONVERT utility converts an RDOS.RB (relocatable binary) file to an AOS object file. The command line takes one argument, the input pathname (you can omit the .RB extension). The RDOS file is not modified and the AOS object file is created with the same name but with the .OB extension.

Convert Switches

None.

Argument Switches

None.

Example

```
)XEQ CONVERT PLUS24)
PLUS24.RB
)
```

Produce an AOS object file named PLUS24.OB from an RDOS object file named PLUS24.RB in the working directory. The CONVERT program displays the message PLUS24.RB when it opens the input file.

Format

COPY destination-file source-file...

Purpose

Copy one or more files to a destination file.

If the destination file does not already exist, then the COPY command creates the destination file using the first source file's specifications. If the first source file is a disk, then the destination file will have the same specifications as the disk. If the first source file is a peripheral device, the destination file's default specifications are as follows:

File Type	User Data File.
Record Type	Unspecified. You must specify record type when you open the file.
Control Parameters	None.
Element Size	512 bytes.
Maximum Index Levels	3
Time Block	Time of creation, time of last access, and time of last modification are set to the current time.

If the destination file already exists, you must use either the /A or /D command switch.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/A
Append new data to existing data in destination file.

/B
Binary mode (for character devices) - no interpretation or translation of special characters.

/D
Delete destination file, if it exists, and recreate a destination file using the specifications of the old destination file.

/IMTRSIZE = record size in bytes
Control the magnetic tape record size for input files.

/OMTRSIZE = record size in bytes
Control the magnetic tape record size for output files.

Argument Switches

None.

Examples

```
)COPY OUTPUTFILE FILEA)
)
```

Copy FILEA to OUTPUTFILE. Create OUTPUTFILE using FILEA's specifications.

```
)COPY/A TESTALL TEST1 TEST2 TEST3)
)
```

Append TEST1, TEST2, and TEST3 to the end of TESTALL.

```
)COPY TESTA @MTA0:0)
)
```

Copy file on @MTA0:0 to TESTA. Use default specifications for creating TESTA.

Format

CREATE pathname *[resolution-pathname]*

ʼurpose

Create a file.

If you omit command switches, the system creates a text file which you can use for text or source code. When you create a ʼink entry for a file, the first argument is the linkname you'll use to access the resolution file, and the second is the *resolution-pathname.*

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/DATASENSITIVE
Create the file with data sensitive record format.

/DIRECTORY
Create a directory. If you also use the /MAXSIZE= switch, the system creates a directory with the specified maximum size (control point directory).

/DYNAMIC
Create the file with dynamic record format.

/ELEMENTSIZE=n
Set the file element size to the value specified by n.

/FIXED=n
Create the file with the specified fixed-length format.

ʼHASHFRAMESIZE=n
Set the hash frame size for this directory or control point directory.

Command Switches (continued)

/I
Take the contents of the file from subsequent lines of the @INPUT file. The last must contain a single).

/INDEXLEVELS=n
Set the maximum number of index levels to the value specified by n.

/LINK
Create a link, named in pathname, to the *resolution-pathname* specified as the second argument.

/M
Take the contents of the file from subsequent lines of the current macro body. The last line of the macro must contain a single).

/MAXSIZE=n
Set the maximum size for a control point.

/TYPE=type
Select all files of the specified type where type can be in the form:

XXX	3-letter mnemonic.
n	decimal number (0-255).
m-n	decimal numbers (0-255) which define a range of file types.
\n	exclude decimal number (0-255).
\m-n	exclude decimal numbers (0-255) which define a range of file types.

You can use more than one /TYPE= switch in a command line.

/VARIABLE
Create the file with variable record format.

Argument Switches

None.

Examples

```
)CREATE/I PROG.FR)
))DIMENSION ARRAY (100))
       .
       .
       .
))))
)
```

Create a FORTRAN IV source file named PROG.FR.

```
)CREATE/DIRECTORY PROJECT1)
)
```

Create a directory file named PROJECT1.

CURRENT *Command*

Format

CURRENT

Purpose

Display the current CLI environment's settings.

Command Switches

/1,/2,/L,/L=pathname,/Q
 See CLI Commands tab.

Argument Switches

None.

Example

```
)CURRENT)
LEVEL            0
SUPERPROCESS     OFF
SUPERUSER        OFF
SCREENEDIT       ON
SQUEEZE          OFF
CLASS1           ERROR
CLASS2           WARNING
LISTFILE         @LIST
DATAFILE         @DATA
DIRECTORY        :UDD:MIKE
SEARCHLIST       :UDD:MIKE,:PER,:UTIL
STRING
PROMPT
)
```

Display the current CLI environment's settings.

Format

DATAFILE *[pathname]*

Purpose

Set or display the current DATAFILE pathname.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/G
Set the filename to @DATA (no arguments allowed).

/K
Set to null string (no arguments allowed).

/P
Set DATA filename to previous environment's DATAFILE (no arguments allowed).

Argument Switches

None.

Example

```
)DATAFILE)
@DATA
)DATAFILE MYFILE)
```

First, display the current DATAFILE, then set it to MYFILE.

DATE *Command*

Format

DATE *[date]*

The date can be set only by the initial CLI process (PID2). You can set the date using either of the following formats:

```
10 01 77
01-OCTOBER-77
```

Purpose

Set or display the current system date.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)DATE 10 01 77)
)DATE)
01-OCT-77
```

Set and display date.

Format

[!DATE]

You may not include arguments to this pseudo-macro.

Purpose

Expand to the current system date.

Macroname Switches

None.

Argument Switches

None.

Example

```
)WRITE TODAY IS [!DATE])
TODAY IS 24-FEB-79.
)
```

This gives you the current system date.

DEASSIGN *Command*

Format

DEASSIGN character-device...

You may use templates in the character-device argument.

Purpose

Deassign a previously ASSIGNed character device.

After you ASSIGN a device, you control it until you either DEASSIGN it or log off the system.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)DEASSIGN @TRA1)
```

Deassign the alternate paper tape reader, which you assigned with a previous ASSIGN command.

Format

DEBUG pathname *[arguments-to-program file]*

Purpose

Enter the Debugger and execute a specified program.

This command creates a subordinate process which executes the program named in pathname (which must be a program file). The new program starts in the Debugger. A summary of DEBUG commands appears later in this manual. See DEBUG/DEDIT commands.

For more information on DEBUG, see the *AOS Debugger and Disk File Editor User's Manual* (093-000195).

Command Switches

/I
Create input for the program from @INPUT. The last line of input must contain a single).

/S
Returns the termination message to string.

Argument Switches

As needed by the new program.

Example

```
)DEBUG MYPROGRAM)
AOS USER DEBUGGER, REV xx
# 0=000000 # 1=000000 # 2=000000  # 3=000000
+
      .
      .
      .
+BYE)

)
```

Enter the Debugger and execute MYPROGRAM.

You should always enter the Debugger *before* any execution.

Format

[!DECIMAL octalnumber]

The number must be a positive octal integer in the range from 0 to 37,777,777,777. The results will be in the range from 0 to 4,294,967,295.

Purpose

Convert an octal number to decimal.

Macroname Switches

None.

Argument Switches

None.

Example

```
)WRITE 112 OCTAL = [!DECIMAL 112] DECIMAL)
112 OCTAL = 74 DECIMAL
)
```

Convert 112 octal to 74 decimal.

Format

```
XEQ DEDIT pathname
```

Purpose

Invoke the Disk File Editor (DEDIT) utility, which allows you to examine and edit the contents of disk file locations.

DEDIT is identical to the Debugger except that it cannot set breakpoints or examine accumulators. A summary of DEDIT commands appears later in this manual. See DEBUG/DEDIT commands.

For more information on DEDIT, see the *AOS Debugger and Disk File Editor User's Manual.* (093-000195).

DEDIT Switches

/I=pathname
: Take DEDIT commands from pathname. This switch lets you build a file of DEDIT commands and execute it with a single CLI command. The file must end with a BYE command.

/L=pathname
: Save all DEDIT commands in a file identified by pathname.

/S=pathname
: Include the symbol table file identified by pathname.

Argument Switches

None.

Examples

```
)XEQ DEDIT/S=MYFILE.ST DIR1:MYFILE.PR)
+ START: 001762
     .
     .
+ BYE)
)
```

This sequence executes DEDIT on user program MYFILE in directory DIR1. DEDIT returns its prompt (+) and accepts editing commands. BYE returns control to the CLI.

DEFACL *Command*

Format

DEFACL *[username,access]...*

Purpose

Set or display the default Access Control List (ACL).

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/D
Return to the system default ACL.

/K
Set the default ACL to no ACL. (This denies everyone except SUPERUSERs access to your files until the ACL is changed again.)

Argument Switches

None.

Examples

```
)DEFACL)
SMITH,OWARE
)DEFACL SMITH,R)
)DEFACL)
SMITH,R
)DEFACL/D)
)DEFACL)
SMITH,OWARE
)
```

The first command displays the current default ACL, which happens to be the system default ACL. The second command changes the default ACL to READ access only. Next, the new ACL is displayed, then the default ACL is changed back to the system default ACL using the /D switch. Finally, the current default ACL is displayed once again.

Format

[!DEFACL]

Purpose

Expand to the current user default Access Control List.

Macroname Switches

None.

Argument Switches

None.

Examples

```
)WRITE THE CURRENT USER DEFAULT ACL IS
   [!DEFACL])
```
THE CURRENT USER DEFAULT IS SMITH,OWARE
```
)
```
The CLI first evaluates the pseudo-macro [!DEFACL], then writes the resulting argument list on the console.

Format

DELETE pathname...

You may use templates in pathname arguments.

Purpose

Delete one or more files.

Deleting a directory deletes all files in that directory. You cannot delete a directory that contains inferior directories.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/C
Confirm each deletion. The CLI displays each filename and waits for you to confirm the deletion. Enter Y to delete the file, N or NEW LINE to retain the file.

/V
Verify each deletion on the console.

Argument Switches

None.

Examples

```
)DELETE/V ADAM)
DELETED ADAM

)DELETE/C TEST.-)
=TEST.01? Y)
=TEST.02? Y)
=TEST.03? )
FILE NOT DELETED
)
```

In the first example, we deleted and verified the deletion of file ADAM. In the second example, we deleted TEST.01 and TEST.02 but we retained TEST.03.

Format

DGL *[switches]* input file *[binary-file/B][list-file/L]*
[error-file/E][option-list/O][code-option/C]

Purpose

Compile a DG/L™ source file.

For a complete description of the DG/L programming language and the CLI DGL command line, see the following manuals:

DG/L™ Runtime Library Reference Manual (093-000159)
DG/L™ Reference Manual (093-000229)

DGL Switches

/B
Produce a brief output listing (source text and storage map only).

/C
Check syntax of source text, but don't generate output code. Semantics are not checked. This option takes less time than a full compilation, and is therefore useful in early debugging of a program.

/E
Directs the compiler to check all errors, but generate no code. If errors exist, a cross-reference and a listing of errors is produced.

DGL Switches (continued)

/F

Nonfloating point program: if your program contains no floating-point operations, you can declare this to the compiler using the /F switch. Compilation in this mode will generate object code with no floating-point instructions, making the code usable on a machine without floating-point hardware. If floating-point operations or real numbers are used in a program compiled under /F, errors will be generated. The DG/L initializer checks for the existence of the FPU when this switch is omitted.

/H

For NOVA® code only. Indicates that the target machine has hardware multiply/divide instructions (otherwise, DG/L will use software multiply/divide).

/L

Produce a listing. When compiling file name.DG, /L directs the listing to @LIST under AOS.

/N

This switch tells the compiler not to generate object code, but otherwise to proceed with all compilation phases.

/O

Restricts global optimization, thus generating less efficient but sometimes more readable code. This is sometimes useful in debugging.

/P

Directs the compiler to assume that the correct number of arguments will always be passed to all external procedures. If you are sure that this will be the case, /P compilation will eliminate the need for many unneccesary runtime checks.

/R
Forces all integer division within subexpressions to be done using floating-point arithmetic. This increases accuracy by reducing rounding and truncation errors.

/S
Directs the compiler to generate code for full subscript checking. Normally, this should be used only while debugging, since an object program without subscript checking runs faster and requires less memory space.

/T
Generates code for string overflow checking. If a program compiled under /T attempts to store before the first character or after the last character of a string, a nonfatal error is generated.

/W
Directs the compiler to produce warning messages. A word of warning: this option can produce voluminous output! These messages come in two basic classes: the first warns you when the compiler makes a necessary modification to your program; e.g., if you tried to declare a REAL (15) number, the compiler would process it as a REAL (4). The second class informs you of potentially dangerous constructs in your program, such as passing a constant parameter to a procedure by reference, risking destruction of the constant.

/X
Generates a full cross-reference table, including constant references. Without this switch, only variables will be cross-referenced.

/Z
Declares to the compiler that you have placed all EXTERNAL integers in page zero of memory. If this is the case, shorter object code can be generated.

DGL Switches (continued)

/BIN=name
Directs the object file (name.OB) to file name. This is the same as the /B argument switch.

/CODE=[N,E,A]
Specifies the code generation option. This is the same as the /C argument switch.

/ERR=name
Directs the error listing to file name. This is the same as the /E argument switch.

/FIL=name
Compiles the file name or name.DG. This is the same as a local name with no switches.

/OPT=letter/digit string
Performs conditional compilation using the specified string. This is the same as the /O argument switch.

Argument Switches

/B
Specifies the file to receive binary object code.

/C
Specifies the code generation option. A /C is a legal combination for AOS. If you omit this switch, DG/L will generate code for the current environment.

/E
Specifies the file to receive error messages.

/L
Specifies the file to receive listing output.

/O
Identifies the option code string, for conditional compilation.

Example

```
)DGL TEST)
```

Compiles file TEST.DG or TEST., generating binary object file TEST.RB, if no errors occur. However, since no error or listing file is designated, error information will appear only on the console output device.

DIRECTORY *Command*

Format

DIRECTORY *[pathname]*

Purpose

Set or display the current DIRECTORY setting.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/I
Set directory to initial working directory (no arguments allowed).

/P
Set directory to previous environment's directory (no arguments allowed).

Argument Switches

None.

Examples

```
)DIRECTORY)
:UDD:USER
)DIRECTORY BETA)
)DIR)
:UDD:USER:BETA
)DIR/I)
)DIR)
:UDD:USER
```

First, display the working directory pathname; then make BETA the working directory and display its pathname. Finally, set the working directory to the initial directory and display it again.

Format

[!DIRECTORY]

This pseudo-macro does not accept arguments.

Purpose

Expand to current or previous environment's working directory.

Macroname Switches

/P
Use previous environment's working directory.

Argument Switches

None.

Examples

```
)WRITE THE WORKING DIRECTORY IS &)
&)[!DIRECTORY])
THE WORKING DIRECTORY IS :UDD:DAN
)DIR EXPENSE)
)WRITE THE PREVIOUS WORKING &)
&)DIRECTORY WAS [!DIRECTORY/P])
THE PREVIOUS WORKING DIRECTORY WAS
   :UDD:DAN
```

DISMOUNT *Command*

Format

DISMOUNT linkname *[operator-message]*

Purpose

Request the operator to dismount a tape.

You must specify the same linkname you used to mount the tape.

NOTE: If you issue a DISMOUNT command for a magnetic tape, the tape will be rewound and reassigned to the operator (i.e., its ACL will be OP,OWARE).

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)MOUNT MYTAPE PLEASE MOUNT TAPE&)
&)XB43)
        .
        .
        .
)DISMOUNT MYTAPE PLEASE SEND &)
&)TAPE TO LIBRARY)
)
```

Format

XEQ DISPLAY pathname

Purpose

Print a file in octal and ASCII.

Each line of the listing contains the octal and ASCII representation of 10_8 16-bit words from the file. Certain ASCII characters, such as form feed, tab, NEW LINE, and carriage return are unprintable. In these cases, DISPLAY writes the word as <nnn>, where nnn is the ASCII code for the character. If a byte contains a quantity which does not have an ASCII equivalent, DISPLAY prints a blank.

Display Switches

/L,/L=pathname
 See CLI Commands tab.

Argument Switches

None.

Example

```
)XEQ DISPLAY/L= @LPT DATA76)
)
```

Produce an octal and ASCII listing of the file DATA76 on the line printer.

DUMP *Command*

Format

DUMP dumpfile *[source-pathname]...*

If you don't supply *source–pathnames,* the template # is assumed.

Purpose

Dump one or more files from the working directory to the specified dumpfile.

Command Switches

/1, /2, /L, /L=pathname, /Q
See CLI Commands tab.

/V
Verify dumped files on @OUTPUT.

/FLAT
Do not maintain tree structure; dump all files from the specified directories into one directory.

/NACL
Dump files without ACLs.

/BUFFERSIZE=bytes
Blocks dumped to the tape will have a block size of bytes.

/BEFORE/TLM=time
Where time is in the form hh:mm:ss. Dump only those files modified before time.

/BEFORE/TLM=date
Where date is in the form dd-mmm-yy. Dump only those files modified before the specified date.

/DENSITY= {800 / 1600}

Command Switches (continued)

/BEFORE/TLM=date:time
Where date:time is in the form dd-mmm-yy:hh:mm:ss. Dump only those files modified before the specified date and time.

/BEFORE/TLA=
Dump only those files last accessed before the specified time, date, or date-time. See /BEFORE/TLM for format.

/AFTER/TLM=
Dump only those files modified after the specified time, date, or date-time. See /BEFORE/TLM for format.

/AFTER/TLA=
Dump only those files last accessed after the specified time, date, or date-time. See /BEFORE/TLM for format.

/TYPE=type
Select all files of the specified type where type can be in the form:

XXX	3-letter mnemonic.
n	decimal number (0-255).
m-n	decimal numbers (0-255) which define a range of file types.
\n	exclude decimal number (0-255).
\m-n	exclude decimal numbers (0-255) which define a range of file types.

You can use more than one /TYPE= switch in a command line.

DUMP (continued)

Argument Switches

None.

Example

```
)DUMP/V/FLAT/NACL FILE6.DUMP&)
&):UTIL:+.PR)
```

Dumps all program files into the UTILities directory without their ACLs to disk file FILE6.DUMP. The CLI verifies the dumped files.

Format

[!ELSE]

Purpose

Include CLI input conditionally.

You can use this pseudo-macro only when you have previously used an !EQUAL or !NEQUAL pseudo-macro. If the preceding !EQUAL or !NEQUAL is true, the CLI executes the input appearing before the !ELSE pseudo-macro and does not execute the input appearing after !ELSE.

If the preceding !EQUAL or !NEQUAL is false, the CLI executes the input after !ELSE up to the next !END pseudo-macro.

Macroname Switches

None.

Argument Switches

None.

Example

```
) [!EQUAL,1,2] WRITE EQUAL [!ELSE]&)
&)WRITE NOT EQUAL [!END])
NOT EQUAL
)
```

Since the !EQUAL pseudo-macro is false, the CLI executes the command(s) that follow the !ELSE pseudo-macro.

!END *Pseudo-Macro*

Format

[!END]

Purpose

End an !EQUAL or !NEQUAL pseudo-macro sequence.

Macroname Switches

None.

Argument Switches

None.

Examples

```
)[!EQUAL,1,1] WRITE EQUAL [!END])
```
EQUAL
```
)[!EQUAL,1,2] WRITE EQUAL [!END])

)
```

The first command line is executed because the condition is true; the second is not executed because the condition is false.

Format

ENQUEUE spooled-device pathname *[pathname]*...

You may use templates in the *pathname* argument.

Purpose

Queue one or more file entries to a spoolable output device (if your system has no EXEC process).

The operator must have enabled spooling to the device you want to use. The CLI places an entry for each file in the spooler output queue for that device.

Command Switches

1/,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

/B
Output in binary mode; don't interpret special control characters (such as TAB). Default is output in text mode.

/D
Delete disk file after output; default retains original file.

/H
Output header; default suppresses header at the top of each page.

/MES=x
Output message x to operator console; default is no message.

/P
Pause for operator response before outputting the file; default is immediate output.

Examples

```
)ENQUEUE @LPA OUTPUT.LS)
)
```

Queues OUTPUT.LS to the line printer.

```
)ENQUEUE @LPA OUTPUT.LS/P)
)
```

Waits for an operator response before outputting the file.

Format

[!EQUAL, argument$_1$, argument$_2$]

Purpose

Include input conditionally.

This pseudo-macro precedes a sequence of text conditionally executed by the CLI. Follow the !EQUAL pseudo-macro with the !END pseudo-macro. The sequence may also include the !ELSE pseudo-macro.

Two arguments must follow the !EQUAL pseudo-macro. The CLI compares these two arguments character by character. If the arguments match, the CLI executes the text up to the !ELSE or !END pseudo-macro. If the arguments do not match, the CLI does not execute this text.

Use commas to separate the arguments in the !EQUAL pseudo-macro.

Macroname Switches

None.

Argument Switches

None.

Example

```
)[!EQUAL,1,1]WRITE EQUAL[!END])
EQUAL
)
```

Format

EXECUTE program-name

Purpose

Execute a program.

The CLI creates a subordinate swappable process with the same priority and privileges as itself. In most cases, the CLI is blocked until the subordinate process terminates. EXECUTE is identical to XEQ or X.

Command Switches

/1,/2,/L,/L=pathname,/Q
 See CLI Commands tab.

/I
 Create input for program from @INPUT.

/M
 Create input for program from macro body.

/S
 Store program termination IPC message in STRING (instead of displaying it).

Argument Switches

As needed by the new program.

Examples

```
)EXECUTE MYSORT)
```

Run a program called MYSORT.

```
)EXECUTE/S PROG1)
```

Run a program called PROG1 and place its termination message in STRING.

Format

[!EXPLODE argument...]

Purpose

Expands argument into single-character arguments.

!EXPLODE expands the argument string by inserting commas between every pair of characters (including commas) in the original string. Use !EXPLODE to access characters of arguments as individual arguments.

If you use !EXPLODE with the WRITE command, as shown in the example below, the CLI displays commas between expanded argument characters as spaces.

Macroname Switches

None.

Argument Switches

None.

Examples

```
)WRITE [!EXPLODE [!DATE]])
0 □ 8 □-□ J □ U □ L □-□ 7 □ 7
```

[!DATE] expands to 0,8,-,J,U,L,-,7,7 and is displayed as shown above. Commas change to spaces when the CLI executes WRITE.

```
)TYPE [!EXPLODE AB])
This is file A.
This is file B.
```

AB expands to A,B and the CLI types the contents of file A and file B.

Format

XEQ *[/L [=listfilename]]* FCU.PR

Purpose

Set horizontal tabs and vertical form settings for a user file or a forms setting in directory :UTIL:FORMS.

You can run the Forms Control Utility (FCU) whenever you wish to create, edit, or list forms control specifications.

FCU Commands

The following is a list of the FCU commands and their meanings.

B
: Terminate the Forms Control Utility.

C
: Create forms control specifications for a file.

E
: Edit forms control specifications for a file.

H
: Display all FCU commands.

L
: Print a file's forms control specifications to the current list file.

NOTE: If you use this command, you must use the /L switch (see FCU switches).

T
: Type forms control specifications for a file.

FCU Switches

/L=listfilename
: List forms control specifications to listfilename. If you use the L command without appending this switch to the FCU command, you'll get an error message.

Argument Switches

None.

Examples

For a complete FCU dialog, see the *AOS Command Line Interpreter User's Manual.*

Format

XEQ FILCOM pathname$_1$ pathname$_2$

Purpose

Compare two files, word (16 bits) by word.

If the contents of the files differ, FILCOM displays the specific word number and the contents of each different word for both files. If one file is longer than the other, excess words in the longer file are displayed and dashes are displayed for the shorter file.

FILCOM Switches

/L
Write output to @LIST instead of @OUTPUT.

/L=name
Write output to name instead of @OUTPUT.

Argument Switches

None.

Example

Assume that the following files exist in our working directory:

```
FILE1          FILE2
ABCDEFG        ABCDEFG
HIJKLMN        ABCDEFG
OPQRSTU        OPQRSTU

)XEQ FILCOM FILE1 FILE2)

           FILE1     FILE2
000004     044111    040502
000005     045113    041504
000006     046115    042506
000007     047012    043412
)
```

Compare FILE1 and FILE2.

Format

[!FILENAMES *[pathname]* ...]

Purpose

Expand to a list of filenames.

If you don't include arguments, !FILENAMES expands to all filenames in the working directory.

Macroname Switches

None.

Argument Switches

None.

Example

```
)QBATCH XEQ MASM/L/E=@LPT&)
&)([!FILENAMES,+.SR]))
```

Create a separate batch job to assemble all files with a .SR suffix in the working directory.

Format

FILESTATUS *[pathname]...*

Purpose

List file status information or a group of filenames.

If you omit arguments, the CLI displays the status of all the files in the working directory. You can use filename templates in the pathname argument(s).

Command Switches

/1,/2,/L,/L=pathname,/Q
 See CLI Commands tab.

File Information Switches

/ASSORTMENT
 Displays an assortment of file information which includes file type, date and time of creation, file length (or if the file is a link, the file type, LNK, and the link resolution name).

/DCR
 File creation date.

/DLA
 Date file was last accessed.

/DLM
 Date file was last modified.

/ELEMENTSIZE
 The number of disk blocks in a file element for this file.

/HASHFRAMESIZE
 Directory's hash frame size.

File Information Switches (continued)

/INDEX
File's current number of index levels, and its maximum number of index levels.

/LENGTH
File's byte length.

/LINKNAME
A link's resolution name.

/PACKET
Display the entire contents of the packet returned by the ?FSTAT system call. See the *AOS Programmer's Manual* for more information.

/RECORD
Display a file's record format, either as an integer (fixed length), or as one of the following mnemonics:

DYN
Dynamic. A record length is specified for each read and write.

VAR
Variable. Each record starts with a header giving the size.

DS
Data sensitive. Records are terminated by specific characters embedded in the text.

/TCR
File creation date and time.

/TLA
Date and time the file was last accessed.

/TLM
Date and time file was last modified.

/TYPE
Displays the file's type, either as a three-character mnemonic or as a decimal number (0-255) if a mnemonic doesn't apply.

Specific Condition Switches

/BEFORE/TLM=time
Where time is in the form hh:mm:ss. List only those files modified before time.

/BEFORE/TLM=date
Where date is in the form dd-mmm-yy. List only those files modified before the specified date.

/BEFORE/TLM=date:time
Where date:time is in the form dd-mmm-yy:hh:mm:ss. List only those files modified before the specified date and time.

/AFTER/TLM=
List only those files modified after the specified time, date, or date-time. See /BEFORE/TLM for format.

/BEFORE/TLA=
List only those files last accessed before the specified time, date, or date-time. See /BEFORE/TLM for format.

/AFTER/TLA=
List only those files last accessed after the specified date, time, or date-time. See /BEFORE/TLM for format.

/TYPE=type
Select all files of the specified type where type can be in the form:

XXX	3-letter mnemonic.
n	decimal number (0-255).
m-n	decimal numbers (0-255) which define a range of file types.
\n	exclude decimal number (0-255).
\m-n	exclude decimal numbers (0-255) which define a range of file types.

NOTE: You can use more than one /TYPE= switch in a command line.

Formatting Switches

/CPL=n
Set the number of characters per line for output of FILESTATUS information (default=72).

/NHEADER
Do not print directory headers; list files with their pathnames, not as simple filenames.

/SORT
Sort the filenames alphabetically.

Argument Switches

None.

Example

```
)FILESTATUS/ASSORTMENT/SORT)

DIRECTORY :UDD:USER

DIR1        DIR    08-DEC-77    14:39:12    123456
LINK2       LNK    :UDD:USER
MIKE_Z      TXT    01-OCT-77    09:13:22    1111
PROG.PR     PRG    23-NOV-77    16:11:55    44876
TEXT.DOC    TXT    07-DEC-76    03:03:30    30000

)
```

This command displays a sorted list of filenames. Information includes the type of file, creation date and time, and character length of the file.

Format

FORT4 *[switches]* inputfilename

Purpose

Compile a FORTRAN IV source file.

FORT4 is a utility which you use to compile a FORTRAN IV source file. The FORT4 command first searches for inputfilename.FR, and if it is not found, FORT4 searches for inputfilename.

For more information about the FORT4 macro and the FORTRAN IV programming language, consult the following manuals:

FORTRAN IV Reference Manual (093-000134)
FORTRAN IV Runtime Library User's Manual (093-000142)

FORT4 Switches

/A
ABORT on system error -- return to CLI.

/NA
Compile only; don't assemble.

/B
Brief listing. Compile source program input only.

/F
Equivalence FORTRAN variable names and statement numbers. FORTRAN variables are equivalenced to assembler.

/N
Do not produce object file.

FORT4 Switches (continued)

/P
Process only the first 72 characters per record.

/U
Ouput user symbols in assembly phase.

/X
Compile statements with an X in column 1.

/E=filename
Output errors to filename. If you use /E without filename, error messages are suppressed. If you omit /E, error messages are sent to the current output filename.

/L=filename
Output listing to filename. If you use /L without filename, list filename is the current LIST filename. If you omit /L, no list file is provided unless you used /B to produce a brief list.

/O=filename
Give the module this filename. If you don't use /O, the object filename is inputfilename.OB.

/S=filename
Save the intermediate source file (compiler output) and name it filename. If you use /O without filename, the source filename is saved and named inputfilename.SR. If you omit the /S switch, the source file is deleted after assembly.

Argument Switches

None.

Example

```
)FORT4/B MYPROG)
```

This example compiles MYPROG.FR, giving a brief listing to the current LIST file. Because /E was omitted, all errors are sent to the current OUTPUT filename. The compiler produces MYPROG.OB as the object file.

Format

F5 *[switches]* inputfilename

Purpose

Compile a FORTRAN 5 source file.

For a complete description of the FORTRAN 5 programming language and the CLI F5 command line, see the following manuals:

FORTRAN 5 Reference Manual (093-000085)
FORTRAN 5 Programmer's Guide (AOS) (093-000227)

F5 Switches

/B
Produce a brief listing. This includes the input source program, the storage map, the list of all subprograms called, the cross-reference, and the error list. The listing does not include the generated code.

/C
Check the syntax of the source program. If you specified a listing file, the source program and the error list are sent to it. The error list is also sent to it. The error list is also sent to the error file, if one exists.

/D
Debug. Compile code that allows the long form error traceback routine to output line numbers. This option doesn't provide more information when an error occurs but does provide for a more convenient form. Don't use /D in final versions of programs.

F5 Switches (continued)

/E
/E=pathname
Output errors to pathname. If you omit =pathname, error messages are suppressed. If you don't use /E, error messages are output to the current CLI output pathname.

/I
Don't list source lines from INCLUDE files. /I permits you to include large parameter files in programs without producing bulky listings. Line numbers on /I listings correspond to those on standard listings.

/L
/L=pathname
Output listing to pathname. If you omit =pathname, the list pathname is the current CLI LIST pathname. If you don't use /L, but use /B, you will get a brief listing as output to the current CLI LIST pathname.

Do not send the listing output directly to the line printer, @LPT; the line printer will print your output in five separate pieces and might include other users' output.

/N
Do not produce an object file.

/O=pathname
Give the object file this name. If you don't use /O, inputpathname.OB is the object pathname.

/P
Use punched card input. Only the first 72 characters of each input line are used as FORTRAN 5 source code, but the entire input line is sent to the LISTFILE, if one exists.

F5 Switches (continued)

/S

Generate code to check subscript references. A runtime routine determines whether or not a reference lies within an array. For singly-subscripted arrays, the check always catches bad references. For arrays with more than one subscript, the check may not catch an out-of-range subscript. For example:

```
DIMENSION A(2,4)
B=A(3,2)
```

produces no error since the address calculated for A(3,2) is the same as that for A(1,3), which is within the array.

/X

Compile lines with an X in column one. If you don't use /X, the system treats these lines as comments.

/NOLEF

Don't generate Load Effective Address instructions (LEFs). This switch is useful if you're using I/O instructions in assembly language routines combined with FORTRAN 5 programs. See the *AOS Programmer's Manual* for further information on the LEF mode of program execution.

Argument Switches

None.

Examples

```
)F5 MYPROG)
```

Compiles either MYPROG.FR or MYPROG, depending on the existence of the .FR file. This command sends a brief listing to the current CLI LIST file. Since there's no /E switch all errors are output to the current OUTPUT pathname. The compiler produces the object file, MYPROG.OB.

```
)F5/E/I/L=PROG.LS PROG)
```

Compiles either PROG.FR or PROG, depending on the existence of the .FR file. This command generates a listing file, PROG.LS. If PROG.LS already exists, the new listing is appended to it. No error file is created and the source listing will not include lines from INCLUDE files.

Format

HELP *[item]...*

Items can be any one of the following:

- CLI Command
- CLI Pseudo-macro
- General CLI topic

Purpose

Explain a CLI command, pseudo-macro, or general topic.

Some CLI HELP files are longer than your screen. To freeze the display, type CTRL-S; to resume the display, type CTRL-Q.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/V
Display verbose description of the specified item. If you don't include /V, the CLI displays a brief description of the item.

Argument Switches

None.

Example

```
)HELP/V MOUNT)
        .
        .    (HELP message)
        .
        .
)
```

Display verbose description of MOUNT command.

Format

```
INITIALIZE physical-unitname...
```

Purpose

Graft a logical disk (LD) into the working directory.

You must name all the units on which the volumes are mounted; if the logical disk contains more than one physical disk, you must name each one. After executing the INITIALIZE command, the system displays the name of the new logical disk.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/S
Do not display the name of the new logical disk; instead, store the name in STRING where you can use it as an argument to commands via the !STRING pseudo-macro.

Argument Switches

None.

Examples

```
)INITIALIZE @DPD0)
CAIN
)
```

CAIN is the highest directory on the grafted logical disk (LD). You must have APPEND access to the directory onto which this LD is grafted.

```
)INITIALIZE @DPD10 @DPD14)
ADAM
)
```

The logical disk ADAM consists of two physical disk units -- DPD10 and DPD14.

LABEL

Format

XEQ LABEL device-name volume-id

Where device-name is the name of the tape (e.g., @MTA0); device-name should not be a labeled tape device (e.g., @LMT). Here volume-id is a six-character identifier.

Purpose

Prepare a magnetic tape with a volume label.

LABEL writes a basic set of labels for a null file. The label set may contain a volume label, a header label, and a trailer label.

Label Switches

/I
Create IBM labels. If you omit this switch, ANSI standard labels are created.

/S
Scratch the tape. This switch causes beginning and end-of-file labels to be rewritten for a null first file, effectively deleting all files on the tape.

Argument Switches

None.

Example

```
)XEQ LABEL @MTA0 100000)
)
```

Once the tape has been labeled, you can refer to it by appending the volume-id and filename; e.g., @LMT:100000:filename.

Format

LEVEL

Purpose

Display the current CLI environment level number.

Use LEVEL in conjunction with the PUSH and POP commands.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Examples

```
)LEVEL)
LEVEL 0
)PUSH)
)LEVEL)
LEVEL 1
)
```

Display current LEVEL. Push a level and display the new current LEVEL.

Format

XEQ LFE function-letter argument *[argument]...*

Purpose

This utility edits and analyzes library files, which are sets of object files having special starting and ending blocks. The extension .LB usually designates library files. See the *AOS Library File Editor User's Manual* (093-000198) for further information.

To execute an LFE command, type the CLI command XEQ, the program name LFE, a single function-letter and one or more arguments. Function letters specify the type of LFE operation you wish to perform, while arguments indicate which files or libraries you wish to include in the LFE operation.

Function Letters

A
Analyze one or more libraries.

D
Delete one or more object files from a library.

I
Insert one or more object files into a library.

M
Merge two or more libraries to form a new library.

N
Create a new library.

R
Replace library object files with new object files.

Function Letters (continued)

T
List the titles of object files in a library.

X
Extract object files from a library and place them into a new object file.

LFE Switches

/L=pathname
Write output to the filename specified in pathname. If you omit the switch, the listing goes to the current @LIST file. This switch is meaningful only in command lines containing function letters A or T.

/F
Form feed: Print the analysis of each object on a separate page. This switch is meaningful only in command lines containing function letter A. If used, it must follow the /L command switch.

Function Switches

/M
Multiple-Input: You can apply this switch only to function letter A. It analyzes all library filenames provided in the argument as one library.

/A
Insert after: You can apply this switch only to function letter I. It inserts object files after the file which the switch modifies.

/B
Insert before: You can apply this only to function letter I. It inserts object files before the file which the switch modifies.

NOTE: When neither an /A or /B switch is given, the utility makes inserts at the beginning of the new library file.

Argument Switches

/C
Ignore .FORC pseudo-op in this module.

/F
Bind this module.

/I
Input library file; This is an original library file.

/O
Output library file: This is a new library file.

NOTE: Arguments modified by /I or /O can appear anywhere in the command line following the function-letter and optional function switch. You must use the /O switch to specify an output file. If you omit the /I switch, LFE searches for original-library.LB or a file with a .OB extension. If it doesn't find such a file, LFE searches for library or object.

Examples

```
)XEQ LFE I TRIG.LB/I TRIG1.LB/O SINH.OB&)
&) COS/B SIN.OB)
)
```

Create a new file TRIG1.LB, which is a copy of TRIG.LB with SINH.OB inserted at the beginning of the file (immediately after the library start block) and SIN.OB inserted immediately before COS.

```
)XEQ LFE T TRIG1.LB)
SINH.OB
SIN.OB
```

List the titles of the object files contained in library TRIG1.LB.

Format

XEQ LINEDIT *[pathname]*

Purpose

Create or edit ASCII text.

This command calls the LINEDIT text editor program. If the file you specify in *pathname* does not exist, LINEDIT asks:

DO YOU WANT THE FILE TO BE CREATED?

Answer Y to create the file; LINEDIT then displays its prompt (?).

If you include *pathname* and the file exists, LINEDIT opens it for editing and displays the prompt.

A summary of LINEDIT commands appears later in this manual. See Editors.

For more information on LINEDIT, see the *AOS LINEDIT Text Editor User's Manual.*

LINEDIT Switches

/L=pathname
 Set @LIST file for LINEDIT.

Argument Switches

None.

Example

```
)XEQ LINEDIT CJM)
DO YOU WANT THE FILE TO BE CREATED? Y)
?APPEND)
        .
        .
        .
?BYE)
CJM BEING UPDATED
)
```

Call LINEDIT and create a file called CJM. Enter the desired LINEDIT commands and terminate LINEDIT with the BYE command.

Format

LISTFILE *[listfilename]*

Purpose

Set or display the current LISTFILE.

The generic LISTFILE is @LIST. If *listfilename* doesn't exist, it is created. If *listfilename* exists, subsequent data is appended to it.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/K
Set LISTFILE to null (no arguments allowed).

/G
Set LISTFILE to the generic @LIST file (no arguments allowed).

/P
Set current LISTFILE to the previous environment's LISTFILE (no arguments allowed).

Argument Switches

None.

Examples

```
)LISTFILE)
@ LIST
)LISTFILE @LPT)
)
```

First, display LISTFILE, which is the generic @LIST. Then set LISTFILE to the line printer.

Format

LOAD dumpfilename *[source-pathname]...*

If you don't supply source pathname(s), the template # is assumed.

Purpose

Load one or more previously dumped files into the working directory.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/DELETE
Delete any existing file with the same filename and load a new file.

/FLAT
Do not maintain tree structure; load all files into the working directory.

/N
Do not load files. Print dumped date and filenames only.

/RECENT
Do not load if the file on disk is newer than the file on dump file.

/V
Verify each loaded file on @OUTPUT.

/BEFORE/TLM=time
Where time is in the form hh:mm:ss. Load only those files last modified before time.

Command Switches (continued)

/BEFORE/TLM=date
Where date is in the form dd-mmm-yy. Load only those files last modified before the specified date.

/BEFORE/TLM=date:time
Where date:time is in the form dd-mmm-yy:hh:mm:ss. Load only those files last modified before the specified date and time.

/AFTER/TLM=
Load only those files last modified after specified time, date, or date-time. See /BEFORE/TLM for format.

/BEFORE/TLA=
Load only those files last accessed before the specified time, date, or date-time. See /BEFORE/TLM for format.

/AFTER/TLA=
Load only those files last accessed after the specified time, date, or date-time. See /BEFORE/TLM for format.

/BUFFERSIZE=bytes
The maximum block size of the tape is bytes.

/TYPE=type
Select all files of the specified type where type can be in the form:

XXX	3-letter mnemonic.
n	decimal number (0-255).
m-n	decimal numbers (0-255) which define a range of file types.
\n	exclude decimal number (0-255).
\m-n	exclude decimal numbers (0-255) which define a range of file types.

You can use more than one /TYPE= switch in a command line.

Argument Switches

None.

Examples

```
)LOAD/V @MTA0:0)
MYFILE.SR
MYFILE.PR
MIKE.CLI
)
```

Load all files contained in file 0 on the magnetic tape mounted on unit 0 into the working directory and list their names on the console.

```
)LOAD/AFTER/TLA=03-DEC-78:13:30:00@MTA0:0)
)
```

Load only those files in file 0 magnetic tape unit 0 which were accessed after 1:30 p.m., December 3, 1978.

Format

[!LOGON]

Purpose

Determine if a user is logged on under the EXEC, and if so, expand to CONSOLE or BATCH.

If you're not sure whether you're logged on under the EXEC, use this pseudo-macro. It will return CONSOLE or BATCH if you are logged on under EXEC, or nothing at all if you are not logged on under EXEC.

Macroname Switches

None.

Argument Switches

None.

Example

```
)WRITE [!LOGON])
BATCH
)
```

Determine whether or not user is logged on under EXEC and return appropriate message.

Format

XEQ MASM pathname...

Purpose

Assemble one or more source files to produce an object file.

Output can be an object file, a listing file, or both.

The MASM command searches first for pathname.SR; if not found, it searches for pathname.

For more information about MASM, see the *AOS Macroassembler Reference Manual* (093-000192)

MASM Switches

/B=name
Name object file name.OB instead of the name of the first source file.

/E
Prevent Pass 2 error messages from appearing on @OUTPUT (unless there is no listing file). Some Pass 1 errors will appear on @OUTPUT automatically.

/F
Generate or suppress a form feed as necessary to produce an even number of listing pages. By default, a form feed is generated after each page.

/L
Produce a listing file, which includes a cross reference of the symbol table. File MASMXR.PR must be available on disk. Listing goes to @LIST.

MASM Switches (continued)

/L=name
Produce a listing file to name instead of to the current @LIST file.

/K
Keep the assembler's symbol file table (MASM.ST) at the end of the assembly. The symbol table is deleted by default.

/M
Flag redefinition of permanent symbols as multiple-definition errors.

/N
Do not produce an object file.

/O
Override all listing suppression controls; e.g., .NOLOC 1.

/P
Add semipermanent symbols to cross reference.

/PS=pathname (or filename)
Define a different .PS file for use by MASM instead of MASM.PS.

/R
Produce an .OB file even if there are assembly errors in the source file(s). By default when there are errors, MASM produces no .OB.

/S
Skip Pass 2 and save version of the symbol table and macro definitions in MASM.PS (delete the old MASM.PS, if one exists).

/U
Include user symbols in the object output.

Argument Switches

/S

Skip this file on Pass 2 of the assembly. You should use this switch only if the file does not assemble any storage words. Macro definition files can be skipped on Pass 2.

Example

```
)XEQ MASM/N TESTA)
```

Assemble file TESTA. This is our first attempt to assemble TESTA, hence we produce no object file or listing in case there are assembly errors. Errors are sent to @OUTPUT.

Format

MESSAGE errorcode...

Purpose

Display text message corresponding to error code arguments.

Command Switches

/1,/2,/L,/L=pathname,/Q
 See CLI Commands tab.

/D
 Arguments are decimal integers. If you omit /D, arguments are octal integers.

Argument Switches

None.

Examples

```
)MESSAGE 25)
25 FILE DOES NOT EXIST
)
```

Display the error message for error code 25.

```
)MESSAGE 45)
45 ILLEGAL DEVICE CODE
```

Display the error message for error code 45.

Format

XEQ MKABS filename 1 filename 2

Purpose

Convert an RDOS save file to an absolute binary file.

This utility runs on your AOS system and produces an absolute binary file which will run on a stand-alone system. Filename 1 is an RDOSBIND save file you want to input to MKABS. Filename 2 is a file you designate to receive output. MKABS outputs filename 1 to filename 2 in absolute binary form. You can then use the binary loader to load the absolute binary file onto a stand-alone system, and execute the program.

MKABS Switches

/START=N

Set the second word in the absolute binary file start block to starting address N, where N is an octal number in the range -1<N<77777. After the file is loaded, the program begins execution at the starting address. If N is negative, the loader will halt after loading. If N is omitted, the system uses the address specified in the RDOS user table (USTSA) as the start address. IF RDOSBIND detected no starting address when the file was created, USTSA will contain a -1, and the loader will halt after loading.

/ZERO

Assume that the save file begins at core image location zero and that it was developed for RTOS. If you omit this switch, MKABS assumes that the save file begins at location 16 octal and that it was developed for RDOS.

MKABS Switches (continued)

/FROM=N
Set the first save file address which MKABS will include in absolute file output equal to N. MKABS evaluates N as an octal number, relative to location zero. If you omit this switch, MKABS uses the first save file address (0 for RTOS files, 16 octal for RDOS files) as the *from* address.

/LAST=N
Set the last save file address which MKABS will include in absolute file output equal to N. MKABS evaluates N as an octal number, relative to location zero. If you omit this switch, MKABS uses the final save file address as the *last* address of that file.

Argument Switches

None.

Example

```
)XEQ MKABS/START=300  RDOS.SV  ABIN)
)
```

Copies the program contained in RDOS save file RDOS.SV into file ABIN; copy it in absolute binary form. After you use the basic binary loader to load ABIN onto a stand-alone system; it will begin executing at octal starting address 300.

Format

MOUNT linkname operator-message

Where linkname is the file which you originate for the device; operator-message is the action that you want the operator to perform.

Purpose

Mount a tape.

After the operator mounts the tape, the system creates linkname in your initial working directory. Linkname contains a pathname to the actual device on which the volume was mounted. (MOUNT creates a link for both labeled and unlabeled tapes.) Once you type this command, your console will lock until the operator responds. You need not use the CLI MOUNT command to mount a volume of tape on a tape drive; simply refer to the tape's filename.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/VOLID=volid logicalname
This MOUNT command refers to a labeled tape or a set of labeled tapes. You must specify the VOLID of every volume which comprises the file set you are using. The VOLID=volid logicalname switches must occur in the order that the volumes are to be used. The EXEC will automatically have the operator change volumes as needed, and the system will verify that the operator has mounted the correct volume each time.

Argument Switches

None.

Examples

```
)MOUNT TAPE1&)
&)PLEASE_MOUNT_TAPE_NO_Z280)
)DUMP TAPE1:0 +.PR)
```

First, tell the operator to mount a tape called Z280 and direct the system to create a link for it named TAPE1. Then, DUMP to TAPE1 all .PR files in the working directory.

```
)MOUNT/VOLID=FIRST MYFILE REMOVE&)
&)ENABLE RING)
)
```

The EXEC creates the link:

```
:UDD:USERNAME:MYFILE LNK @LMT:FIRST
```

and you can refer to a file on the tape by specifying:

```
MYFILE:FILENAME)
```

MOVE *Command*

Format

MOVE destination-directory source-file...

You may use filename templates for the source-files, but not for destination-directory. If you do not supply any source filenames, the template # is assumed. Source-file must be in either the working directory or another directory inferior to the working directory.

Purpose

Move copies of one or more files to destination-directory.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/DELETE
Delete the file in the destination-directory if it has the same filename as a source-file.

/FLAT
Do not maintain tree structure. Move all source files into destination-directory.

/NACL
Give the new files the default ACL, which consists of your username and all access privileges (OWARE).

/RECENT
If there is a file in the destination-directory with the same filename as one of the source-files, move source-file only if it is more recent than the existing file.

/V
List the name of each file moved on @OUTPUT.

Command Switches (continued)

/BEFORE/TLM=time
Where time is in the form hh:mm:ss. Move only those files last modified before time.

/BEFORE/TLM=date
Where date is in the form dd-mmm-yy. Move only those files last modified before the specified date.

/BEFORE/TLM=date:time
Where date:time is in the form dd-mmm-yy:hh:mm:ss. Move only those files last modified before the specified date and time.

/AFTER/TLM=
Move only those files last modified after the specified time, date, or date-time. See /BEFORE/TLM for format.

/BEFORE/TLA=
Move only those files last accessed before the specified time, date, or date-time. See /BEFORE/TLM for format.

/AFTER/TLA=
Move only those files last accessed after the specified time, date, or date-time. See /BEFORE/TLM for format.

/TYPE=type
Select all files of the specified type where type can be in the form:

XXX	3-letter mnemonic.
n	decimal number (0-255).
m-n	decimal numbers (0-255) which define a range of file types.
\n	exclude decimal number (0-255).
\m-n	exclude decimal numbers (0-255) which define a range of file types.

You can use more than one /TYPE= switch in a command line.

Argument Switches

None.

Examples

```
)MOVE□ ↑ DIR1)
)
```

Move the entire subtree inferior to the working directory to DIR1. Note that since you did not specify any source-files, the CLI assumes the template #.

```
)MOVE/AFTER/TLM=12-DEC-79 DIR4 +.SR)
```

Move to DIR4 each file ending with the extension .SR which was last modified on or after December 12, 1979 and which is contained in the working directory.

Format

[!NEQUAL, argument$_1$, argument$_2$]

Purpose

Include input conditionally.

You must follow the !NEQUAL pseudo-macro with the !END pseudo-macro. The sequence may also include the !ELSE pseudo-macro.

You must always include two arguments to the !NEQUAL pseudo-macro. The two arguments are compared character by character. If the arguments *do not match,* the input up to the !ELSE or !END pseudo-macro is executed. If the arguments *do match,* the input up to the !ELSE or !END pseudo-macro is not executed.

Macroname Switches

None.

Argument Switches

None.

Example

In a macro:

```
    .
    .
    .
[!NEQUAL,%1%,*]
WRITE NOT AN ASTERISK
[!ELSE]
WRITE AN ASTERISK
[!END]
```

This macro will write NOT AN ASTERISK if you call it with any argument other than *; otherwise, it will write AN ASTERISK.

Notice that we used commas to separate the arguments in the !NEQUAL pseudo-macro. If we used spaces and argument 1 was null, the spaces on either side of the %1% would have become a single delimiter, giving [!NEQUAL,*]. This format is invalid because the !NEQUAL pseudo-macro takes exactly two arguments.

Format

[!OCTAL decimal-number]

The decimal-number must be a positive decimal integer in the range from 0 to 4,294,967,295. The result will be in the range from 0 to 37,777,777,777.

Purpose

Convert a decimal number to octal.

Macroname Switches

None.

Argument Switches

None.

Example

```
)WRITE [!OCTAL 1000])
1750
)
```

Expand to the octal equivalent of the decimal number 1000.

!OPERATOR *Pseudo-Macro*

Format

[!OPERATOR]

Purpose

Expand to ON or OFF depending on whether operator is on or off duty.

Use !OPERATOR in a macro to change macro behavior based on the presence of an operator.

Macroname Switches

None.

Argument Switches

None.

Examples

In a macro named TRY.CLI:

```
[!EQUAL,[!OPERATOR],ON]
      .
      .
(Sequence of commands if operator is ON)
      .
      .
[!ELSE]
WRITE TRY AGAIN LATER
[!END]
```

Execute the macro (operator is not ON).

```
)[TRY])
TRY AGAIN LATER
```

Format

PATHNAME pathname

Purpose

Display a complete pathname starting at the root directory.

You must have EXECUTE access to the file whose pathname you specified in the command line.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Examples

```
)PATHNAME =)
:UDD:USER:BETA
)PATHNAME TEST)
:UDD:USER:BETA:TEST
)
```

First, display the pathname of =, the working directory; then display the pathname of TEST, a son file.

!PATHNAME *Pseudo-Macro*

Format

```
[!PATHNAME pathname]
```

Purpose

Expand to a file's full pathname.

A full pathname starts at the root directory and ends with the specified filename.

Macroname Switches

None.

Argument Switches

None.

Example

```
)CREATE/LINK NIM [!PATHNAME NIM.PR])
```

Create a link entry named NIM containing a full pathname to the program NIM.PR.

Format

```
PAUSE {seconds       }
      {seconds.tenths}
```

Where seconds is a number between 0 and 4294968, and tenths is a digit between 0 and 9.

Purpose

Pause the CLI for the specified number of seconds.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)PAUSE 8.5)
```

Delay the CLI for 8.5 seconds.

PED

Utility

Format

```
XEQ PED
```

Purpose

Invoke the Process Environment Display (PED) program, which displays runtime data on all system processes and consoles.

PED is a privileged utility.

Referral

For further information on PED, see the *AOS Operator's Guide.*

Format

PERFORMANCE

Purpose

Display information about the CLI.

The information displayed includes:

System calls	The number of system calls the CLI has made since the last PERFORMANCE command, and the number of system calls since the CLI started.
Shared	Number of 2K-byte pages of shared memory.
Unshared	Current number of pages of unshared memory; maximum pages of unshared memory; greatest number of pages of unshared memory since this CLI started.
Stack faults	Number of stack faults since this CLI started; that is, the number of times this CLI process has grown in 2K-byte memory pages.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)PEFORMANCE)
10/15 SYSTEM CALLS
SHARED: 17 PAGES
UNSHARED: CURRENT 1 PAGE
POSSIBLE: 15 PAGES, HIGHEST 1 PAGE
0 STACK FAULTS
)
```

Format

PERMANENCE pathname $\left[\begin{Bmatrix} ON \\ OFF \end{Bmatrix}\right]$

Purpose

Set or display a file's permanence attribute.

A permanent file cannot be deleted (unless you turn its permanence OFF, or delete the directory it is in). The PERMANENCE command permits you to protect key files from accidental deletion.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/V
Display the filename with its PERMANENCE attribute.

Argument Switches

None.

Examples

```
)PERMANENCE ZFILE)
OFF
)PERMANENCE ZFILE ON)
)PERMANENCE/V ZFILE)
ZFILE ON
)
```

First, display the permanence status of file ZFILE; second, set permanence for the file to ON; then display the current PERMANENCE status.

!PID

Pseudo-Macro

Format

[!PID]

The CLI's process ID (PID) is a three-digit number.

Purpose

Expand to your CLI's process ID.

Macroname Switches

None.

Argument Switches

None.

Example

```
)WRITE MY PID IS [!PID])
MY PID IS 017
)
```

Expand to user's PID.

Format

```
XEQ PL1 sourcefilename
```

Purpose

Compile a PL1 (PL/I) source file.

PL/I (PL1) is a high-level language based on ANSI standard PL/I. In order to run a PL/I program, you must first compile it and then bind it. The AOS PL/I Compiler Utility recognizes PL/I words and symbols in the source file and generates code which the AOS Binder Utility uses to produce an executable program. PL/I source files have the extension .PL1. The compiler produces an object file of the same name, but with the extension .OB instead.

To bind a PL/I object file, invoke the Binder with the following macro:

PL1BIND mainobject *[object...]*

Note that you do not precede this macro with XEQ. The PL1BIND macro will accept the switches of the BIND command.

Referral

For detailed information on the PL/I programming language, see the *Plain PL/I Manual* (069-000021) and the *PL/I Reference Manual* (093-000204).

Command Switches

/E=name
Send compiler error messages to name instead of @OUTPUT. If the name file already exists, the error messages will be appended to it.

/L
Produce a listing file, using the current LISTFILE. The listing consists of line-numbered source text, a variable map, any compilation errors, and compilation statistics.

/N
Do not produce an object file.

/NEST
Print the nesting level of blocks and groups on the source listing.

/O=name
Write an object file to name.

/OPT
Invoke the compiler optimization phase, level 3.

/OPT *[=1 =2 =3]*
Invoke the compiler optimization phase at levels 1, 2, or 3.

/STAT
Write compilation statistics to @OUTPUT.

/SUB
Compile into the program code which will check for out-of-bounds subscripts and arguments to SUBSTR.

Examples

```
)XEQ PL1/L=LFILE MYPROG)
```

This command compiles the PL/I source file, MYPROG.PL1. The /L=LFILE switch directs the compiler to output the listing to disk file -- LFILE.

```
)PL1BIND MYPROG)
```

This command binds the object file MYPROG.OB (generated from the MYPROG source file by the compiler). Since no list file is specified, the Binder will output any messages that occur during binding to your terminal.

POP *Command*

Format

POP

Purpose

Return to the previous environment level.

Upon return, restore previous environment settings of SUPERPROCESS, SUPERUSER, SCREENEDIT, SQUEEZE, CLASS1, CLASS2, LISTFILE, DATAFILE, DIRECTORY, SEARCHLIST, STRING, and PROMPT. None of the current settings are preserved. You cannot POP from level 0.

Command Switches

/1,/2,/L,/L=pathname,/Q
 See CLI Commands tab.

/V
 Verify new level.

Argument Switches

None.

Example

```
)LEVEL)
LEVEL 1
)POP/V)
LEVEL 0
```

First display LEVEL, then POP to the previous environment and verify the new LEVEL.

Format

```
XEQ PREDITOR
```

Purpose

Create and edit user profiles.

PREDITOR is a privileged utility.

Referral

For further information on PREDITOR, see the *AOS System Manager's Guide.*

PREVIOUS *Command*

Format

PREVIOUS

Purpose

Display the previous environment's settings.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches:

None.

Example

```
)LEVEL)
LEVEL 1
)PREVIOUS)
LEVEL            0
SUPERPROCESS     OFF
SUPERUSER        OFF
SCREENEDIT       ON
SQUEEZE          OFF
CLASS1           ABORT
CLASS2           WARNING
LISTFILE         @ LIST
DATAFILE         @ DATA
DIRECTORY        :UDD:JOHN
SEARCHLIST       :PER,:UTIL,:
STRING
PROMPT
)
```

First, display the current LEVEL; then display the previous environment's settings.

Format

$$\text{PRIORITY} \left\{\begin{array}{l}\text{username:procname}\\ \text{process id}\end{array}\right\} \textit{[new priority]}$$

Purpose

Set or display the priority of the CLI or a subordinate process.

If you have the SUPERPROCESS privilege, you can set or display the priority of any process.

If you don't specify a new priority, the CLI displays the priority of the existing process. If you do specify a new priority, the selected process will have its priority changed to the new value. (The *new priority* is a decimal number greater than or equal to the priority with which the process was created.)

If you input PRIORITY without arguments, the CLI's priority will be displayed.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)PROCESS SMITH:SON1 )
PID 17
)PRIORITY 17 1 )
)
```

Set the process SON1's priority to 1.

PROCESS *Command*

Format

PROCESS program *[arguments-to-new-process]*

Purpose

Create a son process with the specified program.

You select the process's type, priority, and privileges via command switches, all of which are optional. You cannot pass a privilege which you do not have yourself.

If you omit privilege switches, the new process will be created with no privileges.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/ACCESSDEVICES
Allow the new process to identify and access user devices via system calls ?IDEF, ?DEBL, and ?STMAP.

/BLOCK
Block the CLI until this process terminates. If you omit this switch, the CLI displays the new process ID.

/CALLS=number
Maximum number of concurrent system calls for new process. (Default is the same as that of the creating process.)

/CHPRIORITY
Allow the new process to change its priority to a higher priority than that with which it was created.

Command Switches (continued)

/CHTYPE
Allow the new process to create any other type of process. Also permit the new process to change its own process type.

/CHUSERNAME
Allow the new process to create a process with a different username than its own.

/CONSOLE
New process's console is the same as that of the creating process. Default is no console.

/CONSOLE=consolename
New process's console.

/DATA
New process's generic @DATA filename is the same as that of the creating process.

/DATA=pathname
Make pathname new process's generic @DATA filename.

/DEBUG
Start the new process in the Debugger.

/DEFAULT
Give the new process the same privileges as that of the creating process.

/DIRECTORY
Make the new process's initial directory the creating process's working directory. (Default is creating process's initial directory.)

/DIRECTORY=name
Make name the new process's initial directory.

Command Switches (continued)

/INPUT
New process's generic @INPUT filename is the same as that of the creating process. (Default is no @INPUT filename.)

/INPUT=pathname
Make pathname the new process's generic @INPUT filename.

/IOC
Use this switch in conjunction with the /BLOCK switch. The new process's generic @INPUT, @OUTPUT, and @CONSOLE filenames are the same as that of the creating process.

/IOC=consolename
Make consolename the new process's generic @INPUT, @OUTPUT, and @CONSOLE name.

/IPCUSAGE
Allow the new process to issue the primitive IPC call.

/LIST
Write the new process's output to the current LISTFILE instead of @OUTPUT.

/LIST=pathname
Write the new process' output to the file specified by pathname instead of @OUTPUT.

/MEMORY=pages
Where pages is the maximum memory size of new process in 2K byte pages. (Default is the same as that of the creating process.)

/NAME=name
Where name is the simple process name for the new process. (If omitted, system assigns the name.)

/NOBLOCKPROC
Allow the new process to create another process without blocking.

/OUTPUT
New process's generic @OUTPUT filename is the same as that of the creating process. (Default is none.)

/OUTPUT=pathname
Make pathname the new process's generic @OUTPUT filename.

/PMGRPRIVILEGES
Allow the new process all rights of the peripheral manager.

/PRIORITY=number
Make number the new process's priority. (Default is the same as that of the creating process.)

/PREEMPTIBLE
New process is preemptible.

/RESIDENT
New process is resident.

NOTE: If you omit /PREEMPTIBLE and /RESIDENT switches, the new process is swappable.

/SONS=number
The maximum number of son processes that the new process can create is number. (Default is the same as that of creating process minus one.)

/SONS
Son process may create the same number of processes as the creating process minus one.

Command Switches (continued)

/SUPERPROCESS
Allows you to pass the SUPERPROCESS privilege to a son process.

/SUPERUSER
Allows you to pass the SUPERUSER privilege to a son process.

/UNLIMITEDSONS
Allows the new process the option of creating an unlimited number of son processes, and the option of remaining unblocked while its son(s) executes.

/USERNAME=name
Make name the new process's username. (Default is the same as that of the creating process.)

Argument Switches

As needed by the new program.

Example

```
)PROCESS/IOC=@CON1 UPDATE)
PID 13
)
```

Create a swappable son process with @INPUT, @OUTPUT, and @CONSOLE equivalent to @CON1 and UPDATE as its program. The CLI displays the process ID of the subordinate (13).

Format

PROMPT *[command]...*

Purpose

Set or display the current PROMPT setting.

You can specify up to any eight CLI commands which will be executed before the prompt is issued. When setting a PROMPT argument, you must enter only the CLI command name (no associated arguments or switches).

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/K
Set PROMPT to null.

/P
Set PROMPT to previous environment's setting (no arguments allowed).

Argument Switches

None.

PROMPT (continued)

Examples

```
)PROMPT TIME DATE DIRECTORY)
9:36:15
23-FEB-78
:UDD:USER:JOHN
)PROMPT)
TIME DATE DIRECTORY
9:36:19
23-FEB-78
:UDD:USER:JOHN
)PROMPT/K)
)
```

First, set PROMPT to the CLI commands -- TIME, DATE, and DIRECTORY. These commands will be executed before the prompt character is issued. The second PROMPT displays the prompt, and the last command sets PROMPT to null.

Format

```
PRTYPE { username:procname }
       { process id        }

       [ { PREEMPTIBLE } ]
       [ { RESIDENT    } ]
       [ { SWAPPABLE   } ]
```

Purpose

Set or display the type of an inferior process or any process, if you have the SUPERPROCESS privilege.

The EXEC creates only swappable processes. Hence, if your process was created by EXEC, you cannot change the process type (unless you have privilege ?PVTY or the SUPERPROCESS privilege).

If you don't specify a new process type, the CLI displays the process type of the existing process. If you specify a process type, the process will have its type changed to the new type.

If you input PRTYPE with no arguments, your CLI's process type will be displayed.

Command Switches

/1,/2,/L,/L=pathname,/Q
  See CLI Commands tab.

Argument Switches

None.

Example

```
)PROCESS/PREEMPTIBLE SMITH:PROGA)
PID:14
)PRTYPE 14 SWAPPABLE)
)
```

First, create a preemptible son process with program PROGA. The system assigns a new PID of 14. Then, set process 14's type to swappable.

Format

PUSH

Purpose

Descend to a new environment.

PUSH saves the current environment and then pushes a level. You can now change the environment settings SUPERPROCESS, SUPERUSER, SQUEEZE, CLASS1, CLASS2, LISTFILE, DATAFILE, DIRECTORY, SEARCHLIST, STRING, and PROMPT by using the appropriate CLI commands.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/V
Display the new environment's level.

Argument Switches

None.

Example

```
)LEVEL)
LEVEL 0
)PUSH/V)
LEVEL 1
)
```

First display LEVEL, then PUSH to a new environment and verify the new LEVEL.

QBATCH *Command*

Format

QBATCH argument...

Purpose

Create a batch job file in your working directory, and place an entry for it on the batch queue.

The batch job file begins with commands that set the batch job's working directory and search list to the current setting. If you issue QBATCH without the /I or /M switch, the remainder of the command line becomes the batch job. EXEC deletes the job file after the job runs.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/I
The contents of this file will be taken from subsequent lines of the @INPUT file. Terminate the input mode (/I) with a single right parenthesis,), and NEW LINE.

/M
The contents of this file will be taken from subsequent lines of the current macro body. The last line of the macro file must contain a single right parenthesis,), and NEW LINE.

/S
Store the sequence number in STRING, where you can use it as an argument to commands via the !STRING pseudo-macro.

/V
Display the name of the batch job file.

Command Switches (continued)

/AFTER=date:time
Where date:time is in the form dd-mmm-yy:hh:mm:ss. Process this request after a specified future time. You may use a plus sign (+) to specify a relative time for process delay. For example, /AFTER=+12 says don't process until at least 12 hours have passed.

/CPU=~~time~~ hh:mm:ss
Limit CPU time for batch jobs, where time specifies the maximum amount of CPU time which the request can use. You must allow enough time for all processes spawned in the batch job. If you omit this switch, the system assumes one minute of CPU time.

/HOLD
HOLD this entry until you explicitly unhold it with the CLI QUNHOLD command.

/JOBNAME=jobname
The entry is assigned the name jobname which you can use to QHOLD, QUNHOLD, or QCANCEL the job.

/NORESTART
If the system fails while this entry is being processed, do not restart the job.

/NOTIFY
Causes EXEC to send a message back to your console when the queue request is completed.

/OPERATOR
Don't run this job if operator is not present. Use this switch when you submit a batch job containing a MOUNT request.

/QPRIORITY=n
Give this job priority n. 1 is the highest priority and 255 is the lowest. The priority of n cannot be less than the job priority specified in your user profile.

Command Switches (continued)

/QOUTPUT=pathname
Set the generic output file of the batch process to pathname.

/QLIST=pathname
Set the generic list file of the batch process to pathname.

Argument Switches

As required by the batch job.

Examples

```
)QBATCH XEQ MASM FILE3)
QUEUED, SEQ=65 QPRI=128

)QBATCH/I)
))XEQ MASM FILE3)
))XEQ BIND FILE3)
))XEQ FILE3)
))))
)

)QBATCH/V XEQ MASM FILE3)
:UDD:CLJ:?012.CLI.002.JOB QUEUED,
 SEQ=66 QPRI=128
```

012 indicates the process ID of the issuing CLI. 002 is included to make the filename unique; i.e., the next batch command you execute may generate file ?012.CLI.003.JOB.

Format

$$\text{QCANCEL} \begin{Bmatrix} \text{seq-no} \\ \text{jobname} \end{Bmatrix} \begin{bmatrix} \textit{seq-no} \\ \textit{jobname} \end{bmatrix} \ldots$$

Purpose

Cancel a queue entry.

QCANCEL removes the specified entry from the queue to which it was submitted. Note that you cannot cancel an entry once it has started processing. Also note that cancelled entries will continue to appear in the queue listing marked with the I or J flags until the EXEC can actually remove them.

Command Switches

/1,/2,/L,/L=pathname,/Q
 See CLI Commands tab.

Argument Switches

None.

Example

```
)QCANCEL JOB1 )
)
```

Removes the entry for JOB1 from the BATCH_INPUT queue.

QDISPLAY *Command*

Format

QDISPLAY

Purpose

Display queue information about the name and type of each queue maintained by the operating system.

If there are no entries in the queue, the word OPEN will be displayed with the queue name and type. If you omit switches, all queue names and their entries are displayed. Entries preceded by an asterisk are being processed; other entries are preceded by letters which indicate their status. Status letters are:

A	Sequence number held by user (QHOLD command).
B	Jobname held by user (QHOLD command).
C	Sequence number held by operator.
D	Jobname held by operator.
E	Queued by SUPERUSER.
F	User /DELETE switch in effect.
G	User /NORESTART switch in effect.
H	System failure while job was being processed.
I	Cancelled by operator.
J	Cancelled by user (QCANCEL command).
K	/BINARY switch in effect.
L	/OPERATOR switch in effect.
M	/NOTIFY switch in effect.
N	Unexpired /AFTER switch in effect.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/QUEUE=name
Display only the named queue. Permanent queue names are: BATCH_INPUT, BATCH_OUTPUT, and BATCH_LIST. Check with operator for local queue names.

/SUMMARY
List only queue names, types, and entry summaries.

/TYPE=type
Display queues of type only. Queue types are: BATCH, PRINT, PUNCH, PLOT, HAMLET, and RJE80.

Argument Switches

None.

Examples

```
)QDISPLAY)
```

Display information about all queues.

```
)QDISPLAY/TYPE=BATCH/TYPE=PRINT)
```

Display information about the BATCH and PRINT queues.

```
)QDISPLAY/L=QDFILE)
```

Write information about all queues to QDFILE.

QHOLD *Command*

Format

$\text{QHOLD} \left\{ \begin{matrix} \text{seq-no} \\ \text{jobname} \end{matrix} \right\} \left[\left\{ \begin{matrix} \textit{seq-no} \\ \textit{jobname} \end{matrix} \right\} \right] \ldots$

Purpose

Hold a queue entry.

You can UNHOLD an entry with the QUNHOLD command. You can only hold your own entries in a queue.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)QSUBMIT/QUEUE=BATCH_INPUT&)
&)/JOBNAME=JOY)
        .
        .
        .
)QHOLD JOY)
)
```

Hold jobname JOY until a subsequent QUNHOLD command unholds it.

Format

QPLOT pathname...

Purpose

Place an entry on a digital plotter queue.

Note that the file is not plotted, but merely queued to the plotter, so don't delete or modify the file until it is output. Data is always plotted exactly as it appears in the file. EXEC does not record billing parameters.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/S
Store the sequence number in STRING where you can use it as an argument to commands via the !STRING pseudo-macro.

/AFTER=date:time
Where date:time is in the form dd-mmm-yy:hh:mm:ss. Process this request after a specified future time. You may use a plus sign (+) to specify a relative time for process delay. For example, /AFTER=+12 says don't process until at least 12 hours have passed.

/COPIES=n
Produce n copies of the file where n must be in the range 1 to 25. Default is 1.

/DELETE
Delete pathnames after plotting them.

Command Switches (continued)

/FORMS=type
Specify that special forms must be used. Check with your operator for local form types. If you omit this switch, standard forms are used.

/NORESTART
Do not restart the plotting if the system fails while this file is being plotted.

/NOTIFY
Causes EXEC to send a message back to your console upon completion of the queue request.

/QPRIORITY=n
Give this entry the priority n. 1 is the highest priority and 255 is the lowest priority; n cannot be less than the priority specified in your user profile.

/QUEUE=queuename
Submit job to specified queuename instead of default queue.

Argument Switches

None.

Examples

```
)QPLOT FILE1 FILE2)
```

Send FILE1 and FILE2 to a digital plotter output queue.

```
)QPLOT/DELETE FILE3)
```

Plot FILE3, then delete FILE3 when finished.

Format

QPRINT pathname...

Purpose

Place the file named in pathname in the line printer output queue.

Note that the file is not printed, but merely queued to the printer, so don't delete or modify it until it is output.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/S
Store the sequence number in STRING where you can use it as an argument to commands via the !STRING pseudo-macro.

/AFTER=date:time
Where date:time is in the form dd-mmm-yy:hh:mm:ss. Process this request after a specified future time. You may use a plus sign (+) to specify a relative time for process delay. For example, /AFTER=+12 says don't process until at least 12 hours have passed.

/COPIES=n
Produce n copies of the file; n must be in the range 1 to 25.

/DELETE
Delete files after printing them.

/FOLDLONGLINES
Do not truncate long lines; continue them on the next line of the listing.

Command Switches (continued)

/FORMS=type
Print on special forms. Check with your operator for local form types.

/NORESTART
Do not restart the listing if the system fails while this file is being printed.

/NOTIFY
Causes EXEC to send a message back to your console upon completion of the queue request.

/PAGES=n
Your operator will tell you whether or not to specify /PAGES. If you do, specify n as the maximum number of pages that will be printed. Use this switch if you specify the /COPIES switch.

/QPRIORITY=n
Give this entry the priority n. 1 is the highest priority and 255 is the lowest; n cannot be less than the priority specified in your user profile.

/QUEUE=queuename
Submit job to specified queuename instead of default queue.

/TITLES
Print each page with a title line consisting of pathname, date, time, and page number.

Argument Switches

None.

Examples

```
)QPRINT FILE1 FILE2)
QUEUED, SEQ=28, QPRI=127
)
```

Queue FILE1 and FILE2 to the line printer.

```
)QPRINT/DELETE/FOLDLONGLINES FILE3)
QUEUED, SEQ=35, QPRI=137
)
```

Print FILE3, folding long lines. Delete FILE3 when complete.

```
)QPRINT/COPIES=10 MYFILE)
QUEUED, SEQ=411, QPRI=110
)
```

Print 10 copies of MYFILE.

Format

QPUNCH pathname...

Purpose

Place the file named in pathname in a paper tape punch queue.

Note that the file is not punched, but merely sent to the punch queue, so don't delete or modify the file until it is output.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/S
Store the sequence number in STRING, where you can use it as an argument to commands via the !STRING pseudo-macro.

/AFTER=date:time
Where date:time is in the form dd-mmm-yy:hh:mm:ss. Process this request after a specified future time. You may use a plus sign (+) to specify a relative time for process delay. For example, /AFTER=+12 says don't process until at least 12 hours have passed.

/COPIES=n
Produce n copies of the file; n must be in the range 1 to 25.

/DELETE
Delete pathnames after punching them.

/FEET=n
Where n is the maximum number of feet of tape that will be punched. Use this switch if you specify multiple copies.

Command Switches (continued)

/FORMS=type
Specify that special forms must be used. Check with your operator for local form types.

/NORESTART
Do not restart the listing if the system fails while this file is being punched.

/NOTIFY
Causes EXEC to send a message back to your console upon completion of the queue request.

/QPRIORITY=n
Give this entry the priority n. 1 is the highest priority and 255 is the lowest; n cannot be less than the priority specified in your user profile.

/QUEUE=queuename
Submit job to specified queuename instead of default queue.

Argument Switches

None.

Examples

```
)QPUNCH FILE1 FILE2)
)
```

Send FILE1 and FILE2 to a paper tape output queue.

```
)QPUNCH/COPIES=3/FEET=75 FILE4)
)
```

Punch three copies of FILE4. Do not punch more than 75 feet of paper tape.

Format

QSUBMIT *[pathname]...*

Purpose

Place an entry on a batch or spool queue for each *pathname* you supply in the argument list.

If you omit the /QUEUE switch, the BATCH input queue is assumed.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/S
Store the sequence number in STRING, where you can use it as an argument to commands via the !STRING pseudo-macro.

/AFTER=date:time
Where date:time is in the form dd-mmm-yy:hh:mm:ss. Process this request after a specified future time. You may use a plus sign (+) to specify a a relative time for process delay. For example, /AFTER=+12 says don't process until at least 12 hours have passed.

/CPU=time
Limit CPU time for batch jobs, where time specifies the maximum amount of CPU time which can be used for the request; time is in the form mm: *[ss]*. If you omit this switch, the system assumes one minute of CPU time.

Command Switches (continued)

/DELETE
Delete the *pathname(s)* after processing.

/HOLD
BATCH queue only. Hold this entry until you unhold it with the QUNHOLD command.

/JOBNAME=jname
BATCH queue only. Give this entry the name jname which you can use to QHOLD, QUNHOLD, or QCANCEL a job.

/NORESTART
If the system fails while this entry is being processed, don't restart the job.

/NOTIFY
Causes EXEC to send a message back to your console upon completion of the queue request.

/OPERATOR
Don't run this job if operator is not present. Use this switch when you submit a BATCH job containing a MOUNT request.

/QPRIORITY=n
Give this job priority n. 1 is the highest priority and 255 is the lowest; n cannot be less than the job priority specified in your user profile.

/QUEUE=queuename
Submit job to the specified queuename. If you omit this switch, BATCH_INPUT is assumed.

/XW0=
/XW1=
/XW2=
/XW3=
} Reserved for special applications.

Argument Switches

None.

Examples

```
)QSUBMIT FILE1 FILE2)
)
```

Submit FILE1 and FILE2 to the BATCH_INPUT queue.

```
)QSUBMIT/AFTER=12:30 FILE4)
```

FILE4 will not be processed until after 12:30.

Format

$$\text{QUNHOLD} \quad \left\{ \begin{array}{l} \text{seq-no} \\ \text{jobname} \end{array} \right\} \quad \left[\left\{ \begin{array}{l} \textit{seq-no} \\ \textit{jobname} \end{array} \right\} \right]$$

Purpose

Free a held queue entry.

This command negates a previous QHOLD command. Note that you cannot QUNHOLD an entry held by the operator. If you QHOLD a BATCH_INPUT entry by jobname, then you should QUNHOLD it by jobname. However, you can QUNHOLD any BATCH_INPUT entry by sequence number.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Examples

```
)QHOLD MYJOB)
)
      .
      .
      .
)QUNHOLD MYJOB)
```

Unhold entry MYJOB. If there is an operator hold on MYJOB, or if the entry was held on the BATCH_INPUT queue by sequence number, the command will return with an exceptional condition.

Format

There are five functions of the RDOS utility: LOAD, DUMP, GET, PUT, and LIST. Each function and its format is listed below.

```
XEQ RDOS LOAD rdos-dumpfile [pathname(s)]

XEQ RDOS DUMP rdos-dumpfile [filename(s)]

XEQ RDOS GET/DISK=rdos-diskunit pathnames

XEQ RDOS PUT/DISK=rdos-diskunit
  [/DIR=rdos-directory] filename(s)

XEQ RDOS LIST/DISK=rdos-diskunit pathnames
```

Purpose

Read or write an RDOS dump file on disk. You can use AOS templates in any command.

The LOAD command loads one or more RDOS files from an RDOS-format dump file. The DUMP command dumps one or more data or program files to an RDOS-format dump file. You cannot dump links, directories, and file types other than user programs and data. The GET command gets one or more files from the primary or secondary partition or subdirectory of an RDOS disk. The PUT command puts one or more files on an RDOS format disk. You can put files into one subdirectory or partition only. The LIST command lists the names of files on an RDOS format disk.

Note that the GET, PUT, and LIST commands assume that an RDOS disk is on your system. If you omit the optional *rdos-directory* specifier in the GET, PUT, or LIST commands, they will access the primary partition of the RDOS disk.

RDOS Switches

/A
Abort on an ABORT condition.

/D
(LOAD only.) If a file exists with the same name, delete it, then load the new one.

/DIR=rdos-directory
(PUT only) Access this RDOS directory.

/DISK=rdos-diskunit
Access this RDOS disk in this unit.

/L
List files on LISTFILE.

/L=pathname
List files on file specified by pathname.

/N
(LOAD and GET only.)
Do not load; just verify filenames.

/T
(GET and LIST only.)
Access all files in the specified *rdos-directory* and its subordinate directory (if any).

/V
Verify each file transferred.

Argument Switches

/C
On LOAD and GET, convert carriage returns to NEW LINEs; on DUMP and PUT, convert NEW LINEs to carriage returns.

/N
Do not transfer files matching this template.

Examples

```
)XEQ RDOS LOAD @MTA0:0 +.SR/C +.OB)
```

Load all files ending in .SR and .OB from file 0 on MTA0 to the working directory. Convert all carriage returns in the source files (.SR) to NEW LINEs.

```
)XEQ RDOS DUMP/V @MTA0:1 +/C)
```

Dump all files in the working directory to file 1 of MTA0, and convert all source file NEW LINEs to carriage returns. Also, list the dumped filenames on @OUTPUT.

```
)XEQ RDOS GET/DISK=@DPD1/T DIR1:+/C)
```

Get all the files in rdos-directory DIR1 of the RDOS disk in DPD1. If DIR1 has subdirectories, get its files as well. /C converts all carriage returns to NEW LINEs.

```
)XEQ RDOS PUT/DISK=@DPD0/DIR=DIR2)
```

Write all files in the working directory to RDOS directory DIR2, in the RDOS disk in DPD0.

```
)XEQ RDOS LIST/L=@LPT/DISK=@DPD0/T)
```

List information about every filename on the disk to the line printer.

Format

XEQ RDOSBIND pathname...

Purpose

Produce an RDOS save file from one or more AOS object or library files.

RDOSBIND requires object files and libraries as input in AOS format. You must use the AOS Macroassembler and Library File Editor to prepare input to RDOSBIND.

RDOSBIND Switches

/B
Sort symbol listings alphabetically and by value.

/C
Bind a program for RTOS.

/D
Load an RDOS user debugger and enter global switches in the save file.

/E
Produce a load map on @OUTPUT.

/G=n
Designate that n channels are required.

/H
Print numbers in hexadecimal.

/K=n
Allocate n TCBs. (This overrides a .TSK pseudo-op.)

RDOSBIND Switches (continued)

/L
Produce a listing on the current @LIST file.

/L=pathname
Produce a listing on the file designated by pathname.

/N
Do not search the system library (ASYOB.LB).

/O
Suppress load overwrite error messages (if any).

/P=x
Create a save file named x.SV.

Argument Switches

x/C
Specify a command file named x. You must include this switch if your program will generate overlays.

/O
Suppress load overwrite messages for this module.

/V
Designate a virtual node for use in RLDR.

n/Z
Set the ZREL location counter to n (unless the current ZREL counter is greater than n).

Example

```
)XEQ RDOSBIND FILEZ)
)
```

Produce an RDOS save file from the AOS file called FILEZ.

Format

[!READ argument...]

Purpose

Display text on @OUTPUT and expand to one or more arguments from @INPUT.

Macroname Switches

None.

Argument Switches

None.

Example

```
)DELETE/V [!READ DEL WHICH FILE(S)?])
DEL WHICH FILE(S)? FILEA)
DELETED FILEA
)
```

The CLI writes the message on the console. The file(s) you type in response are deleted. Each deletion is verified on the console.

RELEASE *Command*

Format

RELEASE logical-disk *[...]*

Purpose

Release a previously initialized logical disk (LD) from the working directory.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)RELEASE ALPHA)
)
```

Release an LD named ALPHA that you previously initialized.

Format

RENAME pathname new-filename

Where new-filename must be a simple filename.

Purpose

Change a file's name.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)FILESTATUS)
DIRECTORY :UDD:USER
FILEU CODEA
)RENAME FILEU FILEME)
)FILESTATUS)
DIRECTORY :UDD:USER
FILEME CODEA
)
```

Format

XEQ REPORT *[pathname]...*

Purpose

Print the contents of the SYSLOG log file or a file created by the SYSLOG function.

This is a privileged utility that only the operator process may issue.

Referral

For information on the REPORT utility, see the *AOS Operator's Guide.*

Format

REVISION pathname *[major number.minor number]*

You may use templates in the pathname argument. Major and minor revision numbers can range from 0 to 255.

Purpose

Set or display a program's revision number.

You set the revision level with the .REV pseudo-op in the assembly language source program.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/V
Display the filename with the revision number.

Argument Switches

None.

Examples

```
)REVISION MIKE)
00.00
)REVISION/V MIKE 0.1)
MIKE 00.01
)
```

First, display the revision number of a program file named MIKE in the working directory; then change the revision and display the new one.

REWIND *Command*

Format

```
REWIND  { tapeunit } ...
        { linkname }
```

You must specify either the same linkname(s) you used to MOUNT the volume(s) or the device name on which it is mounted. You may use templates in the tapeunit and linkname arguments.

Purpose

Rewind one or more tapes.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Examples

```
)REWIND @MTA0)
)
```

Rewind the magnetic tape on unit @MTA0.

```
)MOUNT TAPE10 MOUNT_IT_AGAIN_SAM)
      .
      .
      .
)REWIND TAPE10)
)
```

First, request the operator to mount a tape and create a link named TAPE10 to that tape. After performing required operations, rewind TAPE10.

Format

XEQ RPG *[switches]* filename *[listfile/argument switches]*

Purpose

Compile an RPG II source file.

For a complete description of the RPG II programming language and the CLI RPG II command line, see the *RPG II Programmer's Reference Manual* (093-000117).

RPG Switches

You may use more than one switch at a time. To use the Debugger or Analyzer you must specify the /D switch.

Note that if you use the /D switch and the /L switch together, you will have an indicator cross reference as well as a debugger file and a source listing.

/C
: Use of this switch will list the compiler-generated object code to the listing file, following the source listing if you requested one. The object code listing includes line numbers of lines for which specific code was generated.

RPG Switches (continued)

/D

This switch imbeds Debugger and Analyzer information in the object file. This switch produces the same effect as a D in column 15 of the Control Card specification, except that it does not enable DEBUG calculation operations. You must specify this switch if you intend to use the interactive RPG II Debugger or Analyzer.

/D/L

If you give the /D and /L switches together, you get a Debugger and Analyzer file, a source listing, and an indicator cross reference. For example,

```
INDICATOR SUMMARY

NAME   REFERENCES

10      5=   7
LR      8=
3 INDICATOR REFERENCES
```

In the above summary, program line 5 can turn indicator 10 ON or OFF and line 7 will be executed depending upon its state. Line 8 turns indicator LR ON or OFF.

/E

This switch suppresses both page ejection and a new header when the system encounters a comment specification with asterisks in columns 7 and 8. It lets you run RPG II programs that use comments consisting of lines of asterisks as separators.

RPG Switches (continued)

/I

This switch inhibits conditioning indicator code optimization. This is useful for debugging, in that repetitive patterns of conditioning indicators are replaced with a single test and branch. Thus:

```
00002  NO2              1ADD  X       X
00003  NO2              1MOVEZ
00004  NO2  "TOT"       MOVE  L
```

is converted into the functional equivalent of:

```
00001  02               GOTO  LOOP
00002                   1ADD  X       X
00003                   1MOVEZ
00004       "TOT"       MOVE  L
00005       LOOP        TAG
```

Thus, if you set a breakpoint at line 3 and, upon reaching that point, indicator 2 is turned ON, execution will continue unaffected until line 5 if optimization is present.

/L

This switch lists the source file to the listing file. If you don't specify a listing file, the source file is listed to the line printer.

/N

This switch suppresses the printing of notes on the compiler listing. If you do not specify this switch, the compiler prints notes. The /N switch does not suppress warning, error, or fatal error messages.

Argument Switches

/L
Send listing output to file specified by *listfile.*

/O
Name the RPG program file with the name specified in *listfile.*

Examples

```
)RPG/L MYPROG⤶
```

This command compiles the program named MYPROG and sends any error messages and a source listing to the list file - in this case, the line printer.

```
)RPG MYPROG ERRORS/L⤶
```

This command compiles the program named MYPROG and sends error messages to the list file - in this case, a text file named ERRORS.

```
)RPG/L MYPROG PROGLIST/L⤶
```

This command compiles MYPROG and sends a source listing and errors to the list file - in this case, a file named PROGLIST.

```
)RPG/L MYPROG YOURPROG/O⤶
```

This command compiles MYPROG and lists errors and source text to the list file - in this case, @LPT; it also names the executable program YOURPROG.

Format

$$\text{RUNTIME} \left[\begin{Bmatrix} \textit{username:procname} \\ \textit{processID} \end{Bmatrix}\right]$$

Purpose

Display the following runtime information about the specified process:

ELAPSED
Real-time elapsed since this process was created.

CPU
Central processor time used by process.

I/O BLOCK
Number of blocks of data read or written by this process.

PAGE MSECS
Number of memory pages (in 2K bytes) used by this process, multiplied by CPU time in milliseconds.

If you omit the argument, the CLI displays its own runtime information.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Example

```
)RUNTIME 7)
ELAPSED 21:44:09, CPU 0:01:47.008
I/O BLOCKS 927, PAGE MSECS 1728824
)
```

Display runtime information about process 7.

Format

XEQ SCOM sourcefile$_1$ sourcefile$_2$

Purpose

Compare two ASCII text files.

SCOM scans each line from both files. If it finds differences, it outputs either the difference or a message (see SCOM switches below). The program then attempts to get back into synchronization. Synchronization is defined as finding n lines in a row that match (where n is called *matchsize)*. By default, the matchsize is set to four.

SCOM Switches

/L
Write a list of differences to current @LIST file. If you omit this switch, the program doesn't list the the differences. Instead, it returns the message, FILES DIFFER STARTING AT LINE xxx/xxx, or nothing if the files match.

/L=filename
Write differences to the specified filename.

/EOL
The end-of-line character is treated as significant. If you omit this switch, the system ignores EOL characters and blank lines.

/MS=number
Set matchsize to the specified number. If you omit this switch, the default is four.

Argument Switches

None.

Examples

```
)XEQ SCOM MYFILE YOURFILE)
)
```

Compare MYFILE and YOURFILE. In this case, since no information was output, the files are identical.

Format

SCREENEDIT $\left\{\begin{matrix} ON \\ OFF \end{matrix}\right\}$

Purpose

Set or display the SCREENEDIT setting. When SCREENEDIT is on you may modify the current line using cursor control characters. (The control characters are also available in LINEDIT.) The following control characters may be used:

Control Character	Effect on the Cursor
CTRL-A	Move to the end of the character string.
CTRL-B	Move to the end of the previous word.
CTRL-E	Enter/exit the insert character mode.
CTRL-F	Move to the beginning of the next word.
CTRL-H	Move to the beginning of the character string.
CTRL-I	Insert a tab.
CTRL-K	Erase everything right of the cursor.
CTRL-X	Move to the right one character. (The → key on the function key pad has the same effect.)
CTRL-Y	Move to the left one character. (The ← key on the function key pad has the same effect.)
RUBOUT	(delete the previous character), and CTRL-U (delete the line) perform as usual.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/P
Set SCREENEDIT to the previous environment's SCREENEDIT setting.

Argument Switches

None.

Examples

```
)SC ON)
)ANT CHARACTER STRING. (CTRL-H) (CTRL-X...)Y)
)
```

Typing CTRL-H returns the cursor to the beginning of the string. Typing CTRL-X twice positions the cursor at the T in the first word. After the string is corrected by replacing the T with Y, it reads:

```
)ANY CHARACTER STRING
```

Format

SEARCHLIST *[pathname]...*

Purpose

Set or display the SEARCHLIST setting.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/K
Delete the current search list, if any (no arguments allowed).

/P
Set search list to search list of previous environment (no arguments allowed).

Argument Switches

None.

Examples

```
)PUSH)
)SEARCHLIST)
:PER,:UTIL,:
)SEARCHLIST :UDD:HENRY,:PER,:UTIL,:)
)SEARCHLIST)
:UDD:HENRY,:PER,:UTIL,:
)SEARCHLIST/P)
:PER,:UTIL,:
)
```

Push a level and display current SEARCHLIST. Change current SEARCHLIST and display current and previous SEARCHLIST.

Format

[!SEARCHLIST]

Purpose

Expand to the search list.

Macroname Switches

/P
Use the search list of the previous environment.

Argument Switches

None.

Example

```
)SEARCHLIST :UDD:MDIR, [!SEARCHLIST])
```

This command sets the search list. The CLI evaluates the pseudo-macro !SEARCHLIST, then sets the SEARCHLIST to the resulting argument string.

Format

```
        ( processID            )
SEND   <  username:procname    >   message
        ( consolename          )
```

Purpose

Send a message to a process's console.

The target process can be your CLI process, another user process, or the operator process (PID2). The procname (i.e., USERNAME: PROCESSNAME) or processID can be either a simple process or a complete process. The consolename must begin with the @ prefix.

If you send a message which the target process doesn't receive, that process may have message reception disabled.

Do not try to include commas, tabs, or control characters in your messages. Commas and tabs become spaces when the system sends your message. The system can not send control characters.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/I
Send each input line that follows as a separate message. You must end the /I sequence with a single).

/M
Send each line of the current macro file as a separate message. End the /M sequence with a single).

Argument Switches

None.

Examples

```
)SEND 2 PLEASE BRING UP LPT1 )
)
```

Send a message to the system operator.

```
)SEND @CON- SYSTEM WILL SHUT DOWN& )
&)AT MIDNIGHT )
)
```

Send a message to all consoles that are running a process.

```
)
FROM PID 8: IS IT OK FOR ME TO PRINT A LONG
   MANUSCRIPT?
)SEND 8 GO AHEAD )
)
```

Process 8 sends a message and you reply.

Format

```
XEQ SLB/O=shared-routine-name library number
  program-name...
```

Purpose

Build a shared library routine from one or more executable programs.

The SLB searches for program-names(s) with the .PR extension, but you need not type the extension; it always appends the extension .SL to shared-routine-name. The library number is a number from 2 through 63 that you want the new routine to have. (Numbers 0 and 1 are system reserved.)

SLB Switches

/L
  Produce listing to current @LIST file.

/L=pathname
  Produce a listing to file pathname.

/O
  This is the output filename; /O is a mandatory switch.

Argument Switches

None.

Example

```
)XEQ SLB/L=@LPT/O=GEOMETRY 5&)
&)SINE COSINE TANGENT)
```

Create shared library GEOMETRY.SL from executable programs SINE.PR, COSINE.PR, and TANGENT.PR. GEOMETRY.SL is a shared routine; hence, many users can access it from their own programs.

SPACE *Command*

Format

SPACE $\left[\left\{ \begin{array}{l} \textit{control-point-directory [new-max-size]} \\ \textit{logical-disk} \end{array} \right\} \right]$

Purpose

Set or display the amount of disk space in a control point directory or logical disk.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/V
Display the control point directory or logical disk name.

Argument Switches

None.

Examples

```
)SPACE/V⤶
= MAX 1000, CUR 499, REM 501
)
```

Display the space in =, the working directory.

```
)SPACE□:⤶
MAX 37000,CUR 18000,REM 19000
)
```

Display disk space in :, the root directory.

```
)SPACE□=□10000⤶
```

Set the maximum size of the current working directory.

Format

XEQ SPEED *[pathname]*

Purpose

Write or edit ASCII text.

SPEED is a text editor. You can use it to create a new source file or to edit an existing source file. If you specify pathname and the file exists, SPEED reads the first page into the edit buffer. If the file does not exist, SPEED creates and opens the file for editing.

A summary of SPEED commands appears later in this manual. See Editors.

For more information on SPEED, see the *AOS SPEED Text Editor User's Manual.*

SPEED Switches

/I=pathname
Take input from pathname, not from the console keyboard.

Argument Switches

None.

Example

```
)XEQ SPEED MIKESFILE)
SPEED REV x.x
!
```

! is the SPEED prompt. Now you are ready to enter text.

Format

SQUEEZE $\left[\left\{ \begin{array}{c} ON \\ OFF \end{array} \right\} \right]$

Purpose

Set or display the SQUEEZE setting.

When SQUEEZE is ON, the CLI outputs each sequence of two or more tabs or spaces as a single space. Output from the TYPE command is never squeezed. You can turn SQUEEZE ON for any CLI command by appending the /Q switch to the command name.

Command Switches

/1,/2./L,/L=pathname,/Q
See CLI Commands tab.

/P
Set SQUEEZE mode to previous environment's SQUEEZE.

Argument Switches

None.

Examples

```
)SQUEEZE)
OFF
)SQUEEZE ON)
)SQUEEZE)
ON
)
```

First, display the current SQUEEZE setting; then set SQUEEZE to ON and display the new SQUEEZE setting.

Format

STRING *[argument]...*

Purpose

Set or display the STRING setting.

The STRING buffer can hold up to 127 characters.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/K
Set STRING to null (no arguments allowed).

/P
Set STRING to previous environment's STRING (no arguments allowed).

Argument Switches

None.

Examples

```
)STRING)
THIS,IS,A,STRING
)STRING NEW ONE)
)STRING)
NEW,ONE
)STRING/K)
)STRING)

)
```

First, display STRING; then set it to a new string. STRING/K kills the current STRING; therefore, the last command returns a null line.

Format

[!STRING]

Purpose

Expand to the STRING setting.

Macroname Switches

/P
 Returns the previous environment's STRING.

Argument Switches

None.

Examples

```
)WRITE THE CURRENT STRING IS [!STRING])
THE CURRENT STRING IS CURRENT STRING
)WRITE THE PREVIOUS STRING IS &)
&)[!STRING/P])
THE PREVIOUS STRING IS PREVIOUS_STRING
)
```

Format

SUPERPROCESS $\left[\begin{Bmatrix} ON \\ OFF \end{Bmatrix}\right]$

Purpose

Set or display the SUPERPROCESS setting.

Only privileged users may set SUPERPROCESS to ON. The CLI precedes each prompt with a plus sign (+) when you have SUPERPROCESS set to ON (or # if both SUPERPROCESS and SUPERUSER are ON).

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/P
Set SUPERPROCESS setting to the previous environment's SUPERPROCESS setting.

Argument Switches

None.

Examples

```
)SUPERPROCESS)
OFF
)SUPERPROCESS ON)
+)SUPERPROCESS)
ON
+)
```

First, display the current SUPERPROCESS setting; then set SUPERPROCESS to ON and display the new setting.

Format

$$\text{SUPERUSER}\left[\begin{Bmatrix} ON \\ OFF \end{Bmatrix}\right]$$

Purpose

Set or display the SUPERUSER setting.

Only privileged users may set SUPERUSER to ON. The CLI precedes each prompt with an asterisk (*) when you have SUPERUSER set to ON (or # if both SUPERUSER and SUPERPROCESS are set to ON).

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/P
Set current SUPERUSER to previous environment's SUPERUSER setting.

Argument Switches

None.

Examples

```
)SUPERUSER)
OFF
)SUPERUSER ON)
*)SUPERUSER)
ON
*)
```

Display the current SUPERUSER setting; then set SUPERUSER ON and display the new setting.

Format

SYSLOG *[filename]*

Purpose

Start or stop writing to the system usage log file.

This is a privileged command; only the initial CLI process (PID2) may issue the SYSLOG call.

The following information is written to the system log file:

- Information about system users such as when they log on, when they log off, what devices they used, CPU usage, and the size of main memory allocated to them.
- Information about peripheral devices, such as type and number of errors they encounter.

For more information about SYSLOG, refer to the *AOS Operator's Guide.*

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/START
Start the system log.

/STOP
Stop the system log.

Argument Switches

None.

Examples

```
)SYSLOG/START)
)
```

Start recording the system log file.

```
)SYSLOG)
ON
```

SYSLOG with no arguments returns the current state of the log on.

Format

```
TERMINATE { username:procname }
          { processID         }
```

Purpose

Terminate an inferior process.

You must supply the procname or processID of the inferior process (or any process if you have the SUPERPROCESS privilege). The processID can be either a simple process or a complete process.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/BREAKFILE *[=pathname]*
Produce a break file (an exact disk image of a process's address space in main memory at the time of termination) in the working directory. If you invoke TERMINATE with the simple /BREAKFILE switch, the system creates a break file with a default name. If you invoke TERMINATE with /BREAKFILE=pathname, the break file will have the specified name.

Argument Switches

None.

Example

```
)PROCESS SMITH:PROGZ)
PID 17
       .
       .
       .
)TERMINATE 17)
)
```

First, create a swappable son process which runs concurrently with the CLI. The CLI displays the PID of the new process. The last command terminates the process.

Format

TIME *[new-time]*

Only the operator (PID 2) can set the time. *New-time* is in the form hh:mm:ss, where minutes and seconds are optional and colons or spaces can separate entries.

Purpose

Set or display the current system time.

Command Switches

/1,/2,/L,/L=pathname,/Q
 See CLI Commands tab.

Argument Switches

None.

Examples

```
)TIME)
19:30:40
)TIME 8:45)
)
```

The first command displays the system time. The second command, executed by the operator, sets the time to 8:45 a.m.

!TIME

Format

[!TIME]

Purpose

Expand to the current system time.

Macroname Switches

None.

Argument Switches

None.

Example

```
)WRITE IT IS NOW [!TIME])
IT IS NOW 11:22:25
)
```

Format

$$\text{TREE} \left[\begin{Bmatrix} \textit{username:procname} \\ \textit{processID} \end{Bmatrix} \right] \textit{[argument]} \ldots$$

Purpose

Display a process's father and son(s) (if any).

If you omit the argument, the CLI's tree is displayed.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/V
Display the title and type of the file before typing it. If the record type is fixed, the CLI also displays the record length.

Argument Switches

None.

Examples

```
)TREE 7)
PID: 7 FATHER: 4 SONS:8 12 13
)TREE OP:EXEC)
PID: 4 FATHER: 2 SONS:7 9 10 11
)
```

The first command displays the tree of PID 7. The second command displays EXEC's tree.

Format

TYPE pathname...

Purpose

Type the contents of a file.

Unless you include the /L switch, the CLI types the file to the generic @OUTPUT file. SQUEEZE mode does not affect output from TYPE.

Command Switches

/1,/2,/L,/L=pathname
See CLI Commands tab.

/V
Display the name and record type of the file before typing it. If the record type is fixed, the CLI also displays the record length.

Argument Switches

None.

Examples

```
)TYPE MYFILE)
      .
      .  (MYFILE is displayed on your console.)
      .
)TYPE/2=ERROR FILE1 FILE2 FILE3)
      .
      .
      .
)
```

In the second command we set CLASS2 exceptional conditions to ERROR. If any of the named files do not exist, the CLI will display an ERROR message and processing will stop. Files appearing to the right of the nonexistent filename will not be typed.

UNBLOCK *Command*

Format

UNBLOCK $\left[\left\{\begin{array}{l}\textit{username:procname} \\ \textit{processID}\end{array}\right\}\right]$

You must supply either the procname or the processID. To be unblocked, the process must be a previously blocked inferior process (unless you have the SUPERPROCESS privilege). The processID can be either a simple process or a full process.

Purpose

Unblock a previously blocked inferior process.

Command Switches

/1,/2,/L,/L=pathname,/Q
 See CLI Commands tab.

Argument Switches

None.

Example

```
)PROCESS SMITH:PROGA)
PID 17
)BLOCK 17)
      .
      .
      .
)UNBLOCK 17)
)
```

First, create an inferior swappable process that runs concurrently with the CLI. Then, block the new process, perform required operations, and finally, unblock the new process.

Format

[!USERNAME]

Purpose

Expand to the CLI's username.

Macroname Switches

None.

Argument Switches

None.

Example

```
)WRITE CALL ME [!USERNAME])
CALL ME ISHMAEL
)
```

Format

WHO [{ *username:procname* / *processID* }] *[argument]* ...

Purpose

Display process information.

This command displays the PID, username, process name, and program name of one or more processes.

If you omit argument(s), the command displays process information of the CLI.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Examples

```
)WHO)
PID17: KAREN CON13 :UTIL:CLI.PR
```

The current process's ID is 17, its username is KAREN, its process name is CON 13, and the program it runs is CLI.PR.

```
)WHO 007)
PID:7 BARB_Z CON7 :UTIL:CLI.PR
```

Format

WRITE *[argument]...*

Purpose

Display arguments.

The WRITE command is useful in macros either for writing to the console to explain what is happening, or writing a record/log to a list file saying what has happened or is happening in a program. WRITE is also useful when you work with CLI command line operators such as parentheses and angle brackets.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

Argument Switches

None.

Examples

```
)WRITE WE SHALL NOT CEASE FROM&)
&)EXPLORATION)
WE SHALL NOT CEASE FROM EXPLORATION
)
```

```
)WRITE (A□B)_<X Y>)
A_X A_Y
B_X B_Y
)
```

Format

XEQ program-name *[arguments]*

Purpose

Execute a program.

While executing the program, the CLI is normally blocked until the subordinate process terminates. XEQ is identical to EXECUTE or X.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Commands tab.

/I
Create input for program from @INPUT. You must end the /I sequence with a single).

/M
Create input for program from macro body. You must end the /M sequence with a single).

/S
Return program's termination message to STRING instead of @OUTPUT.

Argument Switches

As needed by the new program.

Examples

```
)XEQ BIND OBJ1)
```

Invoke the Binder Utility to bind OBJ1.

```
)XEQ/S PROG2)
```

Execute PROG2 and return PROG2's termination message to STRING instead of @OUTPUT.

CLI Messages

CLI Exceptional Condition Messages

We list CLI exceptional condition messages alphabetically.

/AFTER OR /BEFORE SWITCH REQUIRED

You used the /TLA or /TLM switch without using the /BEFORE or /AFTER switch.

ARGUMENT IS NOT A COMMAND

You used an invalid argument in PROMPT command.

ARGUMENT IS NOT A UNIQUE ABBREVIATION

An argument in PROMPT command is an invalid abbreviation.

ARGUMENT MAY NOT HAVE SWITCHES

You used invalid switches in the PROMPT command.

ARGUMENT MAY NOT BE A NONIMPLEMENTED COMMAND

An argument in a PROMPT command has not yet been implemented.

ARGUMENT MAY NOT BE A COMMAND REQUIRING ARGUMENT(S)

An argument in a PROMPT command requires arguments and therefore, it is an invalid argument to PROMPT.

CAN'T POP FROM LEVEL 0

You issued the POP command from LEVEL 0.

COMMAND ABBREVIATION NOT UNIQUE

You used an invalid command name.

COMMAND DOES NOT ACCEPT ARGUMENTS

You supplied arguments to a command that does not accept arguments.

COMMAND NOT IMPLEMENTED

The CLI code defining this command has not been implemented.

COMMAND REQUIRES ARGUMENT(S)

You didn't supply arguments to a command that requires arguments.

CONFLICTING SWITCHES

One or more command switches contradicts another.

CONSOLE INTERRUPT

You've interrupted the process which controls the console with a CTRL-C CTRL-A control character sequence.

CONSOLE INTERRUPT TASK STACK OVERFLOW

This is a system error; see your system manager.

EXTRANEOUS [!END]

You typed too many !END pseudo-macros in a macro.

FILE NOT A CONTROL POINT DIRECTORY

You issued the SPACE command on a file that is not a control point directory.

ILLEGAL DECIMAL NUMBER

The argument must be an unsigned positive, decimal number.

.LEGAL FORMAT DUMMY ARGUMENT IN MACRO

The format of a dummy argument in a macro is invalid.

ILLEGAL FILENAME TEMPLATE

A filename specification in a command line is invalid.

ILLEGAL REVISION NUMBER

The argument to REVISION command is invalid.

ILLEGAL OCTAL NUMBER

The argument must be an unsigned positive, octal number.

ILLEGAL SEVERITY LEVEL

The argument to the CLASS1 and CLASS2 commands and the value of /1 and /2 command switches must be IGNORE, WARNING, ERROR, or ABORT.

INVALID TIME FORMAT

You entered an invalid argument to the TIME command or a /TLA or /TLM command switch.

INVALID DATE FORMAT

You entered an invalid argument to the DATE command or a /TLA or /TLM command switch.

INDECIPHERABLE DUMP FORMAT

The format of a dump file is invalid.

MESSAGE TOO LONG

The argument string to a SEND command is too long.

MISSING [!END]

You forgot to include !END in a macro.

MISMATCHED BRACKET TYPES

You've mismatched different types of brackets.

NO MACRO INPUT AVAILABLE

You forgot to include the single) after the /M switch in a macro.

NO PREVIOUS VALUE WHEN AT LEVEL 0

You issued the PREVIOUS command or the /P command switch when you were at LEVEL 0.

NOT A COMMAND OR MACRO

You began the command line with an illegal item.

****NOT ENOUGH MEMORY, RESTARTING CLI****

The CLI required more memory than it was allocated. DIRECTORY and SEARCHLIST remain the same, but all other environment parameters are initialized.

****NOT ENOUGH MEMORY TO START UP CLI****

There may be an error in your user profile. See your system manager.

OCCURRED DURING PROMPT EXECUTION,
  PROMPT INITIALIZED

An exceptional condition occurred when the CLI executed a command in the PROMPT buffer. PROMPT is set to null.

PARENTHESIS NOT ALLOWED IN A MACRO NAME

You typed a parenthesis within a macroname.

PATHNAME MUST START FROM WORKING
  DIRECTORY

You specified an invalid pathname in a LOAD, MOVE, or DUMP command.

PATHNAME TOO LONG

Pathnames can't exceed 128 characters.

PROCESS number TERMINATED BY CONSOLE
  INTERRUPT

You've aborted the process in control of the console with a CTRL-C CTRL-B control character sequence.

PSEUDO MACRO UNKNOWN

You entered an invalid pseudo-macro.

PSEUDO MACRO ABBREVIATION NOT UNIQUE

You used an invalid pseudo-macro name.

PSEUDO MACRO HAS WRONG NUMBER OF
  ARGUMENTS

You supplied the wrong number of arguments in a pseudo-macro; one or more arguments may have been evaluated to null.

PSEUDO MACRO NOT IMPLEMENTED

The CLI code defining this pseudo-macro has not been implemented.

PSEUDO MACRO REQUIRES ARGUMENTS

You didn't supply arguments to a pseudo-macro that requires arguments.

PSEUDO MACRO DOES NOT ACCEPT
  ARGUMENTS

You supplied arguments to a pseudo-macro that does not accept arguments.

SEARCHLIST TOO LONG

You specified a search list with more than 511 characters.

SWITCH ABBREVIATION NOT UNIQUE

You abbreviated a switch with a nonunique abbreviation.

SWITCH DOES NOT ACCEPT A VALUE

You supplied a value to a switch that does not accept values.

SWITCH REQUIRES A VALUE

You did not supply a value to a switch that requires a value.

SWITCH FORMAT ERROR

You improperly formatted a switch.

SWITCH UNKNOWN

You entered an invalid switch.

TERMINATED BY ERROR

This process was terminated by an error.

TOO MANY CHARACTERS IN STRING

STRING argument string exceeds 127 characters.

UNABLE TO CREATE BATCH INPUT FILE

The CLI can't create the batch job file because maximum size of the control point would be exceeded.

UNMATCHED [(or <

You have not closed an open [, (, or <.

UNMATCHED]) or >

You have not opened a closed],), or >.

UTILITY TASK STACK OVERFLOW

This is a system error. See your system manager.

WRONG NUMBER OF ARGUMENTS

You supplied the wrong number of arguments to a command or pseudo-macro.

Editors

SPEED Commands

A

Append a page or window from input file to current edit buffer.

:A

Same as A, except command returns a value to the next command depending on the success or failure of the Append. The value is positive (+) if the Append was successful and 0 if the Append failed.

BCx

Copy entire current buffer to buffer x.

[-] nBCx

Copy next n lines from CP (current position) in current buffer to buffer x. If you include the minus sign, copy the preceding n lines up to CP to buffer x.

m,nBCx

Copy characters (m+1) through n from current buffer to buffer x.

BFR, BFW, BFC, BFNR, BFNW, BFO, BFU, BFB

Execute an FR, FW, FC, FNR, FNW, FO, FU, or FB command local to the current buffer.

BKx

Deactivate buffer x.

BSx

Make buffer x the current edit buffer.

BTx, nBTx, -nBTx, m,nBTx

Same as BCx, nBCx, -nBCx, and m,nBCx respectively, but delete all characters transferred from current buffer.

B? *[x]*

Type buffer status for all current buffers. If you include the optional argument, type buffer status of buffer *x*.

↑Bx

Insert contents of buffer x into command string in place of ↑Bx command.

Ctext$_1$ $text$_2$

Search entire buffer for text$_1$ and replace with text$_2$.

[-] nCtext$_1$ $text$_2$

Search through the next n carriage returns for text$_1$ and replace with text$_2$. If you include the minus sign, search the preceding n lines up to the CP for text$_1$ and replace with text$_2$.

0Ctext$_1$ $text$_2$

Search the current line from beginning to CP for text$_1$ and replace with text$_2$.

m,nCtext$_1$ $text$_2$

Search characters (m+1) through n for text$_1$ and replace with text$_2$.

nD

Delete n characters starting at CP.

E

Copy current buffer and rest of input file to output file.

↑E

Match any number of spaces or tabs in a Search command.

FB

Copy current buffer and rest of input file to output file and close all input and output files. If files were opened by FO command, create backup file with .BU extension. Clear buffer.

FC

Close all current global input and output files. FC does not write the text in the edit buffer to the output file before closing it.

FNR *[pathname]*

Close current input file. If you include the *pathname*, close current input file and open another input file with the specified name. You can't use either form of the FNR command if you opened the file with the FO command.

FNW *[pathname]*

Close current output file. If you include *pathname*, close current output file and open a new output file with the specified name. SPEED transfers no data from the edit buffer or input file.

FOpathname

Open pathname for input, pathname.TM for output, and Yank (Y) a page into buffer. Set update mode ON for this file.

FRpathname

Open pathname for input. You can't use FR if update mode is ON.

FU

Transfer current buffer and remainder of input file to output file. Close input/output files. If files were opened with FO command, delete input file and rename output file to input filename.

FWpathname

Create and open a new output file, pathname.

F?

Type status of global and local files.

↑Fpathname

Insert contents of pathname into command string in place of ↑F command.

↑G

(Model 6052 and 6053 CRTs only.) Position CP at location of ↑G in Search string.

H

Normal exit from SPEED. Return to SPEED's parent process, usually the CLI.

↑Itext

Insert text into buffer at position of CP.

nI

Insert ASCII decimal equivalent of n at position of CP.

↑Itext

Insert tab plus text into buffer at position of CP.

n\

Insert ASCII representation of decimal n at position of CP.

J,0J,1J

Move CP to beginning of buffer.

nJ

Position CP at the beginning of line n.

[-] nK

Delete characters from CP through next n carriage returns. If you include the minus sign, delete the preceding n lines up to the CP.

m,nK

Delete the characters (m+1) through n in the current buffer.

K,0K

Delete characters from beginning of line up to CP.

#K

Delete entire buffer.

L,0L

Move CP to beginning of current line.

[-] nL

Move CP to beginning of line following the nth NEW LINE. If you include the minus sign, move CP to beginning of nth line preceding current line.

nM

Move CP across n characters. If n is positive, the CP moves to the right; if n is negative, CP moves to the left.

Ntext

Search the current buffer and the rest of the input file for text. Copy buffer to output file if text is not found.

↑Nx

Match any character in this position of Search string except x.

Ostring

Transfer control to label string.

P

Copy edit buffer to output file with appended form feed.

[-] nP

From CP, copy n lines to output file with appended form feed. If you include the minus sign, copy the preceding n lines plus characters on current line up to CP to output file with appended form feed.

0P

Copy current line from beginning through CP to output file with appended form feed.

m,nP

Copy characters (m+1) through n in buffer to output file with appended form feed.

PW

Copy edit buffer to output file without form feed.

[-] nPW

From CP, copy n lines to output file without form feed. If you include the minus sign, copy the preceding n lines plus characters up to CP on the current line to the output file with no form feed.

0PW

Copy current line from beginning through CP to output file without form feed.

m,nPW

Copy characters (m+1) through n in buffer to output file without form feed.

SPEED Commands (continued)

Qtext

Same as Ntext except SPEED does not copy the buffer to the output file if text is not found.

R

Copy current buffer to output file, clear buffer, and Yank next page from input file.

nR

Repeat R command n times.

Stext

Starting at CP, search for text in current buffer.

[-] nStext

Search for text from CP through next n carriage returns. If you include the minus sign, search preceding n lines plus characters up to CP on current line.

OStext

Search for text in current line from beginning of line up to the CP.

m,nStext

Search characters (m+1) through n in buffer for text.

T

Type current line, indicating current location of the CP, (↑).

[-] nT

Type n lines of the current buffer. If you include the minus sign, type n lines preceding and including current line up to CP.

0T

Type current line from beginning to CP.

m,nT

Type characters (m+1) through n in the current buffer.

#T

Type entire buffer.

Vv

Represents current value of variable v.

VC

Represents decimal value of ASCII character which follows CP.

VC=

Display decimal value of ASCII character which follows CP.

VDv

Decrement value of variable v; represents the decremented value.

VIv

Increment value of variable v; represents the incremented value.

SPEED Commands (continued)

VL

Represents number of the current line.

VL=

Display number of the current line.

VN

Represents number of lines in current buffer.

VN=

Display number of lines in the current buffer.

nVSv

Set variable v to value n and return that value.

WC=

Display current value of case control mode.

0	Case control off
1	up shifting
-1	Down shifting

WC

Return value representing case control mode (0, 1, or -1).

0WC

Turn case control off.

nWCx

If n is positive, shift up any character preceded by x. If n is negative, shift down any characters preceded by x.

nWCxy

If n is positive, shift up using x as shift character and y as shift-lock character. If n is negative, shift down using x as shift character and y as shift-lock character.

WM=

Display current state of data input mode (i.e., page or window; 0 means page mode, n is window mode).

WM

Represents value of data input mode.

nWM

Change from page mode to a window mode of n lines, or change lines/window.

0WM

Change from window to page mode.

WS

Represents current status of case mode.

WS=

Display current status of case mode.

0WS

Let Search command(s) match text, regardless of whether upper- or lowercase.

nWS

Case of text to be found must match text in Search command; n is a decimal number from 0 to 65535.

Xclicommand

Execute a CLI command without exiting from SPEED.

Y

Clear buffer and read in one page from input file.

Z

Represents total number of characters in current buffer.

Z=

Display total number of characters in current buffer.

↑Z

Accept any character in this position (use with Search commands S,C,N, and Q).

.(period)

Represents the current CP position.

.=

Display current CP position.

:(colon)

1. Modify Search and file input commands (A,Y,R,S,C,N, and Q) to return a value of 1 if the command succeeds, and 0 if it fails.

2. Modify output commands (P,nP,m,nP,PW, nPW,m,nPW) to delete all characters in the buffer after output.

3. Modify execute command (X) to execute a program from SPEED.

n <command string>

Perform enclosed command n times. If n < 0, skip command loop.

;

Jump out of command loop if last Search command failed.

:;

Jump out of command loop if last Search command was successful.

n;

Jump out of command loop if n < 0.

n:;

Jump out of command loop if n > 0.

n\

Insert ASCII representation of the decimal number n into the buffer at the CP location.

n=

Type out value of numeric variable n.

←x

Place previous command string in buffer x. (This must be the first command after prompt.)

@

Modify Insert and Search commands to change text delimiter. The first symbol following the command is @ which defines the delimiter for this command.

#

Equivalent to double argument 0,Z (entire edit buffer).

↑SHIFT-N (↑↑) or ↑SHIFT-G (↑↑)

Position CP at location of SHIFT-N (ASCII 036_8) in search string if search is successful. (Graphic varies according to keyboard, see ↑G.)

↑SHIFT-O (↑_)

Interpret next character literally, not as a special character. (CTRL-SHIFT 0 is ASCII 037_8. Graphic varies according to keyboard.)

n"Gcommand string'

Execute command string if $n > 0$.

n"Lcommand string'

Execute command string if $n < 0$.

n"Ecommand string'

Execute command string if $n = 0$.

n"Ncommand string'

Execute command string if $n \neq 0$.

!string!

Define a label named string in the command string.

DEBUG/DEDIT