AOS and AOS/VS User's Handbook

093-000150-02

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 02, April 1982

NOTICE

AOS and AOS/VS
User's Handbook
093-000150

Revision History:	Effective with
Original Release - July 1978	
First Revision - June 1979	
Second Revision - April 1982	AOS Rev. 4.0 AOS/VS Rev. 1.2

Preface

The *AOS and AOS/VS User's Handbook* is a capsulized version of the *AOS and AOS/VS User's Manual*. To make this handbook even handier, in addition to the CLI commands that comprise the major portion of this book, we've included a summary of SPEED, SED and SWAT commands.

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 terminal reference book, you should first familiarize yourself with the appropriate manuals describing the AOS and AOS/VS CLI, compilers, utilities, and subsystems operating under AOS and AOS/VS. Once you have done so, keep this book near your terminal. It should prove to be an invaluable aid in a minimal format.

Referrals

In writing this handbook, we have compiled information from many manuals. If you have specific questions about any topic, you'll find detailed information in the appropriate book. If you want to know about other AOS and AOS/VS manuals, consult the *Software Documentation Summary Card* 069-000013-04 for their order numbers, and the Product Summary Series publication *Data General Documentation*, 014-000637, for descriptons of their contents.

AOS

Introduction to AOS (069-000016)
Learning to Use Your AOS System (069-000018)
AOS Software Documentation Guide (069-000020)
Command Line Interpreter User's Manual (AOS and AOS/VS) (093-000122)
AOS/VS SED Text Editor's Manual (093-000249)
SPEED Text Editor's Manual (AOS and AOS/VS) (093-000197)
AOS Programmer's Manual (093-000120) 1982
AOS Macroassembler Reference Manual (093-000254)
AOS Link User's Manual (093-000254)
AOS Debugger and File Editor User's Manual (093-00195)
AOS Library File Editor User's Manual (093-000198)
Running AOS On Your ECLIPSE MV/8000 Computer (093-00259)
AOS System Manager's Guide (093-000193)
AOS Operator's Guide (093-000194)
How to Load and Generate Your Advanced Operating System (093-000217)

AOS/VS

AOS/VS SED Text Editor's User's Manual (093-00249)
SPEED Text Editor User's Manual (AOS and AOS/VS) (093-000197)
AOS/VS Programmer's Manual (093-000241)
AOS/VS Macroassembler Reference Manual (093-000242)
AOS/VS Link and File Editor User's Manual (093-000245)
AOS/VS Debugger User's Manual (093-000246)
Managing AOS/VS (093-000243)
AOS/VS Operator's Guide ((093-000244)
BASIC User's Manual (AOS/VS) (093-000252)

Names of manuals describing other Data General Corporation products follow.

APL Reference Manual (093-000274)
Using Your AZ-TEXT™ Word Processing System (093-000220)

Managing Your AZ-TEXT™ Word Processing System (093-000271)
Basic User's Manual (AOS/VS) (093-000252)
basic BASIC (069-000003)
Guide to Using Business Basic (AOS/RDOS/DOS) (069-000028)
Business BASIC Directory (093-000226)
Business BASIC System Management (093-000228)
Extended BASIC User's Manual (093-000065)
COBOL Reference Manual (AOS) (093-000223)
COBOL Reference Manual (AOS/VS) (093-000289)
DATAPREP® Data Entry Subsystem Supervisor's Guide (093-000281)
DATAPREP® Operator's Guide (093-000163)
Guide to Using Your DG/DBMS System (069-000025)
DG/DBMS Reference Manual (093-000163)
PRESENT User's Manual (093-000168)
DG/L™ Reference Manual (093-000229)
DG/L™ Runtime Library User's Manual (AOS, AOS/VS) (093-000159)
DG/SNA Programmer's Manual (093-000282)
SNA/3270 Operator's Guide (AOS and AOS/VS) (093-000287)
DG/SNA Operator's Guide (093-000283)
DG/3278 User Terminal Guide (093-000284)
Data General's FORTRANs: A Technical Comparison (069-000029)
FORTRAN IV User's Manual (093-000053)
FORTRAN IV Runtime Library User's Manual (NOVA®) (093-000068)
FORTRAN IV Runtime Library User's Manual (ECLIPSE®) (093-000142)
MP/FORTRAN IV Programmer's Reference (093-4000040)
FORTRAN QCALLS Reference Manual (AOS) (093-000239)
FORTRAN 5 Reference Manual (093-000085)
FORTRAN 5 Programmer's Guide (AOS, AOS/VS) (093-000154)
FORTRAN 77 Reference Manual (093-000162)
FORTRAN Commercial Subroutine Package Reference Manual (093-000107)

HASP II Workstation Emulator User's Manual (AOS, AOS/VS) (093-000158)
Idea Concepts and Facilities (AOS, AOS/VS) (069-000023)
Idea Programmer's Reference (AOS, AOS/VS) (093-000151)
The INFOS® System Storybook (069-000019)
INFOS® System User's Manual (AOS, AOS/VS) (093-000152)
INFOS® II Query/Report Writer User's Manual (093-000214)
MP/PASCAL Programmer's Reference (093-400003)
Plain PL/I (A PL/I Primer) (093-000216)
PL/I Reference Manual (AOS, AOS/VS) (093-000204)
PL/I Reference Manual (AOS/VS) (093-000270)
PROXI™ User's Manual (055-000038)
RCX70 Terminal Operator's Guide (093-000170)
RCX70 Reference Manual (093-000172)
Remote Job Emulator (RJE) User's Manual (AOS, AOS/VS) (093-000157)
RPG II Reference Manual (093-000117)
RPG II Optimizing Compiler User's Manual (AOS, AOS/VS) (093-000279)
Sort/Merge Utility User's Manual (AOS, AOS/VS) (093-00155)
Sort/Merge Utility User's Handbook (093-000176)
SWAT™ User's Manual (093-00258)
TPMS Reference Manual (093-000205)
TPMS Operator's and System Manager's Guide (AOS) (093-000206)
TRENDVIEW™ Graphics Charting Package User's Manual (093-400014)
Learning to Microprogram Your ECLIPSE® Computer (014-000098)
Microprogram Utilities User's Manual (093-000179)
XODIAC™ Network Management System User's Manual (093-000178)
XODIAC™ Guide for Operators and Network Managers (093-000260)

Reader, Please Note:

We have used the terms console and terminal interchangeably in this manual.

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.
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 this 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 carriage return (CR) 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, in examples we use

THIS TYPEFACE TO SHOW YOUR ENTRY)
THIS TYPEFACE FOR SYSTEM QUERIES AND RESPONSES.

) is the CLI prompt.

Contacting Data General

If you:

- Have comments on this manual -- Please use the prepaid Remarks Form that appears after the Index.
- Require additional manuals -- Please contact your local Data General sales representative.
- Experience software problems -- Please notify systems engineering.

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,FORT3.LB,&↵
&)IMPYD.LB↵
```

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 either the AOS or AOS/VS Command Line Interpreter user's manual.

Control Characters

AOS and AOS/VS provide control characters which give you additional control over your terminal. To issue a control character instruction, simultaneously depress the key labeled CTRL and one or more of the control characters listed in the following table:

Control Character(s)	Operation Performed
CTRL-O	Cancel the display of whatever information is appearing on your terminal. You can restart the display by typing CTRL-Q.
CTRL-S	Postpone the display of information on your terminal until you decide to restart it by typing CTRL-Q.
CTRL-Q	Restart the display of information on your terminal.
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 either the AOS or the AOS/VS Command Line Interpreter user's manual.

End of Preface

Contents

AOS and AOS/VS CLI Commands

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

Command Switches

/1 = { IGNORE | WARNING | ERROR | ABORT }	Set CLASS1 to the specified severity level for this command.
/2 = { IGNORE | WARNING | ERROR | ABORT }	Set CLASS2 to the specified severity level for this command.
/E	Write error messages to file specified by pathname instead of @OUTPUT.
/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.

ACL *Command*

Format

ACL pathname *[user access]...*

Purpose

Set or display the access control list for a file.

To display a file's ACL, supply pathname as the only argument to the ACL command. To set or change a file's ACL, specify user and access as well as pathname. You may use templates in the pathname argument and in the user argument. The arguments must be separated by a CLI separator: one or more blanks, one or more tabs, one comma, or any combination of these (e.g., one comma and one or more blanks).

The CLI displays the access control list (ACL) in the following format:

username-template access-types username-template access-types ...

where access-types is a string of one or more of the following characters:

Character	Meaning
O	Owner access.
W	Write access.
A	Append access.
R	Read access.
E	Execute access.

Command Switches

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

/V
Display the filename with the ACL.

/K
Delete ACL; this denies everyone except Superusers access to the file until the ACL changes again (takes pathname argument only).

/D
Give the file the user's default ACL (takes pathname argument only).

Argument Switches

None.

Examples

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

The first ACL command displays the access control list for file TEST.PR. The second ACL command sets a new access control list for that file, and the third command displays the new access control list preceded by the filename.

!ACL *Utility*

Format

[!ACL pathname]

This pseudo-macro requires one pathname argument.

Purpose

Expand to a file's access control list.

Macroname Switches

None.

Argument Switches

None.

Example

```
) WRITE FILE'S ACL IS [!ACL FILE]
FILE'S ACL IS COLLATE_DEBTS,OWARE
)
```

The CLI first evaluates the pseudo-macro [!ACL FILE], then writes the resulting argument list on the terminal.

AOSGEN is the Advanced Operating System generation program.

Purpose

Generate a new operating system (AOS only). Refer to the AOS manual: *How to Load and Generate Your Advanced Operating System.*

Format

XEQ APL *[initial-workspace-pathname]*

Purpose

Invoke the APL interpreter (AOS/VS only.)

You may specify an initial workspace. If you do not, the interpreter looks by default for a workspace named CONTINUE. If this file exists, it is automatically loaded. If it does not exist, execution begins in a clear workspace.

Some APL switches require a value that identifies your terminal or input device to the system. The following codes are acceptable for the switches that take a terminal-type value:

Term-type	Meaning
0	Batch (This is the default for disk files).
1	605x or D200 compatible terminal without the APL character set.
2	6110 APL display terminal (This is the default for CRTs).
3	TP2 model 6193 with the APL character set downline loaded (This is the default for hardcopy devices).
4	APL/ASCII typewriter pairing terminal.
5	APL/ASCII bit pairing terminal.

For terminal types 4 and 5, APL sends the ASCII shift-out character (ASCII SO) to your terminal. This changes your character set to the APL character set. It then sends the ASCII shift-in character (ASCII SI), which changes the character set

back to ASCII. If your terminal does not respond to the shift characters, or if you use a keyboard switch to change character sets, you should also use the /INS, /ONS, or /LNS switches to suppress the shift characters.

APL Switches

/L=pathname
See CLI Commands page.

/ESC
Do not interpret the ESCAPE key as CTRL-C CTRL-A.

/I=pathname
Specify pathname as the input file. If you do not use this switch, the default input file is @INPUT.

/INS
Do not write ASCII shift characters to the input file.

/ITT=term-type
Specify the input file's terminal type. The possible values for term-type are listed above.

/LNS
Do not write ASCII shift characters to the log file.

/LTT=term-type
Specify the log file's terminal type. The possible values for term-type are listed above.

/MINUS
Print APL's overbar as - on all output.

/O=pathname
Specify pathname as the output file. If you do not use this switch, the default output file is @OUTPUT.

/ONS
Do not write ASCII shift characters to the output file.

APL (continued)

/OTT=term-type
Specify the output terminal's type. The possible values for term-type are listed above.

/PW=integer
Set PW (the page width) to integer number of characters when a clear workspace is activated.

/SLX
Suppress execution of LX (the latent expression).

/TAKE
Use the uparrow (↑) as an error indicator.

/WSLIMIT=integer
Specify the maximum amount of space, in bytes, that you wish to use.

Example

```
) XEQ APL/L=LOGGING.FILE PROG2)
```

Invoke the APL interpreter, loading PROG2.WS as the initial workspace. In addition, use LOGGING.FILE as the log file.

Format

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

Each octal number must be a positive integer in the range 1 to 377.

Purpose

Expand to characters corresponding to octal arguments.

Macroname Switches

None.

Argument Switches

None.

Example

You can use !ASCII to enter special characters that wouldn't normally be interpreted correctly by the CLI. For instance, if you want to use the WRITE command to ring your terminal's bell, you cannot type the bell character (CTRL-G) into a WRITE command. If you tried to do this, the CLI would merely echo a ↑G (CTRL-G).

The example shows you how to use !ASCII to include a bell character with the parity bit set.

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

ASSIGN *Command*

Format

ASSIGN character-device *[character-device]...*

Purpose

Assign a character device for your exclusive use.

Character devices include card reader, printer, terminals, etc.

After you assign a device, you control it until you either deassign it or log off the system. You cannot assign a spooled device under the EXEC. You can use templates in the character-device argument.

Command Switches

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

Argument switches

None.

Example

```
) ASSIGN @CRA)
)
      .
      .          (you have exclusive use of the device)
      .
) DEASSIGN @CRA)
)
```

Format

For AOS:

XEQ BASIC

For AOS/VS:

XEQ BASIC *[program-pathname]*

Purpose

Invoke the BASIC interpreter.

BASIC is a programming language interpreter. Several versions are available, each designed to run under a different operating system. Use it to create and execute BASIC programs.

Under AOS/VS: you may provide a program pathname as an argument. The program must be a program file (type PRV), a BASIC core-image file (type BCI), or a BASIC source file. BASIC can use this pathname as an argument to its CHAIN command, which stops execution of the current program and loads and runs the specified program.

For more information on BASIC and the BASIC utility, see the documentation for the version of BASIC that runs under your operating system:

basic BASIC (AOS only)
Extended BASIC User's Manual (AOS only)
BASIC User's Manual (AOS/VS)

BASIC (continued)

BASIC Switches

/NOSIGN
(AOS/VS only) Do not output sign-on and sign-off messages.

Argument Switches

None.

Example

```
) XEQ BASIC↵
*
      .          (Enter BASIC commands)
      .
* BYE↵
)
```

Format

BIAS *[minimum number [maximum number]]*

Purpose

Set or display the system's bias factor.

Any process can display the system's bias factor but only PID 2 can set it. The default minimum is zero and the default maximum is 'no limit'. If you set only the minimum number, the maximum will automatically be set to 'no limit'. See the *AOS System Manager's Guide* or *Managing AOS/VS* for a description of the system's bias factor.

Command Switches

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

Argument Switches

None.

BIAS (continued)

Examples

```
) BIAS↵
MINIMUM: 0, MAXIMUM: NONE
)
```

Display the system's bias factor.

```
) BIAS 0↵
)
```

Set the system's bias factor to MINIMUM: 0, MAXIMUM: NONE.

```
) BIAS 0 21↵
)
```

Sets the system's bias factor to MINIMUM: 0, MAXIMUM: 21.

Format

XEQ BIND objectmodule *[argument]...*

Purpose

Bind object modules to form an executable program file (AOS only).

Use the Binder to build an executable program file from object files.

objectmodule is the name of the first object to be bound. Unless you specify otherwise with the /P switch, the program file will be named objectmodule.PR. *[argument]...* can be any of the following:

- Another object module
- A shared or unshared library name
- A symbol name or integer, with the appropriate switch
- A command file, with the /C switch

The command file specifies objects you want to bind as overlays, along with their switches. These will be placed in overlay file objectmodule.OL, to correspond with program file objectmodule.PR. To bind overlays, you must build (CREATE) the command file from the binaries which you want to be overlays. See the *AOS Binder User's Manual* for more information.

BIND Switches

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

BIND (continued)

/B
Produce symbol file listing, ordered alphabetically and numerically.

/E
Output load map to @OUTPUT, even though listing is specified.

/H
Print all numbers in hexadecimal.

/I
Build nonexecutable program file without user status table, task control blocks, or other system tables. Do not scan user runtime library (URT.LB). Switch can help check for BIND errors, such as multiply defined or undefined .ENT symbols.

/K=integer
Set number of tasks to integer. This number overrides any .TSK pseudo-op statement included in source file.

/M=integer
Save integer number of 1K-word pages of memory for shared library use.

/N
Do not scan user runtime library (URT.LB).

/O
Allow load overwrites to occur.

/P=pathname
Name the program file pathname.PR. Default is the name of the first object module.

/S
Produce shared routine. Include switch to build shared library.

/T=integer
Set decimal integer as top of shared area. BIND rounds area to an even 1K boundary.

/Z=integer
Set decimal integer as stack size for program. By default, BIND allocates 30 words (decimal).

Argument Switches

/AM=integer
Set total overlay area to integer number of basic areas. Applies only to right bracket in an overlay specification.

/B
Bind shared library into shared area. (This applies only to shared library.)

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

/D
Bind nonshared code in module into nonshared data area.

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

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

/U
Write local symbols from module to symbol file. If macroassembler produced this .OB file, do not use switch unless you also specify /U to the macroassembler.

BIND (continued)

symbol/V=integer
Assign value integer to accumulating symbol, defined by pseudo-op .ASYM

/X
(Used in conjunction with the /B switch.) Exclude shared library routine in library included by the /B switch.

integer/Z
Set the ZREL base to octal integer. If current ZREL base exceeds value, system ignores switch.

Examples

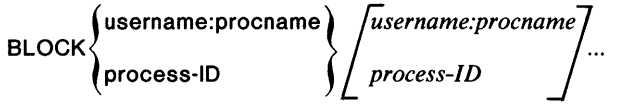
```
) XEQ BIND/L=LFILE MYPROG MYLIB 100/Z↵
```

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

```
) CREATE/I COMMANDFILE↵
)) [OVLY1,OVLY2 OVLY3,OVLY4]↵
))) ↵
)
```

Create a command file containing the names of four object files. Use this command file to create three overlays. Since no comma separates OVLY2 from OVLY3, the system will bind them into one overlay. Because the overlays are enclosed in one set of brackets, the system will reserve one overlay area in memory for them. Each will occupy this area as the program calls it. Each overlay should specify the same kind of relocation: shared or unshared.

Format



Purpose

Block a process.

Supply a process ID or process name that must be an inferior process (unless you have the SUPERPROCESS privilege to block any process). **procname** must be a full process name.

Command Switches

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

Argument Switches.

None.

BLOCK (continued)

Examples

```
)BLOCK 19↓
)
```

Block the process with PID (process) ID 19.

```
)BLOCK SMITH: PROG1↓
)
```

Block the process names PROG1.

Format

XEQ BRAN break-file-pathname *[symbol-table-pathname]*

Purpose

Analyze an AOS/VS break file (AOS/VS only).

The BRAN utility produces a report from an AOS/VS break file, giving global information about the process and information for each active task. Process information includes the program type, memory usage, the current task (active when the process terminated), and the number of free tasks and active tasks. For each active task, the BRAN report describes the task identifier (task ID), the task's priority, and the values of the program counter (pc), the accumulators, and the stack pointers. In addition, the report lists the system call being serviced at the time of the termination.

If you cite the symbol table pathname as an argument, BRAN prints certain values (such as the pc) symbolically.

Command Switches

/L=pathname
See CLI Commands page.

BRAN (continued)

Example

```
) XEQ BRAN/L=REPORT ?010.023_026_016.BRK&J
&)MYPROG.STJ
)
```

Analyze the break file ?010.023_026_016.BRK from program MYPROG.PR, and write the break file report to file REPORT. Since the command line cites the symbol table MYPROG.ST, list the pc symbolically.

Format

BYE *[argument]...*

Purpose

Terminate CLI process.

The CLI returns any arguments as a string to the father process.

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 respond NO, the CLI does not terminate.

Command Switches

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

/WARNING
/ERROR
/ABORT
If you use any of these switches, the CLI will terminate, signalling the specified severity level. If you include any arguments to the command, the arguments will return to the calling process as a string.

Argument Switches

None.

BYE (continued)

Example

) BYE↵
AOS CLI TERMINATING 26-NOV-80 13:00:19

Terminate the CLI. If the CLI's father was EXEC, you are logged off the system. (Note that, under AOS/VS, the termination message reads *AOS/VS CLI TERMINATING.*

Format

CBIND objectmodule *[argument]...*

Purpose

Bind object modules to form an executable COBOL program (AOS only).

CBIND is a macro that invokes the AOS Binder utility to make COBOL object modules into an executable program.

objectmodule specifies the main program. If you do not provide a filename extension, CBIND assumes that the complete filename is objectmodule.OB. You may also include other arguments on the command line. An argument might specify a subprogram, shared library, accumulating symbol, or some other part of the loaded program. ICALL is the COBOL interface to the INFOS system (supplied by INFOS), which you need if you use INFOS indexed files. Either use LFE to add ICALL to URT.LB, or include ICALL on the CBIND command line.

For a complete description of the COBOL programming language and the CLI CBIND command line, see the *COBOL Reference Manual (AOS).*

CBIND Switches

/L,/L=pathname,/Q
See CLI Commands page.

/B
List symbol table in alphabetical and numeric order.

/D
Bind in COBOL debugger program. Load COBOL program modules as unshared code.

CBIND (continued)

/E
Output load map to @OUTPUT, even though another listing file is specified.

/H
List all numbers in hexadecimal.

/I
Build nonexecutable program file, lacking a UST, TCBs, and all other system databases.

/K=integer
Allocate integer number of TCBs for multitask use, regardless of the number specified in a .TSK statement.

/N
Do not scan user runtime library, URT.LB

/O
Suppress error flags when bind overwrites occur.

/T=integer
Set highest address in shared partition. If integer is not a multiple of 2048 bytes, binder rounds it down to next lower 2048-byte multiple.

/Z=integer
Set stack size for default task.

Argument Switches

/B
Bind externally referenced routines from shared library into the root context.

/C
Use module as command file (required to define overlays using square brackets).

/D
Load nonshared code in module as unshared data.

/H
Load unshared code in module as shared code.

/O
Allow overwrites in module. See the /O CBIND switch.

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

/S
Convert shared code modules to unshared code modules.

/U
Load local symbols from module into the symbol file. /U works only if applied to this module in an earlier macroassembler command.

symbol/V=integer
Create accumulating symbol and initialize it to integer.

integer/Z
Set current ZREL base to number specified by argument.

Example

```
) CBIND/L=MYFILE.MP MYFILE UPDATSUB&)
&)HACKSUB)
```

Bind MYFILE, the main program, and two subprograms, UPDATSUB and HACKSUB. The binder output listing goes to MYFILE.MP.

CHAIN *Command*

Format

CHAIN pathname *[argument-to-new-program]...*

Purpose

Overwrite the CLI with the program named in pathname and transfer CPU control to that program's entry point. Arguments are placed in the initial inter-process communication (IPC) message to the new process. The new process can access the arguments through the ?GTMES system call. On an AOS/VS system, a 32-bit program cannot chain to a 16-bit program, and a 16-bit program cannot chain to a 32-bit program.

The CLI first tries to chain to pathname.PR. If that fails, the CLI chains to pathname.

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

Command Switches

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

/D
Enter the Debugger.

Argument Switches

Use any argument switches appropriate for the program specified in pathname.

Examples

) CHAIN MYPROG↵

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

) CHAIN MASM/L TESTA↵

Load MASM into memory and begin execution at its entry point. /L produces a listing file.

CHARACTERISTICS *Command*

Format

CHARACTERISTICS *[device]...*

Purpose

Set or display device characteristics.

If you do not supply an argument, your terminal becomes the default device. Set or display the device characteristics for a character device. Device characteristics control the way the device interprets input or sends output. The characteristics you set will be in effect until you change them or log off the system. You can issue successive CHARACTERISTICS commands.

Command Switches

/1,/2,L,L=pathname,/Q
See CLI commands page.

/LPP=n
Lines per page, in decimal.

/CPL
Characteristics per line, in decimal.

/DEFAULT
Used alone, this switch displays the default characteristics of a device. Used with other switches, it sets the default characteristics of a device. PID 2 is the only process authorized to set default characteristics.

/RESET
Set the characteristics of a device to its default characteristics. This switch must be used alone.

/PREVIOUS
Set the current characteristics of a device to the previous environment's characteristics (no arguments or other switches allowed).

/ON
Set the following characteristics ON until the /OFF switch or a delimiter occurs. This bit is automatically set unless you you include the /OFF switch. (This switch is optional).

/OFF
Clear the bit in the device characteristics words for each of the command switches that follow, until the /ON switch or a delimiter occurs.

/8BT
All 8 bits of an ASCII character are interpreted as data.

The following octal codes will echo an uparrow followed by an alphabetic character regardless of whether this switch is set or not:

1 to 10
13
16 to 32
34 to 37

/EB0
For echoing to occur on your terminal, you must set /EB0 or /EB1. /EB0 echos control characters such as ↑A, ↑B, etc. It echos ESC as $. For more information see ?GCHR in the AOS or AOS/VS programmer's manual.

/EB1
Echo characters exactly as they are input. For more information, see ?GCHR in the AOS or AOS/VS programmer's manual.

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

CHARACTERISTICS (continued)

/EOL
Do not output a new line if CPL line length is exceeded on output.

/ESC
ESC character produces ↑C↑A interrupt.

/FF
Output a form feed on open.

/FKT
Permit function keys to serve as delimiters in data-sensitive read operations.

WARNING: Do not use function keys to end CLI commands.

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

/MOD
Device is on a modem interface.

/MR
(AOS/VS only) Monitor Ring Indicator.

/NAS
If this switch is on, set non-ANSI standard bit. The device is considered non-ANSI standard. On input, this switch converts carriage return to NEW LINE and line feed to carriage return. On output, it converts line feed to carriage return-line feed.

/NNL
Do not automatically append NEW LINES to card images.

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

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

/PBN
Packed format on binary read, 4 columns are put in 3 words; if this switch is off, columns are right-justified in memory (card readers only).

/PM
Page mode: if this switch is on, write LPP lines per page on output, then suspend output until the user types CTRL-Q.

/RAC
If this switch is on, send 2 rubouts after each NEW LINE and carriage return.

/RAF
If this switch is on, send 21 (decimal) rubouts after each form feed.

/RAT
If this switch is on, send 2 rubouts after each tab (CTRL-I).

/SFF
If this switch is on, simulate form feed.

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

/ST
Simulate a tab stop every 8th column.

/TO
Enable time-outs.

CHARACTERISTICS (continued)

/TSP
Include trailing spaces; if this switch is off, suppress trailing spaces (card readers only).

/UCO
Output and convert lowercase to uppercase.

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

/WRP
Hardware generates new line on line-too-long.

You can identify your terminal with any of the following switches (the system also displays these to identify your terminal):

/HARDCOPY	Hard-copy terminals.
/4010I	DGC Model 4010I.
/6012	DGC Model 601.
/605x	DGC Model 6052 or 6053.
/6130	DGC Model 6130.
/CRT4	Other video display terminals.

Argument Switches

None.

Examples

```
) CHARACTERISTICS↵
/HARDCOPY/LPP=24/CPL=80
/ON/ST/SPO/EB0/ULC/WRP
/OFF/SFF/EPI/8BT/RAF/RAT
/RAC/NAS/OTT/EOL/UCO/LT/FF
/EB1/PM/NRM/MOD/TO/TSP
/PBN/ESC/FKT/NNL
)
```

Display the characteristics of the terminal; in this case, a hard-copy terminal.

```
) CHARACTERISTICS/LPP=24↵
)
```

Set the number of lines per page to 24 for your terminal.

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

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

```
) CHARACTERISTICS/CPL=132 @LPA↵
)
```

Set the characters per line for the line printer to 132 decimal. To set characteristics, use the device name -- in this case @LPA -- rather than a queue name (e.g., @LPT).

CHECKTERMS *Command*

Format

CHECKTERMS

Purpose

Check for the termination of a son process.

Displays the process termination message from any son processes. If a process has terminated abnormally (e.g., terminal interrupt trap, etc.), the CLI outputs an appropriate message. See either your AOS or AOS/VS programmer's manual for a discussion of ?RETURN.

Command Switches

/1,/2,/L,/L=filename,/Q

Argument Switches

None.

Example

```
) 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 PID 14's termination message.

CLASS1 *Command*

Format

CLASS1 *[severity level]*

Purpose

Set or display CLASS1 setting.

The following are severity levels:

IGNORE	The CLI displays no message; it continues processing your input as best it can.
WARNING	The CLI displays a warning message and continues processing your input as best it can.
ERROR	The CLI displays an error message and discards the input that is in the command buffer at the time it encounters the mistake. The command buffer contains all input from the last prompt to the NEW LINE character. This may be one command, multiple commands, or a macro.
ABORT	Your process terminates at once.

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

Command Switches

/1,/2,/L,/L=filename,/Q
See CLI Commands page.

/P

Set CLASS1 severity level to the previous environment's severity level (no arguments allowed).

Argument Switches

None.

Example

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

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

Format

CLASS2 *[severity level]*

Purpose

Set or display CLASS2 setting. The following are severity levels:

IGNORE	The CLI displays no message; it continues processing your input as best it can.
WARNING	The CLI displays a warning message and continues processing your input as best it can.
ERROR	The CLI displays an error message and discards the input that is in the command buffer at the time it encounters the mistake. In this instance the command buffer contains all input from the last prompt to the NEW LINE character. This may be one command, multiple commands, or a macro.
ABORT	Your process terminates at once.

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

Command Switches

/1,/2,/L,/L=filename,Q
See CLI Commands page.

/P
Set CLASS2 severity level to the previous environment's CLASS2 severity level (no arguments allowed).

Argument Switches

None.

Example

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

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

CLINK (AOS/VS only) *Utility*

Format

CLINK main-objectmodule *[subprogram-objectmodule]...*

Purpose

Link object modules to form an executable COBOL program (AOS/VS only).

CLINK is a macro that invokes the AOS/VS Link utility, to make COBOL object modules into an executable program. For a list of the switches that the macro accepts, see the *AOS/VS Link and Library File Editor (LFE) User's Manual.*

Format

For AOS:

COBOL source-pathname *[listfile/L] [objectfile/R]*

For AOS/VS:

COBOL source-pathname

Purpose

Compile a COBOL source file. COBOL is a macro that you use to compile a COBOL source file.

source-pathname specifies the source program file you want compiled. *listfile* specifies the file or device to which you want the listing file output. This may be the console (@OUTPUT), the line printer (@LIST), or a disk or tape file. *objectfile* specifies the name to be assigned to the object file the compiler produces. By default, the compiler names the object file source-pathname.OB.

For a complete discussion of the COBOL programming language and the CLI COBOL command line, see the *COBOL Reference Manual (AOS)* or the *COBOL Reference Manual (AOS/VS)*.

Two sets of COBOL switches are given below, one for AOS and one for AOS/VS. Only a few switches are common to both sets; argument switches are valid only for AOS.

COBOL Switches (AOS only)

/L, /Q
See CLI Commands page.

COBOL (continued)

/A
Produce an address map of the relative locations of the Procedure Division lines.

/C
Source code is in card format. By default, the compiler assumes the source is in text format.

/D
Compile debug lines and load code for the interactive debugger.

/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 /A.

/M
Produce a map of data and procedure storage in the object file.

/P
Do not generate an object file.

/S
List compilation statistics (e.g., number of lines, speed of compilation).

/V
Compile for virtual code. This switch provides automatic segmentation of procedure division calls.

/W
Suppress warning messages.

/X
Include a cross-reference table in the listing file.

COBOL Switches (AOS/VS only)

/E=pathname,/L,/L=pathname
See CLI Commands page.

/ANSI
Various ANSI standards override the usual AOS/VS COBOL data-manipulation methods.

/CARD or /C
Source code is in card format. By default, the compiler assumes the source is in text format.

/CODE
Print a generated code listing on the list file. This switch overrides /CODEMAP. /L must accompany the /CODE switch.

/CODEMAP
Print a code offset map on the list file. /L must accompany this switch. If both /CODE and /CODEMAP appear in the same command line, the compiler ignores /CODEMAP.

/D
Compile debug lines.

/DEBUG
Output symbol and line information for use by the SWAT™ debugger.

/ERRORCOUNT= integer
Terminate compilation after the specified number of errors. The default value is 100.

COBOL (continued)

/HALT
Suppress generation of object code.

/LINEID
Generate code to keep track of source line numbers at execution time and to print the line number if a fatal error occurs. /LINEID includes the function of /PROCID.

/MAPCASE
Translate all identifiers into uppercase before compilation.

/N
Suppress production of the object file.

/NOCOPIES
Suppress printing of all copy files.

/NOMAP
Suppress printing of the storage map.

/NOWARNINGS
Suppress severity 1 error messages.

/NOX
Perform extended arithmetic operations in nonextended mode.

/O=pathname
Write the object file to pathname.OB.

/OCTAL
Produce octal values for alphanumeric literals.

/PROCID
Save the procedure names at runtime, and print the procedure name if a fatal error occurs.

/STAT
Write compilation statistics to @OUTPUT.

/XREF
Include a cross-reference table in the listing file.

Argument Switches (AOS only)

If an argument specifies the list file or the object file, it must have the appropriate argument switch appended to it. The pathnames may appear in any order.

/L
List the source code at listfile. If you do not use this switch, @LIST is assumed.

/R
Produce the object file at objectfile. If you do not use this switch, the compiler uses the source file's name with an .OB extension.

COBOL (continued)

Examples

For AOS:

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

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

For AOS/VS:

```
) COBOL/L=FILE1.LS/E=FILE1.ER/DEBUG FILE1
```

Compile the source file FILE1 and produce an object file named FILE1.OB (the default name). The listing file FILE1.LS will contain a source listing (/L). Error messages will be sent to FILE1.ER. Symbol and line information are generated for later use by the SWAT debugger (/DEBUG).

Format

CONNECT { username:procname | process-ID }

Purpose

Establish a customer-server connection.

This command directs the system to establish a connection between you and the server process that you specify. After making the connection you should monitor the server process with the CHECKTERMS command. If the server process terminates for any reason, then you must disconnect from the server. You can disconnect by using the CLI command DISCONNECT. (See the appropriate programmer's manual for a complete description of the customer-server relationship).

Command Switches

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

/S
Store the server's process ID in the current string.

Argument Switches

None.

CONNECT (continued)

Examples

```
) CONNECT OP:SRVR↵
SERVER'S PID: 14
)
```

Connect the user's process with server process OP:SRVR which is PID 14.

```
) CONNECT/S 22↵
SERVER'S PID: 22
) STRING↵
22
)
```

Connect the user's process to PID 22 and store the PID number in STRING. The /S switch lets you use the server's PID as an argument.

Format

[!CONSOLE]

Purpose

Expand to the terminal name.

This pseudo-macro does not accept arguments.

Macroname Switches

None.

Argument Switches

None.

Example

```
) WRITE MY CONSOLE NAME IS [!CONSOLE]↓
MY CONSOLE NAME IS CON12
)
```

CONTROL *Command*

Format

CONTROL ipcport argument *[argument]...*

Purpose

Send a control message to a process.

Argument is a message string sent as an Interprocess Communication (IPC) to the process being controlled. The system operator normally uses this command to control the EXEC or a system spooler. You can use **CONTROL** to control user programs if you've written the program to receive IPCs. See your AOS or AOS/VS programmer's manual for more information about IPCs.

Command Switches

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

/I
Messages from @INPUT follow on successive lines. The system sends each line as a separate IPC. The messages end when you type a line containing a single).

/M
This macro file contains the messages. The system sends each line of the macro as a separate IPC. The macro file ends on a line containing a single).

Argument Switches

None.

Examples

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

Direct the spooler to restart output on the line printer.

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

Direct the EXEC process to enable @CON1.

CONVERT *Utility*

Format

XEQ CONVERT pathname

Purpose

Convert an RDOS .RB file to an AOS or AOS/VS .OB file.

RDOS is another Data General operating system. It supports an .RB relocatable binary module, which is not compatible with AOS or AOS/VS.

The CONVERT utility can convert an RDOS.RB relocatable binary file to an AOS or AOS/VS object file. The command line takes one argument, the input pathname (you can omit the .RB extension). CONVERT does not modify the RDOS file; it creates an AOS or AOS/VS object file with the same name but with the .OB extension.

CONVERT Switches

None.

Argument Switches

None.

Example

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

Produce an AOS or AOS/VS 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).

COPY *Command*

Format

COPY **dest-file sourcefile** *[sourcefile]...*

Purpose

Copy one or more files to a destination file.

If the destination file does not already exist, then its specifications depend on the first (or only) sourcefile. If the first **sourcefile** is a disk file, then **dest-file** will have the same specifications as **sourcefile**. If the first **sourcefile** is a peripheral device, then **dest-file**'s default specifications are as follows:

File type	User Data File.
Record type	Unspecified. You must specify the record type when you open the file or defer it until you read or write the file.
Control parameters	None.
Element size	512 bytes (under AOS/VS, you can specify a default element size during system generation).
Maximum index levels	3
Time block	Time of creation, time of access, and time of last modificaton are set to the current time.

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

Command Switches

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

/A
Append new data to the existing data in **dest-file**.

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

/D
Delete **dest-file** (it must exist) and recreate **dest-file** using the same specifications as for the old **dest-file**.

/IDENSITY=mode
Control the magnetic tape density for input files. Use this switch with MTB, model 6026 tape drives only. These are your mode options:

MODE	DENSITY
800	800 BPI
1600	1600 BPI
ADM	Automatic Density Matching

/IMTRSIZE=block-size (bytes)
Control the magnetic tape block size for input files.

COPY (continued)

/ODENSITY=mode
Control the magnetic tape density of output files. Use this switch with MTB, model 6026 tape drives only. These are your mode options:

MODE	DENSITY
800	800 BPI
1600	1600 BPI
ADM	Automatic Density Matching

/OMTRSIZE=block-size (bytes)
Control the magnetic tape block size for output files.

/V
Display the pathname of the source file copied.

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 and use default specifications to create TESTA.

```
) COPY/V TEST1 TEST2↵
TEST2
)
```

Copy TEST2 to TEST1, using the /V switch to display the source file's pathname.

CPUID (AOS/VS only) *Command*

Format

CPUID

Purpose

Display the CPU identification (AOS/VS only).

This command displays your computer's central processing unit (CPU) identification. For a description of this number, consult the *ECLIPSE® MV/8000 Principles of Operation manual.*

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Command page.

Argument Switches

None.

Example

```
)CPUID↓
CPUID 4437000407
)
```

Format

CREATE pathname *[resolution-pathname]*

Purpose

Create a file.

If you omit switches for this command, the system creates a text file that you can use for text or source code. When you create a link entry to a file, **pathname** is the link name you'll use to access the resolution file, and *resolution-pathname* is the resolution file's 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 the /MAXSIZE= switch is also used, create a control point directory (type CPD) with the specified maximum size.

/DYNAMIC
Create the file with dynamic record format.

/ELEMENTSIZE=n
Set the file element size to the specified value. Under AOS, the default size is 1 block. Under AOS/VS, the default size is 4 contiguous blocks, or the value chosen at system generation.

CREATE (continued)

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

/HASHFRAMESIZE=n
Set the hash frame size for this directory or control point directory. The default hash frame size is 7.

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

/INDEXLEVELS=n
Set the maximum number of index levels to the specified value. The default is 3.

/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 body unit must contain a single).

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

/TYPE=type
Create a file of type **type**.

type can be in the following forms:

XXX 3-letter mnemonic

n decimal number (64-255)

/VARIABLE
Create the file with variable record format.

Argument Switches

None.

Examples

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

Create a FORTRAN IV source file named PROG.FR.

```
) CREATE/LINK LNAME :UDD:USERNAME:FILE1↓
)
```

Create a link file containing a complete pathname to FILE1.

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

Create a directory called PROJECT1.

CURRENT *Command*

Format

CURRENT

Purpose

Display the current CLI environment's settings.

Command Switches

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

Argument Switches

None.

Example

```
) CURRENT
LEVEL                   0
SUPERUSER               OFF
SUPERPROCESS            OFF
SCREENEDIT              OFF
SQUEEZE                 OFF
CLASS1                  WARNING
CLASS2                  ERROR
TRACE
VARIABLES               0  0  0  0  0
                        0  0  0  0  0
LISTFILE                @LIST
DATAFILE                @DATA
LOGFILE
DIRECTORY               :UDD:JOHN
SEARCHLIST              :UDD:JOHN,:PER
DEFACL                  JOHN,OWARE
STRING
PROMPT
CHARACTERISTICS         /605X/LPP=24/CPL=80
                        /ON/ST/EB0/ULC/WRP
                        /OFF/SFF/EPI/8BT/SPO
                        /RAF/RAT/RAC/NAS/OTT
                        /EOL/UCO/LT/FF/EB1/PM
                        /NRM/MOD/TO/TSP/PBN
                        /ESC/FKT/NNL
)
```

Display the current CLI environment's settings.

DATAFILE *Command*

Format

DATAFILE *[pathname]*

Purpose

Set or display the current DATAFILE pathname.

Set the DATAFILE to pathname. The DATAFILE is passed to any process created by an EXECUTE, XEQ, or DEBUG command. The CLI itself does not use the DATAFILE. When coding a program that must open and use a data file, you can use the generic filename (@DATA) within your program instead of the specific name. Then, before you create a process, you can specify the DATAFILE pathname which you want the CLI to pass as the generic @DATA to the created process.

Command Switches

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

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

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

/P
Set DATAFILE 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.

!DATAFILE *Pseudo-Macro*

Format

[!DATAFILE]

Purpose

Expand to the pathname of the current DATAFILE.

This pseudo-macro does not accept arguments.

Macroname Switches

/P
Expand to previous environment's data file pathname.

Argument Switches

None.

Examples

```
) WRITE THE CURRENT DATA FILE IS [!DATAFILE]↓
THE CURRENT DATA FILE IS @DATA
) PUSH↓
) DATAFILE :UDD:USER:WORK↓
) WRITE NOW THE CURRENT DATA FILE IS&↓
&)[!DATAFILE]↓
NOW THE CURRENT DATA FILE IS
:UDD:USER:WORK
) WRITE [!DATAFILE/P]↓
@DATA
)
```

First, evaluate [!DATAFILE] and write the current data file pathname, which is the generic @DATA. Then change environment, and set a new data file for the new environment. Evaluate and write [!DATAFILE] for the current environment, and then, using the /P switch, for the previous environment.

DATE *Command*

Format

DATE *[date]*

Purpose

Set or display the current system date.

The date can be set only by username OP (process-ID 2), the initial CLI. Use one of the following formats for date:

11 26 80
26-NOV-80

In the second format, you can use an abbreviation if it uniquely identifies the month.

Command Switches

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

Argument Switches

None.

Examples

```
) DATE 11 26 80↵
) DATE↵
26-NOV-80
)
```

Set and display the date.

```
) DATE 26-N-80↵
) DATE↵
26-NOV-80
)
```

Set and display the date. (Note that you can use N because no other month begins with N).

!DATE *Pseudo-Macro*

Format

[!DATE]

Purpose

Expand to the current system date.

This pseudo-macro does not accept arguments.

Macroname Switches

None.

Argument Switches

None.

Example

```
) WRITE TODAY IS [!DATE].)
TODAY IS 26-NOV-80.
)
```

Format

DEASSIGN character-device *[character-device]...*

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.

You may use templates in the character device argument.

Command Switches

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

Argument Switches

None.

Example

```
) DEASSIGN @CRA1)
)
```

Deassign the alternate card tape reader, that you previously assigned.

DEBUG *Command*

Format

DEBUG pathname *[argument-to-new-program]...*

Purpose

Execute the specified program and enter the debugger.

DEBUG creates a subordinate process that executes the program named in pathname (it must be a program file). The new program starts in the debugger. The arguments are placed in the initial IPC message to the new process. The new process can access these arguments through the ?GTMES system call.

The CLI first tries to debug pathname.PR. If this fails, the CLI tries pathname.

See the EXECUTE command for additional information about subordinate processes. For more on DEBUG, see the *AOS Debugger and File Editor User's Manual* or the *AOS/VS Debugger and File Editor User's Manual.*

Command Switches

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

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

/M
Create input for the program from the macro body. The last line of input must contain a single).

/S

Return the termination message to STRING.

Argument Switches

These are explained in the *AOS Debugger and Disk File Editor User's Manual* and in the *AOS/VS Debugger and File Editor User's Manual.*

Example

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

) DEBUG VSPROGRAM↵

AOS/VS USER DEBUGGER- REV. XX

00000000000 00000000000 00000000000 00000000000
00000000000

_$Z
)
```

Enter the Debugger and execute MYPROGRAM. Always enter the Debugger *before* execution.

!DECIMAL *Pseudo-Macro*

Format

[!DECIMAL octal-number]

Purpose

Convert an octal number to decimal.

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

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

Edit disk file locations (AOS only).

DEDIT invokes the Disk File Editor utility, that allows you to examine and change the contents of disk file locations. The DEDIT utility uses a subset of AOS Debugger commands. It is functionally identical to the AOS Debugger except that it cannot set breakpoints or examine accumulators. For more on DEDIT, consult the *AOS Debugger and Disk File Editor User's Manual.*

DEDIT Switches

/I=pathname
Take DEDIT commands from pathname. Build a file of DEDIT commands and execute it with a single CLI command. Terminate commands in the file normally; i.e., with either NEW LINE or carriage return.

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

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

DEDIT (continued)

Example

```
) XEQ DEDIT DIR1:MYFILE.PR↵
+ START: 001762 (CR or LF)
.
.
+ BYE↵
)
```

This sequence executes DEDIT on user program MYFILE in directory DIR1; DEDIT then prints its prompt (+) and accepts editing commands. At the session's end, the BYE command returns control to the CLI.

Format

DEFACL *[username,access]...*

Purpose

Set or display the default access control list.

To display the current default access control list (ACL), enter DEFACL without arguments. The system default ACL is username, OWARE. When you create a new file, this is usually its ACL.

To change the current default ACL, enter the username and specify new access values.

Command Switches

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

/D
Return to the system default ACL (username, OWARE).

/K
Set the default ACL to no ACL (no arguments allowed). This denies everyone except Superusers access to your files until the ACL changes again.

/P
Set default ACL to the previous environment's default ACL (no arguments allowed).

DEFACL (continued)

Argument Switches

None.

Examples

```
) DEFACL↵
COSTLEY, OWARE
) DEFACL COSTLEY,R↵
) DEFACL↵
COSTLEY,R
) DEFACL/D↵
) DEFACL↵
COSTLEY,OWARE
)
```

The first command displays the current default ACL, that happens to be the system default ACL. The second command changes the ACL to Read access only. The next command, displays the new default ACL. Then the following command uses the /D switch to change the ACL back to the system default ACL. Finally, the last command displays the current default ACL once again.

Format

[!DEFACL]

Purpose

Expand to the current user default access control list.

This pseudo-macro does not accept arguments.

Macroname Switches

/P
Use the previous environment's default ACL.

Argument Switches

None.

Example

```
) WRITE THE CURRENT USER DEFAULT ACL IS&↓
&)[!DEFACL]↓
THE CURRENT USER DEFAULT ACL IS
COSTLEY,OWARE
)
```

First, evaluate the pseudo-macro [!DEFACL], then write the resulting argument list on the terminal.

DELETE *Command*

Format

DELETE pathname *[pathname]...*

Purpose

Delete one or more files.

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

You may use templates in the pathname arguments.

Delete one or more files.

Command Switches

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

/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 NEWLINE to retain file.

/V
Verify each deletion with a list of the files deleted.

Argument Switches

None.

Examples

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

Format

XEQ DGL source-pathname *[argument]...*

Purpose

Compile a DG/L™ source file.

The DGL compiler looks for source-pathname. By default, executing the command produces an object file named source-pathname.OB, and produces no listing.

The optional arguments, with the appropriate switches appended, can specify the listing, error, and object files, or various code generation and selective compilation options.

For a complete description of the DG/L programming language and the CLI DGL command line, see the *DG/L ™ Runtime Library (AOS and AOS/VS) User's Manual* and the *DG/L Reference Manual.*

DGL Switches

/E=pathname, /L, /L=pathname.
See CLI Commands page.

/A
Continue compilation past the phase that catches syntax errors, even if syntax errors are found. Without this switch, errors found in the syntax phase cause the compiler to skip the other phases.

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

/C

Check syntax of source text, but do not check semantics or generate code. Since syntax checking takes less time than a full compilation, this option is useful in early program development.

/CODE=symbol

Generate code for the specified machine/operating system combination. You may generate AOS or RDOS code on AOS/VS, and RDOS code on AOS. The legal values for symbol are as follows (note that each machine/operating system combination has several symbols, and that some symbols designate more than one combination):

SYMBOL	MACHINE	OPERATING SYSTEM
N	NOVA	RDOS
NOVA	NOVA	RDOS
E	ECLIPSE	RDOS
ECLIPSE	ECLIPSE	RDOS
RDOS	ECLIPSE	RDOS
A	ECLIPSE	AOS
AOS	ECLIPSE	AOS
16	ECLIPSE	AOS
X16	ECLIPSE	AOS
VS16	ECLIPSE	AOS
A	MV/8000-16	AOS/VS
AOS	MV/8000-16	AOS/VS
16	MV/8000-16	AOS/VS
X16	MV/8000-16	AOS/VS
VS16	MV/8000-16	AOS/VS
X	MV/8000-32	AOS/VS
X32	MV/8000-32	AOS/VS
32	MV/8000-32	AOS/VS
VS32	MV/8000-32	AOS/VS

If you omit this switch, DG/L generates code for the current environment.

/DEBUG
Produce and DL blocks in the object file for use by the SWAT debugger.

/F
Produce an error message if the compilation results in an attempt to generate a floating-point instruction.

/G
Create local symbols out of line numbers, and global symbols out of procedure names. The debugger can use these symbols.

/H
(For NOVA ® code only.) Generate code usable on a target machine that has hardware multiply/divide instructions (otherwise, DG/L will use software multiply/divide).

/I
Do not list the contents of INCLUDE files on the compilation listing.

/INNER
Produce code that is to be linked for an inner (4-6) ring, and that can be called through a gate array from an outer ring. This switch is valid only when /CODE= has the value X, X32, or VS32

/M
Generate code that will run on a mapped ECLIPSE ® machine. That is, the compiler can use short LEF instructions. The compiler assumes /M when it generates code for AOS systems, but not for ECLIPSE RDOS.

/N
Do not generate object code, but proceed otherwise with all phases of compilation.

/NOLEF
Do not use short LEF instructions in the code. This switch is the opposite of /M.

/O=pathname
Write the object file to pathname.

/OPT=string
Perform conditional compilation; that is, compile lines surrounded by /**xxx*/ where xxx are any characters contained in the string specified with this switch. This DGL switch has the same effect as the /O argument switch.

/P
During compilation, assume that the correct number of arguments will always be passed to all external procedures. Compiling with /P eliminates the need for many runtime checks.

/Q
Allow the use of question marks (?) in identifier names.

/R
Use floating-point arithmetic to perform all integer division within subexpressions. This increases accuracy by reducing rounding and truncation errors.

/REV=rev-num
Enter a revision number for an .OB or .RB object file. The default value is the current DG/L compiler revision number. For RDOS, AOS, or 16-bit AOS/VS, the format for rev-num is major-rev-num*[.minor-rev-num]*. For 32-bit AOS/VS, the format is major-rev-num*[.minor-rev-num[.update num [.pass-num]]]*. For RDOS code, each number must be less than 100; for AOS and AOS/VS, each number must be less than 256.

/S
Generate code for full subscript checking. Note that an object program without subscript checking runs faster and requires less memory.

DGL (continued)

/T
Generate code for string overflow checking.

/TEMP=directory
Put temporary files in this directory. If the directory is on a fixed-head disk, compilation may be faster.

/V
Add information to the listing, giving the status of each line, i.e., the block level, whether the line is from an INCLUDE file, and whether the line compiled conditionally.

/W
Produce warning messages during compilation.

/WSAVS
(For MV/8000-32 only) Generate WSAVSs instead of WSAVRs.

/X
Generate a full cross-reference table, including constant references. Without this switch, only variables are cross-referenced.

/Y
Put constants into the shared code area, instead of the shared data segments of memory. This switch applies to AOS and AOS/VS-16 programs containing overlays.

/Z
Find all EXTERNAL integers in page zero of memory. This lets the compiler generate shorter object code.

Argument Switches

Note that each argument switch has an equivalent DGL command switch.

/B
Write the binary object code to the file specified by this argument. This argument switch has the same effect as the /O=pathname DGL command switch.

symbol/C
Generate code for the specified machine/operating system combination. This argument switch can be appended to any of the codes listed under the /CODE= DGL command switch. If you omit this switch, DG/L generates code for the current environment. This argument switch has the same effect as the /CODE= DGL command switch.

/E
Write error messages to the file specified by this argument. This argument switch has the same effect as the /E=pathname DGL command switch.

/L
Write a listing to the file specified by this argument. This argument switch has the same effect as the /L=pathname DGL command switch.

string/O
Perform conditional compilation, using the string specified by this argument. This argument switch has the same effect as the /OPT=string DGL command switch.

DGL (continued)

Example

```
) XEQ DGL/G/L=TEST.LS/E=TEST.E/V TEST.DG
```

The DGL command switch /G creates a local symbol block in the .OB file, to be used for debugging. The /L switch sends the listing to file TEST.LS, and /E writes the error messages to TEST.E. The /V switch causes additional information about the lines of code to be written to the listing file: the block level, whether the line is from an INCLUDE file, and whether it compiled conditionally.

Format

DIRECTORY *[pathname]*

Purpose

Set or display the current directory setting.

Command Switches

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

/I
Set working directory to initial working directory.

/I=pathname
Set working directory to directory specified by pathname, which starts from the initial working directory.

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

Argument Switches

None.

DIRECTORY (continued)

Examples

```
) DIRECTORY↓
:UDD:USER
) DIRECTORY BETA↓
) DIRECTORY↓
:UDD:USER:BETA
) DIRECTORY/I GAMMA:DELTA↓
) DIRECTORY↓
:UDD:USER:GAMMA:DELTA
) DIRECTORY/I↓
) DIRECTORY↓
:UDD:USER
)
```

First, display the working directory's pathname. Next, make BETA the working directory and display its pathname. Then, using the /I switch and a pathname, make DELTA the working directory. Finally, use the /I switch without an argument to set the working directory to the initial directory.

Format

[!DIRECTORY *[pathname]]*

Purpose

Expand to the current or previous environment's working directory.

Macroname Switch

/P
Use previous environment's working directory.

/I
Expand to the initial working directory setting.

/I=pathname
Expand to the directory specified by pathname, which starts from the initial working directory setting.

Argument Switches

None.

!DIRECTORY (continued)

Example

```
) WRITE THE WORKING DIRECTORY IS&↓
&)[!DIRECTORY].↓
THE WORKING DIRECTORY IS :UDD:DAN :
ALPHA.
) WRITE [!DIRECTORY/I]↓
:UDD:DAN
) WRITE [!DIRECTORY/I TEST]↓
:UDD:DAN:TEST
)
```

Format

DISCONNECT process-ID

Purpose

Break a customer-server connection.

The process ID must be the PID of a server process to whom you have previously been connected.

Command Switches

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

Argument Switches

None.

Example

```
) DISCONNECT 12
)
```

Disconnect the user from the server process whose ID is 12.

DISMOUNT *Command*

Format

DISMOUNT *linkname [message]*

Purpose

Request operator to dismount a tape.

This command requests the operator to dismount a tape. Specify the same **linkname** you used to mount the tape.

NOTE: If you issue a **DISMOUNT** command for a magnetic tape, the system will rewind the tape and restore the drive's ACL to what it was before the mount.

Command Switches

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

Argument Switches

None.

Example

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

Format

XEQ DISPLAY *input-pathname [destination-pathname]*

Purpose

Print a file in octal and ASCII values.

This utility produces a copy and/or a listing of the input file.

The default listing format is 16 input characters per line, printed first as octal values and then as text. ASCII characters that cannot be printed as text (those whose octal value is less than 40 and greater than 176_8) are replaced by a period(.). Repeated identical lines are indicated by ****.

The default values for the input parameters are FIRST=0, INCREMENT=1, and LAST=32767. If the input is on tape, the tape input block size is the size specified for the device when the system was generated.

If you specify a tape destination pathname, the default values for the output parameters are: OBLOCKSIZE=the block size of the input tape; and ODENSITY=0. DISPLAY will create the destination file if it does not already exist. If it does exist, DISPLAY will delete the file and then recreate it.

DISPLAY Switches

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

/ALL
Process all files to logical end-of-tape on the input tape. Use this switch only for input files on tape. The destination file may be either another tape or a disk file. The tapes must be specified without a file.

DISPLAY (continued)

/APPEND
Append the output to the destination file if it exists. The default is to first delete the destination file and then recreate it.

/BYTE
List the file in byte format (octal values) with no text.

/CONVERT
Convert EBCDIC input into ASCII.

/DECIMAL
List the file in decimal values, not octal.

/FIRST=m
First block to be processed, where m is greater than or equal to zero and less than or equal to 32767.

/HEXADECIMAL
List the file in hexadecimal values rather than octal values.

/IGNORE
Ignore the logical end-of-tape on the input file.

/INCREMENT=i
Process every i**th block in the input file.

/IPHYSICALEOT
Ignore physical end-of-tape on the input file and go to the next end-of-file.

/LAST=n
Last block to be processed, where n is greater than m and less than or equal to 32767.

/LISTUDA
List the UDA of the input file.

/NOLIST
Suppress the listing file.

/NUMERICONLY
List only the numeric value of the input file, and not the text value.

/OBLOCKSIZE=b
Specify the block size of the destination tape, where b is the number of bytes.

/ODENSITY=d
Specify the density of the destination tape. Use this switch only with MTBs. The following are valid values for d:

Value	Density
800	800 BPI
1600	1600 BPI
ADM	Automatic Density Matching

/TEXTONLY
List only the text value of the input file, and not the numeric value.

/UPPERCASE
Convert any lowercase characters to uppercase in the text part of the listing.

/WIDTH=j
Set the width of the lines in listing file, where j represents the number of characters per line. j must be an even number and less than or equal to 124.

DISPLAY (continued)

Examples

```
) XEQ DISPLAY/L=@LPT DATA76↵
```

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

```
) XEQ DISPLAY/ALL/HEXADECIMAL @MTA0↵
```

Produce a hexadecimal and ASCII listing of all the files on MTA0.

```
) XEQ DISPLAY/CONVERT/NOLIST @MTA0 DFILE↵
```

Create an ASCII file on disk of the first EBCDIC file on the tape.

Format

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

Purpose

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

You may use templates for the *source-pathname* argument(s). If you supply no source pathnames, the template # is assumed. Unless you use the /FLAT switch, the directory structure of the dumped files will be maintained.

Command Switches

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

/V
Verify dumped files on @ OUTPUT.

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

/NACL
Dump files without ACLs (later when you LOAD the files, they will be given default ACLs).

/RETAIN=days
Set the retention period of a labelled tape to days. This switch applies only to labelled tape. It will have no effect on other kinds of dump tapes.

DUMP (continued)

/DENSITY=mode
Control the magnetic tape density of your dump tape. Use this command with MTB, model 6026 tape drives only. The following are your mode options:

MODE	DENSITY
800	800 BPI
1600	1600 BPI
ADM	Automatic Density Matching

/BUFFERSIZE=bytes
Blocks dumped to the tape will have length of n bytes.

/BEFORE/TLM=
date:time
Dump only the files modified before the specified time, date, or date:time.

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

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

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

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

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

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

NOTE: You may specify a range of dates by using both the /BEFORE and /AFTER switches. However, you must use the same modifier for both: either /TLM= or /TLA= but not both /TLM= and /TLA= at the same time.

/TYPE=type
Select all files of the specified type. Types are provided in Table 2-3. Type can be in the following forms:

XXX	3-letter mnemonic
n	decimal number (0, 10-13, or 64-255)
m-n	decimal numbers which define a range of file types
\n	decimal number to exclude
\m-n	decimal numbers which define a range of file types to exclude

You can use more than one /TYPE= switch in a command line; for example

/TYPE=64-68/TYPE=\66

selects types 64, 65, 67 and 68

Argument Switches

None.

DUMP (continued)

Examples

```
) DUMP/V/FLAT/NACL FILE6.DUMP :UTIL:+.PR↵
19-JUN-81 10:19:50
    DISPLAY.PR
    SPEED.PR
    EXEC.PR
    XLPT.PR
    LINK.PR
    SED.PR
)
```

Dump all program files in the utilities directory, without their ACLs, to disk file FILE6.DUMP; verify dumped files.

```
) DUMP/AFTER/TLM=4-JUL-80:11:03:42 @MTA0:3↵
)
```

Dump all files in the working directory that were last modified after 11:03:42 a.m. on July 4, 1980 to file 3 of the tape mounted on @MTA0.

Format

[!EDIRECTORY pathname *[pathname]...*]

Purpose

Expand to the directory portion of a pathname.

This pseudo-macro expands to the directory portion of a pathname. By directory portion, we mean the pathname string from the leftmost character up to, but excluding, the rightmost colon or prefix character.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE [!EDIRECTORY :UDD:AOS:MODS:SCALL.SR]↓
:UDD:AOS:MODS
) WRITE [!EDIRECTORY @CON32 =MODS:FILE
:CLI.PR]↓
@ =MODS :
)
```

The first example shows the directory portion of the full pathname :UDD:AOS:MODS:SCALL.SR. The second example shows the directory portions of the three pathnames used as input to the !EDIRECTORY pseudo-macro.

!EEXTENSION *Pseudo-Macro*

Format

[!EEXTENSION pathname *[pathname]...*]

Purpose

Expand to the extension portion of a pathname.

This pseudo-macro expands to the extension portion of a pathname. By extension portion, we mean the input string from the last period after the rightmost colon or prefix character to the end of the string, inclusively. If there is no period in the string, this pseudo-macro returns a null.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE [!EEXTENSION :UDD:AOS:MODS:SCALL.SR]↓
.SR
) WRITE [!EEXTENSION :UDD:CLI:SRC.SR.BU]↓
.BU
) WRITE [!EEXTENSION @CON32 =MODS:FILE
:CLI.PR]↓
.PR
)
```

The first example shows the extension .SR. The second example shows .BU, the characters that follow the last period after the rightmost colon. The third example shows three pathnames. Since only the last pathname has an extension, the CLI returns the proper extension for that pathname, and null for the first two pathnames.

!EFILENAME *Pseudo-Macro*

Format

[!EFILENAME pathname *[pathname]...*]

Purpose

Expand to the filename portion of a pathname.

This pseudo-macro expands to the filename portion of a pathname. By filename portion, we mean the input string from the rightmost colon or prefix character to the end of the string. The filename includes any extension.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE [!EFILENAME :UDD:AOS:MODS:SCALL.SR]
SCALL.SR
) WRITE [!EFILENAME :UDD:CLI:SRC.SR.BU]
SRC.SR.BU
)
```

Format

[!ELSE]

Purpose

Include CLI input conditionally.

This pseudo-macro does not accept arguments. You can use [!ELSE] only after one of the pseudo-macros that begin conditional branching.

If the condition stated in the initial pseudo-macro is true, then the CLI executes the input lines that appear before the !ELSE pseudo-macro and does not execute the input lines that appear after !ELSE. If the initial condition is false, then the CLI skips the input between the initial pseudo-macro and !ELSE, and executes the input lines which appear after !ELSE up to the next !END pseudo-macro.

Macroname Switches

None.

Argument Switches

None.

!ELSE (continued)

Examples

```
) [!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(s) the !ELSE pseudo-macro.

Format

[!ENAME pathname *[pathname]...*]

Purpose

Expand to the name portion of a pathname.

This pseudo-macro expands to the name portion of a pathname. By name portion, we mean the input string after the rightmost prefix (: @ = &?) up to, but excluding, the rightmost period. If there are no prefix characters in the string, the name begins with the leftmost character in the string. If there is no period in the string, the name ends at the rightmost character.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE [!ENAME :UDD:APS:MODS:SCALL.SR]↓
SCALL
) WRITE [!ENAME :UDD:CLI:SPR.SR.BU]↓
SPR.SR
) WRITE [!ENAME @CON32 =MODS:FILE :CLI.PR]↓
CON32 FILE CLI
)
```

Format

[!END]

Purpose

End an !EQUAL or !NEQUAL macro loop.

This pseudo-macro does not accept arguments. Use !END to terminate the sequence of CLI input that follows one of the conditional branching pseudo-macros. See Chapter 5 for a discussion of conditional pseudo-macros.

Macroname Switches

None.

Argument Switches

None.

Examples

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

The CLI executes the first command line because the condition is true; it does not execute the second because the condition is false.

Notice that there are no spaces between the [!EQUAL] or [!END] statement and the WRITE command.

ENQUEUE *Command*

Format

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

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.

You may use templates in the pathname arguments.

Command Switches

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

/V
Display the names of the queued files.

Argument Switches

/B
Output in binary mode; do not interpret special control characters (such as TAB); default is output in text mode.

/D
Delete disk file after output; by default, the system retains the 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 file; default is immediate output.

If you specify /P (and no /MES=x), the spooler process displays

PAUSE FROM queuename

on the operator terminal before output.

If you specify /MES=x (and omit /P), the spooler process displays the message

FROM queuename:

on the operator terminal and then outputs the file.

If you use both /MES=x and /P, then, before it starts the output, it prefixes the message displayed on the operator terminal with

PAUSE FROM queuename.

The operator must respond with an appropriate control command, such as

```
) CONTROL @SPOOL CONTINUE
```

to start file output.

ENQUEUE (continued)

Example

```
) ENQUEUE @LPT OUTPUT.LS/P/MES=TWO_&
&)PART_FORMS
)
```

Queue OUTPUT.LS to the line printer. Before the line printer begins, the system pauses and displays the message TWO_PART_FORMS.

Format

[!EPREFIX pathname *[pathname...]*]

Purpose

Expand to the prefix portion of a pathname.

The prefix is the input string from the leftmost character up to, and including, the rightmost prefix character (: @ = &?). If there is no prefix character in the string, the pseudomacro returns a null.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE [!EPREFIX :UDD:AOS:MODS:SCALL.SR]↓
:UDD:AOS:MODS:
) WRITE [!EPREFIX @CON32 =MODS:FILE :CLI.PR]↓
@ =MODS: :
) WRITE [!EPREFIX MYFILE]↓
)
```

!EQUAL *Pseudo-Macro*

Format

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

Purpose

Include input conditionally.

This pseudo-macro begins a sequence of text that the CLI is to conditionally execute. You must end the sequence with the !END pseudo-macro. The sequence can optionally include the !ELSE pseudo-macro.

The !EQUAL pseudo-macro must always have two arguments. !EQUAL compares the two arguments character by character. If they match, the CLI executes the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI doesn't execute the input following it up to the !END pseudo-macro.

If the arguments don't match, the CLI does not execute the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI executes the input following the !ELSE up to the !END.

Macroname Switches

None.

Argument Switches

None.

Examples

Given a macro containing

```
[!EQUAL,%1%,*]
WRITE STAR
[!ELSE]
WRITE NOT STAR
[!END]
```

This will write STAR if you call the macro with the argument *; otherwise, it will write NOT STAR.

Note that you can also code the macro as follows:

```
WRITE [!EQUAL,%1%,*]STAR[!ELSE]NOT STAR[!END]
```

Notice that we used commas to separate the arguments in the !EQUAL pseudo-macro. If we used spaces, and argument$_1$ was null (or not present), the spaces on either side of the %1% would have become a single delimiter, giving [!EQUAL,*]. This format is invalid, since the !EQUAL pseudo-macro takes exactly two arguments. Notice that there are no spaces between the bracketed !EQUAL statement and other commands and arguments.

EXECUTE *Command*

Format

EXECUTE pathname *[argument-to-new-program]...*

Purpose

Execute a program. EXECUTE is identical to XEQ or X.

The CLI creates a subordinate swappable process with the same priority and privileges that it has. It takes the subordinate process's program from the program file specified in the command line. Arguments to the new program are placed in the initial IPC message to the new process. The program can access these arguments by using the ?GTMES system call. The subordinate process's generic @INPUT, @OUTPUT, and @CONSOLE are the same as its parent's (the CLI's). The subordinate process's generic @LIST and @DATA files are the current list file and data file settings. (These are not necessarily the same as the CLI's generic @LIST and @DATA files.)

The CLI first tries to execute pathname.PR. If that fails, the CLI tries pathname.

The CLI is blocked until the subordinate process terminates. The CLI may take exceptional action depending on whether or not the subordinate process returns exceptional condition flags.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Command page.

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

/M

Create input for pathname from the macro body. The last line of the macro body must contain a single).

/S

Store the program termination IPC message in the current STRING (instead of displaying it to @OUTPUT).

Argument Switches

Use any argument switches appropriate for the program specified in pathname.

Examples

) EXECUTE MYSORT↓

Run a program named MYSORT.

) EXECUTE/S PROG1↓

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

!EXPLODE *Pseudo-Macro*

Format

[!EXPLODE argument *[argument]...*]

Purpose

Expands arguments into single character arguments.

!EXPLODE interprets its arguments as a single string and converts all spaces and tabs into commas (the CLI delimiter) in accordance with standard CLI rules. !EXPLODE expands the string and inserts commas between every pair of characters (including commas) in the original string. Use !EXPLODE to gain access to characters of arguments as individual arguments.

Macroname Switches

None.

Argument Switches

None.

Examples

[!EXPLODE ABC]

Expands to *A, B, C*

[!EXPLODE A]

Expands to *A*

[!EXPLODE A,C]

Expands to *A,,,C*

[!EXPLODE[!TIME]]

Expands to *0,9,:,2,3,:,4,7*

[!EXPLODE[!DATE]]

Expand to *2,6,-,N,O,V,-,8,0*

In a macro
WRITE [!EXPLODE ABC]

The system responds with

A B C

FCU *Utility*

Format

XEQ FCU

Purpose

Set nonstandard forms parameters for files to be printed.

FCU (Forms Control Utility) specifies horizontal tabs and vertical forms settings for a user file or a forms entry in directory :UTIL:FORMS. Create a forms entry in any directory, but the file must reside in :UTIL:FORMS for EXEC to use it. Run FCU to create, edit, or list forms control specifications.

To create forms control specifications, you must have created a file before invoking the FCU; the FCU does not itself create the file. The specifications are part of the file's UDA. If the file size was 0 before you invoked the FCU, it will still be 0 after you have created the specifications.

FCU Commands

B
Terminate the Forms Control Utility.

C
Create forms control specifications for an existing file.

E
Edit forms control specifications for a file.

H
Display all FCU commands.

L

Print a file's forms specifications to current list file. You must have previously set list file or have executed FCU with the /L= switch.

T

Type forms control specifications on the terminal.

FCU Switches

/L= listfilename

List the forms control specifications for a file (L command). If you try to list a file wothout appending this switch to the FCU or setting the CLI list file you get an error message.

Argument Switches

None.

Example

```
) XEQ FCU
      .
      .
      .
FCU TERMINATING 26-NOV-80          13:05:00
)
```

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

Format

XEQ FED pathname

Purpose

Edit disk file locations.

FED calls the File Editor utility and allows you to examine and modify locations in AOS/VS disk files. (Use DEDIT for AOS disk files.) For more information, see *AOS/VS DEBUG and FED USER'S MANUAL*

FED Switches

/I=filename
Use the commands in filename for the editing session. By using this switch, you can build a file of FED commands and execute them all at once by issuing a single command. Your file must end with a BYE command.

/L=filename
Save all FED commands and responses in filename. If filename does not exist, FED creates it; if it does exist, FED appends the new information to the existing file.

/N
Do not open a symbol table (.ST) file.

/P
Treat the disk file as a program file.

/R
Open the file for read access only. Do not allow modification.

/S=filename
Use filename as the symbol table file.

/U
Treat the disk file as a user data file.

/X
Treat the disk file as an AOS/VS system file. Use this switch only to apply a patch released by Data General Corporation.

Example

```
) XEQ FED PROG1↓
FED- Disk File Editor- REV. xx
_
.
(FED commands)
.
.
_$Z
)
```

FILCOM *Utility*

Format

XEQ FILCOM $pathname_1$ $pathname_2$

Compare two files.

This utility compares $pathname_1$ to $pathname_2$. If the files differ, FILCOM displays the word number and the octal value of each different word for both files. If one file is longer than the other, the system displays all of the words in the longer file and dashes for the shorter file.

FILCOM Switches

/L,/L=pathname.
See CLI commands page.

Example

The following three files are in the working directory:

```
FILE1     FILE2     FILE3
ABCDEFG   ABCDEFG   ABCDEFG
HIJKLMN   ABCDEFG   HIJKLM
OPQRSTU   OPQRSTU   OPQRSTU
                    VWXYZ

) XEQ FILCOM FILE1 FILE2↵

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

) XEQ FILCOM FILE1 FILE3↵

          FILE1     FILE3
000014    ------    053127
000015    ------    054131
000016    ------    055012

)
```

First, compare FILE1 and FILE2, then compare FILE1 and FILE3. Note that two characters are packed into each word and that the word number in the left column is in octal.

!FILENAMES *Pseudo-Macro*

Format

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

Expand to a list of filenames.

You may use templates in the pathname argument(s). If used without arguments, this pseudo-macro expands to all filenames in the working directory (i.e., the template + is the default).

Macroname Switches

None.

Argument Switches

None.

Example

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

Create a separate batch job to assemble each file with a .SR suffix in the working directory.

Format

FILESTATUS *[pathname]...*

Purpose

Display status information for one or more files.

FILESTATUS lists all the specified filenames with the information requested in the switches. The default listing has a header for each directory the filenames are listed from, and lists only simple filenames.

Use filename templates in the pathname argument(s). If you omit arguments, the CLI displays the names and status of all the files in the working directory.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Command page.

File Information Switches

Use command switches to return other information about specific files; e.g.,

```
) FILESTATUS/TYPE/DCR MYFILE YOURFILE)
```

Different forms of these switches display filenames and statistics by selected group. If a switch doesn't apply to a specific file (e.g., /ELEMENTSIZE for a directory), the system ignores it and fills its field with dashes.

/ASSORTMENT
Display an assortment of fileinformation: file type, date and time of creation, and file length. If the file is a link, display file type, LNK, and link resolution name.

FILESTATUS (continued)

/DCR
Display file creation date.

/DLA
Display date file was last accessed.

/DLM
Display date file was last modified.

/ELEMENTSIZE
Display the number of disk blocks in the element for this file. A file element is the smallest unit by which a disk file can grow (a disk block = 512 bytes.)

/HASHFRAMESIZE
Display directory's hash frame size.

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

/LENGTH
Display file's byte length.

/LINKNAME
Display link's resolution name.

/PACKET
Display the entire contents of the packet returned by the ?FSTAT system call.

/PERMANENCE
Display PERM if a file or directory has its permanence on. Nothing is displayed if permanence is off.

/RECORD
Display 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 containing the size.

DS (DATA SENSITIVE) - records are terminated by specific characters embedded in the text.

/TCR
Display file creation date and time.

/TLA
Display date and time file was last accessed.

/TLM
Display date and time file was last modified.

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

FILESTATUS (continued)

/UDA
Display UDA if the file or directory has a User Data Area.

Specific Condition Switches

/BEFORE/TLM=
List only those files that were last modified before the specified time, date, or date:time.

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

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

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

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

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

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

You may specify a range of dates by using both the /BEFORE and /AFTER switches. You must use the same modifier for both: either /TLM= or /TLA= but not /TLM= and /TLA= at the same time.

/TYPE=type
Select all files of the specified type.

Type can be in the form:

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

You can use more than one /TYPE= switch in a command line; for example:

/TYPE=64-68/TYPE=\66

selects types 64, 65, 67, and 68.

Formatting Switches

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

FILESTATUS (continued)

/NHEADER
Do not print directory headers. Also list files with their complete pathnames; default is directory headers.

/SORT
Sort the filenames alphabetically.

Argument Switches

None.

Example

```
) FILESTATUS/ASSORTMENT/SORT↓

DIRECTORY :UDD:USER

 DDAY.DOC   TXT   6-JUN-80     6:00:00   30000
 DIR1       DIR   8-DEC-79    14:39:12   123456
 LINK2      LNK   :UDD:USERA
 PROG.PR    PRG   24-NOV-80    20:44:26   1324

)
```

Display a sorted list of filenames with an assortment of information including type, creation date and time, length in characters, and (for a link file) link resolution pathname.

Format

FORT4 source-pathname

Purpose

Compile a FORTRAN IV source file (AOS only). FORT4 is a macro that you use to compile a FORTRAN IV source file. The FORT4 command first searches for source-pathname.FR. If that is not found, it searches for source-pathname.

Output can be an assembled object file, an intermediate assembly language file, or a source listing. By default, FORT4 produces no listing, and its execution generates an intermediate source file, source-pathname.SR (compiler output), and an object file, source-pathname.OB (assembler output).

Each FORTRAN main program, external subroutine, and external function must be compiled separately. Once you have successfully compiled your program, use the BIND or LINK utility to load it.

For a complete description of the FORTRAN IV programming language and the CLI FORT4 command line, see the *FORTRAN IV User's Manual* and *FORTRAN IV Runtime Library User's Manual.*

FORT4 Switches

/L,L=pathname
See CLI Command page.

/A
Abort on system error.

FORT4 (continued)

/B
Produce a brief listing by compiling only the source program input.

/E[=pathname]
Write error messages to the file specified by pathname. Including /E without a pathname suppresses error messages. If you omit /E, error messages are written to the current output file.

/F
Make FORTRAN variable names and statement numbers equivalent to memory locations and offsets, so that they are recognizable to the debugger.

/N
Do not produce an object file.

/NA
Compile only; do not assemble.

/O=pathname
Name the object module **pathname**. If you omit this switch, the object name is **pathname.OB**.

/P
Process only the first 72 characters of each record.

/S=pathname
Save the intermediate source file (compiler output) and name it **pathname**. If you use /S but omit a pathname, the source is saved and named **source-pathname.SR**. If you omit /S, the source file is deleted after assembly.

/U
Output user symbols in the assembly phase.

/X
Compile statements with an X in column 1.

Argument Switches

None.

Examples

```
) FORT4/B MYPROG↓
```

Compile MYPROG.FR, writing a brief listing to the current list file. Because /E is omitted, all errors are sent to @OUTPUT. The compiler produces MYPROG.OB as the object file.

```
) FORT4 MAIN↓
) FORT4 XSUB1↓
) FORT4 XFUN↓
) FORT4 XSUB2↓
) XEQ BIND MAIN XSUB1 XFUN XSUB2 FORT.LB↓
```

Compile separately the four modules that make up the complete program. Then use the Bind utility to load the program. You must load the FORTRAN IV libraries with the program.

Format

F5 source-pathname

Purpose

Compile a FORTRAN 5 source file.

The F5 macro is used to compile a FORTRAN 5 source file. The macro first searches for source-pathname.FR. If that is not found, it searches for source-pathname.

Each FORTRAN 5 main program, subroutine, and subprogram must be compiled separately. Output will be an object file (binary output) named pathname.OB. By default, compilation produces no listing.

Link the separate FORTRAN 5 modules into an executable program by using the F5LD macro, which invokes the Link utility.

Under AOS/VS, FORTRAN 5 runs as a 16-bit process. In addition, F5LD is a link to the F5LDVS16.CLI macro.

For a complete description of the FORTRAN 5 programming language and the CLI F5 command line, see the *FORTRAN 5 Reference Manual* and the *FORTRAN 5 Programmer's Guide (AOS)*.

F5 Switches

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

/B
Produce a brief listing: the input source program, the storage map, the list of subprograms called, the cross-reference, and the error list. The generated code is not included.

/C

Check the syntax of the source program. The source program and error list are sent to the listing file, if one is specified. The error list is also sent to the error file, if one is specified.

/D

or

/DEBUG

Debug. Compile code that allows the long- form, error-traceback routine to output line numbers. Do not use this switch when compiling the final version of your program.

/E[=*pathname*]

or

/ERRORS

[=*pathname*]

Write errors to *pathname*. If you use /E without specifying a pathname, error messages are suppressed. If you do not use this switch, error messages are written to @OUTPUT.

/I

Do not list source lines from INCLUDE files.

/N

Do not produce an object file.

/NOLEF

Do not generate Load Effective Address instructions (LEFs). This switch is useful if you are using I/O instructions in assembly language routines combined with FORTRAN 5 programs.

/O

or

/OBJECT=pathname

Name the object file pathname.

/P

Use punched card format. The first 72 characters of each input line are used as FORTRAN 5 source code, although the entire line is sent to the listing file.

F5 (continued)

/S
or
/SUBCHECK
Generate code to check subscript references. A runtime routine determines if a reference is outside the range of an array.

/X
Compile lines with an X in column 1. If you do not use this switch, the compiler treats these lines as comments.

Argument Switches

None.

Examples

```
) F5 MYPROG
```

Compile either MYPROG.FR or MYPROG, depending on whether the source filename has the .FR extension. The compile produces the object file MYPROG.OB.

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

Compile either PROG or PROG.FR. Generate a listing file, PROG.LS, and do not include lines from INCLUDE files in the source listing. Suppress all error messages.

```
) F5 MAIN
) F5 SUB
) F5 XFUN
) F5 XSUB

) F5LD MAIN SUB XFUN XSUB
```

Compile separately the four modules that make up the complete program. Then use the macro F5LD.CLI to invoke the Bind utility, which builds the executable program. (This example is valid only for AOS. For an AOS/VS example, replace the last line with a line that invokes the Link utility.)

F5LD *Utility*

Format

F5LD objectmodule *[objectmodule]...*

Purpose

Link object modules to form an executable FORTRAN 5 program.

F5LD is a macro that invokes the LINK utility to make FORTRAN 5 object modules into an executable program. (Under AOS/VS, F5LD is actually a link to the macro F5LDVS16.CLI.)

In general, the object modules appear on the command line in the following sequence:

- Main FORTRAN 5 program
- User subprograms and optional user modules
- Support libraries (e.g., Commercial Subroutine Package)

In addition, if you use QCALLS within the FORTRAN 5 modules, you must add the FORTRAN 5 library F5ASYS.LB to the F5LD command line. For a complete description of the FORTRAN 5 programming language and the CLI F5LD command line, see the *FORTRAN 5 Reference Manual* and the *FORTRAN 5 Programmer's Guide.*

F5LD Switches

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

/B
Produce a listing of the symbol file, with symbols ordered both alphabetically and numerically.

/E
List all numbers in hexadecimal.

/P=pathname
Name the executable program file pathname.PR. By default, the file assumes the name of the first object module in the command line, plus the .PR extension.

Argument Switches

/S
Convert this shared code module to an unshared code module.

/U
Bind local symbols from this module into the symbol file. /U works only if you used it for this module in an earlier macroassembler command. The FORTRAN 5 compiler outputs no local symbols.

Example

```
) F5LD/L=NEWPROG.LM/P=NEWPROG.PR&J
&)MYPROGJ
```

Build compiled FORTRAN 5 module MYPROG.OB into an executable program named NEWPROG.PR. In addition, write a listing to the file named NEWPROG.LM.

Format

For AOS:

F77 source-pathname

For AOS/VS:

F77 source-pathname *[source-pathname]...*

Purpose

Compile a FORTRAN 77 source file.

The F77 macro is used to compile a FORTRAN 77 source file. The compiler looks first for a file named pathname.F77, and then for a file named pathname.

The F77 macro supplied with AOS automatically includes the /INTEGER=2 and the /LOGICAL=2 switches. You can override the automatic value by explicitly appending the switch to the command and specifying the value 4. If you append one of these switches to the F77 macro, you must use the entire switch name, not an abbreviation. (You can use unique abbreviations for all other AOS switches, and for all AOS/VS switches.) The macro supplied with AOS/VS does not include any such automatic values.

For a complete description of the FORTRAN 77 programming language and the CLI F77 command line, see the *FORTRAN 77 Reference Manual.*

F77 Switches

/E=pathname,/L,/L=pathname
See CLI commands page.

/CARDFORMAT
Impose card format. Read only the first 72 characters of each line, and pad lines with fewer than 72 characters. Treat all characters read as significant. You may place any other text, such as sequence numbers, in columns 73-80.

/CODE
Generate an assembly language listing of the program and write it to the file specified by the /L switch. If you omit /L, the /CODE switch is ignored.

/DEBUG
Generate symbols and code for later use by the SWAT high-level debugger. Do not use this switch when compiling the final version of a program.

/DOTRIP=1 or 0
If DOTRIP=1, force each DO loop to execute at least once. By default, the compiler generates DO loops that will not execute if they do not need to execute. Some user programs, written for early ANSI standards, expect the results produced by specifying /DOTRIP=1.

/INTEGER=2 or 4
Set the default integer length for this compilation to the specified number of bytes. The F77 macro supplied with AOS automatically contains the switch /INTEGER=2, but you can explicitly specify /INTEGER=4 to override the automatic value.

/LINEID
(AOS/VS only.) Generate code that will identify a line causing a runtime error. /LINEID includes the function of the /PROCID switch. Do not use this switch when compiling the final version of a program.

F77 (continued)

/LOGICAL=2 or 4
Set the default logical entity length to the specified number of bytes. The F77 macro supplied with AOS automatically contains the switch /LOGICAL=2, but you can explicitly specify /LOGICAL=4 to override the automatic value.

/N
Scan the source file as usual, but do not produce an object file. This switch is useful for a quick syntax check.

/NOMAP
(AOS/VS only) Do not include a storage map of program entities as part of the listing file.

/OPTIMIZE
[=1 or 2 or 3]
Set optimization to the specified level, ranging from 1 (lowest) to 3 (highest). If you omit the switch, the compiler performs no optimizations. If you include the switch but do not specify a level, the default level is 3. Do not use this switch in conjunction with /DEBUG, /PROCID, or /SUB.

/PROCID
(AOS/VS only) Include all procedure names in the object file. At runtime, FORTRAN 77 will be able to report the program unit in which an error occurs. /LINEID includes the function of the /PROCID switch.

/SAVEVARS

Generate code to save all entities on returns from subprogram units, and generate all storage as static. This switch has the effect of a SAVE statement in each unit. Named and blank COMMON blocks, values passed in dummy arguments, and values assigned with DATA are always saved, even if you do not use this switch.

/STATISTICS

Write statistics (such as number of lines compiled) to @OUTPUT. If you specify /L, the statistics are also included in the listing file.

/STRINGS=ANSI
or
DG

If /STRINGS=ANSI, accept angle brackets and the characters within them as literals. By default, the compiler interprets angle brackets as control characters. This switch, with the ANSI value, lets you use angle brackets as literal printed characters.

/SUB

Generate code for subscript and character substring checking. At runtime, the program reports any subscript or substring references that are out of range. Do not use this switch when compiling the final version of a program.

/XREF

(AOS/VS only) Generate an alphabetical listing of all program entities and include it in the listing. If you omit /L, the /XREF switch is ignored.

Argument

None.

F77 (continued)

Example

) F77 MYPROG↵

Compile either MYPROG.F77 or MYPROG, depending on whether the source filename has the .F77 extension. The compiler will describe all errors on @OUTPUT (the terminal), will not produce a listing file, and will generate object file MYPROG.OB.

) F77/SUB/XREF/L=@LPT PROG↵

Compile either PROG or PROG.F77. Generate code for subscript and substring checking. Generate a listing file with alphabetical cross references, and send the listing to the line printer.

) F77/OPTIMIZE PROG↵

Compile either PROG or PROG.F77. By default, optimization is set to 3, the highest level.

Format

F77LINK objectmodule *[objectmodule]...*

Purpose

Link object modules to form an executable FORTRAN 77 program.

F77LINK is a macro that invokes the Link utility to make FORTRAN 77 object modules into an executable program. Use this macro after you have compiled a FORTRAN 77 program with the F77 macro.

Neither the F77LINK macroname nor the switches can be abbreviated. Any LINK switch can be used as an F77LINK switch.

For a complete description of the FORTRAN 77 programming language and the CLI F77LINK command line, see the *FORTRAN 77 Reference Manual.*

F77LINK Switches

/CIS
(AOS only) Use the commercial instruction set (CMP, CMV, and FINT). This will allow faster character comparisons and truncation of real numbers. Use this switch only if you have the commercial instruction set.

/DEBUG
(AOS/VS only) Include code for the SWAT high-level debugger. For the SWAT debugger to work, you must have compiled using the /DEBUG switch.

F77LINK (continued)

/PRECONNECTIONS
=NONE
Sever all preconnections.

/PRECONNECTIONS=*
Use only the preconnections to units 5 and 6.

/PRECONNECTIONS=pathname
Use the user-defined preconnections in the .OB file specified by pathname.

/ROUND
(AOS/VS only) Direct the ECLIPSE MV/8000™ floating-point unit to use rounding for floating-point values. /ROUND is the default; /TRUNCATE is an option.

/TRUNCATE
(AOS/VS only) Direct the ECLIPSE MV/8000 floating-point unit to truncate floating-point values.

Argument Switches

None.

Example

```
) F77LINK MYPROG↵
```

Build compiled FORTRAN 77 module MYPROG.OB into an executable program, MYPROG.PR.

Format

HELP *[item]...*

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

The CLI contains information files that you can display at your terminal by typing the HELP command. However, some of these information files are longer than your screen. If you display such a file, you can freeze the display by typing CTRL-S. You can then read the screen at your leisure and, when you want to resume display, type CTRL-Q.

An item can be any one of the following:

- a CLI command
- a CLI pseudo-macro
- a general CLI topic
- a utility

Command Switches

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

/V
Display verbose description of the specified item. By default, HELP displays a terse description.

Argument Switches

None.

HELP (continued)

Example

```
) HELP MOUNT
MOUNT       - REQUIRES ARGUMENT(S)
SWITCHES: /1= /2= /L(=) /Q /VOLID=
/READONLY

FOR MORE HELP TYPE
HELP/V MOUNT
)
```

Format

[!HID *[hostname]*]

Purpose

Expand to a Host ID.

If no argument is given, !HID will return the ID of the local host. If you supply an argument, it must be a valid hostname. This pseudo-macro always returns a three-digit number.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE My host ID is [!hid]↓
My host ID is 014
) WRITE The host ID of NET_SYS17 is [!HID&↓
&)NET_SYS17]↓
The host ID of NET_SYS17 is 017
)
```

HOST *Command*

Format

HOST *[hostID]...*

Purpose

Display a System's hostname.

This command displays the hostname which corresponds with each of the *hostID* arguments. If you don't provide any arguments, HOST will display the hostname of the system on which you are running.

Command Switches

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

/S
Place the hostname in the current STRING.

Argument Switches

None.

Examples

```
) HOST)
NET_SYS1
) HOST 4)
NET_SYS4
)
```

Format

[!HOST */HID/*]

Purpose

Expands to a hostname.

If you don't provide an argument, the CLI returns the local hostname. If you do supply an argument, it must be a decimal number between 1 and 127.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE [!HOST]↓
NET_SYS1
) WRITE [!HOST 14]↓
NET_SYS14
)
```

INITIALIZE *Command*

Format

INITIALIZE physical-unitname *[physical-unitname...]*

Purpose

Graft a logical disk into the working directory.

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

Command Switches

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

/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. 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 10 and 14.

LABEL *Utility*

Format

XEQ LABEL device-name volid

Purpose

Prepare a magnetic tape with a volume label.

In order to use a labeled tape on AOS or AOS/VS, you must first use the LABEL utility to prepare the tape. This utility 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. See Chapter 2 for more information on labeled tapes.

device-name is the name of the tape (e.g, @MTA0 or @MTA1). Note that device-name should not be a labeled tape device (e.g., @LMT).

volid is an identifier from one to six characters long. You will refer to the tape later by this volume identifier.

LABEL Switches

/I
Create IBM labels. If you do not include this switch, the utility will create ANSI standard labels.

/S
Scratch the tape. This switch rewrites beginning and end-of-file labels for a null first file, effectively deleting all files on the tape. Include the volid.

/UVL=string
Write user volume label(s) after the volume label. The string cannot be longer than 76 characters. The system writes one user volume label for each occurrence of this switch. The limit is nine user volume labels.

/OWNER=string

Write string in the volume label's owner field. If you're writing ANSI labels, the maximum length of string is 14 characters. If you're writing IBM labels, the maximum length of string is 10 characters.

Example

```
) XEQ LABEL @MTA0 100000↵
```

Once the tape has been labeled, you can reference it by appending the volid and filename; e.g.,

```
) XEQ LABEL @MTA0 @LMT:100000:filename↵
```

LMT is the mnemonic for labeled mag tape; 100000 is the volume identifier; filename is the filename you wish to reference.

LEVEL *Command*

Format

LEVEL

Display the current CLI environment level number.

Use in conjunction with the PUSH and POP commands.

Command Switches

/1, /2, /L, /L=pathname, /Q
See CLI Command page.

Argument Switches

None.

Examples

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

Display current level. Push a level and display the current level again.

Format

[!LEVEL]

Purpose

Expand to the current CLI environment level number.

This pseudo-macro does not accept arguments.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE THE CURRENT LEVEL IS [!LEVEL]↓
THE CURRENT LEVEL IS 0
) PUSH↓
) PUSH↓
WRITE NOW THE CURRENT LEVEL IS [!LEVEL]↓
NOW THE CURRENT LEVEL IS 2
)
```

First, evaluate [!LEVEL] and write the current environment level number. Then, after pushing two levels, evaluate and write [!LEVEL] again.

LFE *Utility*

Format

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

Purpose

Call the Library File Editor and update library files.

LFE edits and analyzes library files, that are sets of object files with special starting and ending blocks. The extension .LB usually designates library files. See the *AOS Library File Editor User's Manual* or the *AOS/VS Link and Library File Editor (LFE) User's Manual* for further information.

Format

XEQ LINEDIT *[pathname]*

Purpose

Edit ASCII text (AOS only).

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 a pathname and that file exists, LINEDIT opens it for editing and displays the prompt.

If your user profile allows you to create two or more subordinate processes, you can execute CLI commands from LINEDIT. See the *AOS LINEDIT Text Editor User's Manual*, for more information.

LINK *Utility*

Format

XEQ LINK objectmodule *[argument]...*

Purpose

Link object modules to form an executable program file.

The Link utility creates executable program files from one or more object and library files.

In the command line, the optional argument can be any of the following:

- An object file created by an assembler or compiler
- A library file created by the LFE utility
- The literals "!*", "!", "*!", which begin, divide, and close an overlay area
- A symbol name modified by special switches

For more information about the Link utility, see either the *Advanced Operating System/Virtual Storage (AOS/VS) Link and Library File Editor (LFE) User's Manual* or the *Advanced Operating System (AOS) Link User's Manual.*

LINK Switches

/E=pathname,/L,/L=pathname
See CLI Commands page.

/ALPHA
List all symbols, sorted alphabetically, and write the list to the list file. /L must accompany this switch.

/BUILDSYS
Build an AOS program file.

/CHANNELS=integer
Create entry symbol ?CHAN having value integer; if /SYS=RDOS, define the maximum number of channels the program will open at one time.

/DEBUG
Create the undefined external symbol DEBUG, and optionally generate .DL or .DS output files.

/HEX
Convert octal output values to hexadecimal.

/KTOP=integer
Set the output program file's address space to integer number of 1K pages.

/MAP
Write a map of all partitions, common areas, and overlays to the list file. /L must accompany this switch.

/MODMAP
Produce a listing of each module's contributions to all partitions. /L must accompany this switch.

/MODSYM
Produce a listing of each module's entry symbols. /L must accompany this switch.

/MTOP=integer
(AOS/VS only) Set the output program file's address space to integer number of megabytes.

/N
Do not create any output files, except the error and listing files.

/NBOT=integer
Set the lowest NREL address to integer.

/NRP
Optimize some resource calls.

/NSLS
Suppress the scan of the system library, which by default occurs at the end of the first pass.

/NTOP=integer
Set the highest NREL address to integer.

/NUMERIC
List all symbols, sorted numerically, and write the list to the list file. /L must accompany this switch.

/O=pathname
Give all output files, except the listing and error files, the name pathname plus the appropriate extension.

/OBPRINT
Dump the contents of each object block. /L must accompany this switch.

/OVER
Suppress the overwrite error message when overwriting occurs.

/PRSYM
Place an RDOS-style radix-50 symbol table in the program file.

/REV=number
Set the revision number of the program file to the specified value. Under AOS/VS, the format for number is ww[.xx[.yy[.zz]]]. Under AOS, the format is ww[.xx].

/RING=integer
(AOS/VS only) Set the ring of the program file to an integer in the range 0 through 7.

/SRES=integer
Reserve integer number of reserved pages for ?SPAGE I/O.

/STACK=integer
Set the default stack size to integer number of words.

/SUPST
Do not generate an output .ST (symbol table) file.

/SYS=string
Specify the operating system for which the program file should be built. Under AOS and AOS/VS, the following are legal values for string: "RTOS", "RDOS", and "AOS". In addition, the following are legal only under AOS/VS: "VS16" and "VS32".

/TASKS=integer
Set the maximum number of tasks that the program file needs to execute to an integer.

/TEMP=pathname
Place Link's temporary files in pathname.

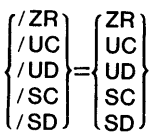
/UDF
Build a user data file (UDF). UDFs are not executable: they lack system tables and system library modules.

/ULAST
=partition-name
Place the partition identified by partition-name in the highest unshared portion of the program file, just before the default stack.

/V
List the input file pathnames on the listing.

LINK (continued)

/ZBOT=integer
Set the lowest ZREL address to integer.



Append the contents of the default partition on the left of the equal sign to the default partition on the right. Switches are mnemonic: Zero-Relocatable, Unshared Code, Unshared Data, Shared Code, and Shared Data.

Link object modules to form an executable program file.

Argument Switches

/ALIGN=integer
Align the contents of this partition on a boundary of 2 raised to the power of integer, where integer is a decimal value in the range 1 through 10. You may append this switch to a left overlay delimiter (!*), to a labeled common symbol, or to a normal partition.

/LOCAL
Add this module's local symbols to the symbol table. You may append this switch to an object or library filename.

/MAIN
Create entry symbol .MAIN, whose value is the starting address of this module. You may append this switch to an object or library filename.

/MULT=integer
Give the overlay area integer number of "basic" areas. You may append this switch to the left overlay delimiter (!*)

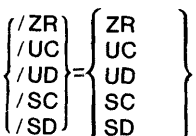
/OVER
Suppress the overwrite error message for the duration of this module. You may append this switch to a left overlay delimiter (!*), or to an object or library filename.

/SHARED
Place this partition or common area in the shared area. You may append this switch to a symbol name.

/START
Use this module's start address as the program's start address. You may append this switch to an object or library filename.

/VAL=integer
Create this accumulating symbol and initialize it to integer. You may append this switch to a symbol name.

/VIRTUAL
Create an RDOS virtual overlay. You may append this switch to a left overlay delimiter (!*)



Append the contents of the default partition on the left of the equal sign to the default partition on the right, for the duration of this module.

Examples

```
) XEQ LINK/L=@LPT/OVER/ULAST=&
&)UC/NSLS/O=MY_PROG ALPHA BETA
```

Link two modules, ALPHA and BETA, and name the output files MY_PROG.extension. /L=@LPT generates a listing and sends it to the lineprinter. /OVER suppresses overwrite messages, and /ULAST=UC places the contents of partition UC last in the unshared area of memory. /NSLS suppresses the search of the system library.

```
) XEQ LINK ALPHA BOOT/VAL=2 BETA !* GAMMA&
&)DELTA ! RHO *!
```

Bind five modules: ALPHA, BETA, GAMMA, DELTA, and RHO. The overlay designators specify an overlay area with two overlays, one consisting of GAMMA and DELTA, the other consisting of RHO. BOOT/VAL=2 creates the accumulating symbol BOOT and initializes its value to 2.

Format

LISTFILE *[pathname]*

Purpose

Set or display the current LISTFILE.

If the file specified by pathname doesn't exist, LISTFILE creates it. If that file does exist, LISTFILE appends the subsequent data to LISTFILE. The LISTFILE pathname is passed to any process created by an EXECUTE, XEQ, or DEBUG command.

Command Switches

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

/K
Set list file to null (no arguments allowed).

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

/P
Set list file to the previous environment's list file (no arguments allowed).

Argument Switches

None.

LISTFILE (continued)

Examples

```
) LISTFILE↵
@LIST
) LISTFILE :UDD:PHIL:LIST↵
)
```

First, display current list file which is the generic @LIST file. Then, set list file to the file LIST.

Format

[!LISTFILE]

Purpose

Expand to the pathname of the current LISTFILE.

This pseudo-macro doesn't accept arguments.

Macroname Switches

/P
Expand to the previous environment's list file pathname.

Argument Switches

None.

!LISTFILE (continued)

Examples

```
) WRITE THE CURRENT LIST FILE IS [!LISTFILE]↓
THE CURRENT LIST FILE IS @LIST
) PUSH↓
) LISTFILE :UDD:USER:WORK↓
) WRITE NOW THE CURRENT LIST FILE IS [!LISTFILE]↓
Expand to the pathname of the current LISTFILE. NOW
THE CURRENT LIST FILE IS :UDD:USER:WORK
) WRITE [!LISTFILE/P]↓
@LIST
)
```

First, evaluate [!LISTFILE] and write the current list file, which is the generic @LIST. Then change environment, and set a new list file for the new environment. Evaluate and write [!LISTFILE] for the current environment, and then, using the /P switch, for the previous environment.

Format

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

Purpose

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

Unless you include the /FLAT switch, LOAD will maintain dumpfile's directory tree structure in the working directory. Load one or more previously dumped files into the working directory. You may use templates for the source-pathname argument(s). If you supply no source-pathname arguments, the default is the template #.

Command Switches

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

/DELETE
Delete any existing nondirectory file with the same name as a file in the dump file and load the file from the new dump file.

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

/NACL
Give loaded files the default ACL.

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

LOAD (continued)

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

/DENSITY=mode
Control the magnetic density of your load tape. Use this command with MTB, model 6026 tape drives only. The following are your mode options:

Mode	Density
800	800 BPI
1600	1600 BPI
ADM	Automatic Density Matching

/V
Verify each loaded file on @OUTPUT. When used with /DELETE or /RECENT, the /V switch also verifies each deletion.

/BEFORE/TLM=
Load only those files that were last modified before the specified time, date, or date:time.

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

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

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

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

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

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

Specify a range of dates by using both the /BEFORE or /AFTER switches. Use the same modifier for both: either /TLM= or /TLA=, but not /TLM= and /TLA= at the same time.

/BUFFERSIZE=bytes
The maximum blocksize of the tape is bytes.

/TYPE=type
Select all files of the specified type. Types are provided in Table 2-3. Type can be in the form:

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

You can use more than one /TYPE= switch in a command line; for example:

.M A1@E1
W0

LOAD (continued)

/TYPE=64-68/TYPE=\66

selects types 64, 65, 67, and 68.

Argument Switches

None.

Examples

```
) LOAD/D/V @MTA0:0↵
DELETED MYFILE.SR_
MYFILE.SR
MYFILE.PR
ZFILE.CLI
DELETED OBS.SR
OBS.SR
)
```

Load all files that are contained in the first file of the magnetic tape mounted on unit 0 into the working directory and list their names on the terminal. If a file in the working directory has the same name as a file in the dump file, delete the file in the working directory and load the file from the dump file. Also, verify all such deletions.

```
) LOAD/RECENT @MTA0:1 +.SR↵
)
```

Load into the working directory each file from @MTA0:1 that ends in the extension .SR and is newer than any file with the same name.

Format

LOGEVENT message

Purpose

Enter a message in the SYSLOG (system log) file.

LOGEVENT enters the message you specify into the system log file, SYSLOG. You must be in SUPERUSER mode (SUPERUSER on) to execute LOGEVENT.

Command Switches

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

Argument Switches

None.

Example

```
)SUPERUSER ON)

*)
*)LOGEVENT BOOTED NEW SYSTEM)
*)
```

This sends the message BOOTED NEW SYSTEM to the system log event file. (We turned SUPERUSER on before issuing the command.) Note that the REPORT utility will display only the first 55 characters of the message.

LOGFILE *Command*

Format

LOGFILE *[pathname]*

Purpose

Set or display the current log file.

If the file you specify in *pathname* does not exist, LOGFILE creates it. If it already exists, the CLI appends subsequent data to it.

Command Switches

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

/K
Set the log file to null (no arguments allowed).

/P
Set current log file to the previous environment's log file.

/V
Verify the log file to @ OUTPUT.

Argument Switches

None.

Examples

```
) LOGFILE↵
) LOGFILE/V FILE.LOG↵
:UDD:JAMES:FILE.LOG
)
```

First display the current log file, which is null, and then set the log file to :UDD:JAMES:FILE.LOG.

!LOGON *Pseudo-Macro*

Format

[!LOGON]

Purpose

Determine if you are logged on under the EXEC and, if so, expand to CONSOLE or BATCH.

!LOGON does not accept arguments. If you're not sure whether you're logged on under the EXEC, enter this pseudo-macro. It will return CONSOLE or BATCH (if you're logged on in a batch stream) or nothing at all (if you aren't under the EXEC).

!LOGON is most useful in macros.

Macroname Switches

None.

Argument Switches

None.

Example

Assume that you want a macro to be interactive unless it is in a batch job. You could use the example below in the macro to find out if the process is logged on in a batch stream.

```
.
.
.
:[!EQUAL,[!LOGON],BATCH]
```

Format

XEQ MASM source-pathname *[source-pathname]...*

Purpose

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

There are two primary macroassembler (MASM) utilities: one for AOS, and one for AOS/VS. AOS MASM assembles one or more 16-bit (AOS) assembly language source files to produce an object file compatible with the AOS system. AOS/VS MASM assembles one or more 32-bit (AOS/VS) source files to produce an object file compatible with AOS/VS. Output from both utilities can consist of object files, listing files, or both.

To assemble object files for your system, use the appropriate macroassembler. For more information about the MASM utilities and their switches, consult the *AOS Macroassembler (MASM) Reference Manual* or the *AOS/VS Macroassembler (MASM) Reference Manual.*

To assemble 16-bit modules for use on AOS/VS, use the 16-bit MASM designed for AOS/VS, MASM16.PR.

MASM Switches (AOS only)

/E=pathname, /L, /L=pathname
See CLI Commands page.

/8
Use the first eight characters of symbols, not just the first five characters.

MASM (continued)

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

/E
Do not write Pass 2 error messages to @OUTPUT, unless there is no listing file. Some Pass 1 error messages are automatically written to @OUTPUT.

/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.

/K
Keep the assembler's symbol file table (MASM.ST) after the assembly is completed. By default, the symbol table is deleted.

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

/N
Do not produce an object file.

/O
Override all listing suppression controls.

/P
Add semipermanent symbols to the cross-reference.

/PS=pathname
Use pathname.PS as the .PS file for MASM, instead of MASM.PS

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

/S
Skip Pass 2, and save a version of the symbol table and macro definitions in MASM.PS

/U
Include user symbols in the object output.

Argument Switches (AOS only)

/S
Skip this file on Pass 2 of the assembly.

MASM Switches (AOS/VS only)

/E=pathname,/L,/L=pathname
See CLI Commands page.

/CPL=integer
Place integer characters on each line of the various output listings, where integer is in the range from 80 to 136.

/FF
Generate or suppress a form feed character as necessary to produce an even number of listing pages. /L must accompany this switch.

/LOCAL
Include user symbols in the object file.

/LPP=integer
Place integer lines on each page of the various output listings, where integer is in the range from 6 to 144. /L must accompany this switch.

MASM (continued)

/MAKEPS
Skip Pass 2, do not produce an object file, and save the temporary symbol table in MASM.PS

/MULT
Flag symbol redefinition statements as errors.

/N
Do not generate an object file.

/NOPS
Do not use a permanent symbol table to resolve symbols in the source module.

/O=pathname
Name the object file pathname.OB, rather than the name of the first source file.

/PS=pathname
Use pathname, rather than MASM.PS, as the permanent symbol table for the current assemble.

/STATISTICS
Include assembly statistics in the listing. /L must accompany this switch.

/SYMBOL=integer
Use the first integer characters to resolve symbols. /MAKEPS or /NOPS must accompany this switch.

/ULC
Distinguish between upper- and lowercase letters.

/XPAND
List all source lines, regardless of listing suppression directives appearing in the source. /L must accompany this switch.

/XREF= { NONE | USERSYMBOLS | ALL }

Include symbols of the specified type(s) in the cross-reference listing. /L must accompany this switch. By default, the macroassembler includes user symbols in the cross-reference.

Argument Switches (AOS/VS only)

/PASS1

Do not process this source file on assembly Pass 2.

Example

```
) XEQ MASM/L=LFILE MAINPROG SUBR1 SUBR2↵
```

Assemble three modules--MAINPROG, SUBR1, and SUBR2--and produce the object file MAINPROG.OB. Also, write a listing to the file named LFILE.

MASM16 *Utility*

Format

XEQ MASM16 pathname *[pathname]*

Purpose

Assemble one or more 16-bit source files on an AOS/VS system.

Use MASM16 on AOS/VS to assemble 16-bit source files. MASM16 runs under AOS/VS, but creates object files that you can use to produce program files for both AOS and AOS/VS. Note that MASM16 uses MASM16.PS, not MASM.PS. Refer to the *AOS Macroassembler (MASM) Reference Manual* for a complete description of 16-bit MASM and its switches.

Format

MESSAGE errorcode *[errorcode]...*

Purpose

Display text message corresponding to error code arguments.

MESSAGE looks up the text for error codes in the error message file and displays it on @OUTPUT (if you do not use the /L switch) or on @LIST (if you use the /L switch).

Command Switches

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

/D
Accept arguments as decimal integers; default is octal.

Argument Switches

None.

Example

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

Display the error message for error code 25.

MKABS (AOS only) *Utility*

Format

XEQ MKABS filename1 filename2

Purpose

Convert an RDOS save file to an absolute binary file (AOS only).

This utility runs on both AOS and AOS/VS systems and produces an absolute binary file which will run on a stand-alone system. Filename1 is a LINK save file that you want to input to MKABS. Filename2 is a file you designate to receive output. MKABS outputs filename1 to filename2 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

/FROM=N
Set the first save file address that 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_8 for RDOS files) as the from address.

/LAST=N
Set the last save file address that MKABS will include in absolute file output equal to N. MKABS evaluates N as an octal number, relative to location 0. If you omit this switch, MKABS uses the final save file address as the last address of that file.

/START=N

Set the second word in the absolute binary file's 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 LINK 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_8 and that it was developed for RDOS.

Argument Switches

None.

Example

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

Copy 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 starting address 300_8.

MOUNT *Command*

Format

MOUNT linkname message

Purpose

Mount a tape.

linkname is a filename that you create for the device. message is the text displayed on the operator's terminal.

This command requests the operator to mount a reel of magnetic tape. After the operator mounts the tape, the system creates a link named linkname in your initial working directory. (MOUNT creates a link for both labeled and unlabeled tapes.) linkname contains a pathname to the actual device on which the volume was mounted. In subsequent commands, refer to the physical device by linkname.

After you enter this command, the system locks your terminal until the operator responds.

You need not use the CLI MOUNT command to explicitly mount a volume of labeled tape on a tape drive; reference the tape's filename and it will be mounted implicitly.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Command page.

/VOLID=volid
Signal that the MOUNT command refers to a labeled tape or set of labeled tapes. If you include this switch, the EXEC will instruct the operator to mount the specified volid and will create a link: *:UDD:username:linkname LNK @LMT:volid*

Include the VOLID= switch for each volume in the set in the order in which the system will use the volumes. The EXEC will automatically tell the operator to change volumes as they're needed and the system will verify that the operator has mounted the correct volume each time.")

/READONLY
Signal that all tapes mounted with this switch are to have ACLs set to username,RE. Request that the write ring be removed from all tapes mounted with this switch.

Argument Switches

None.

Examples

```
) MOUNT TAPE1 PLEASE MOUNT TAPE NO Z280↵
) DUMP TAPE1:0 +.PR↵
)
```

First, request the operator to mount a tape numbered Z280 and direct the system to create a link for it named TAPE1. Then dump all of the files with the extension .PR in the working directory to TAPE1.

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

Have EXEC create the link named MYFILE in directory :UDD:USERNAME. This link resolves to @LMT:FIRST when you issue an appropriate system call or CLI command. You may substitute MYFILE for @LMT:FIRST in the pathname you provide.

```
) MOUNT/READONLY MYFILE HI DENSITY PLEASE↵
```

Request the operator to mount a tape without a write ring. The tape is therefore read-only.

Format

MOVE dest-dir*[sourcefile]...*

Purpose

MOVE moves copies of one or more files to the destination-directory, dest-dir. Normally, all source files must be inferior to the working directory. Use templates for the sourcefile argument(s). If you supply no sourcefile, the template # is assumed. Use filename templates for sourcefiles but not for the dest-dir.

Command Switches

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

/BUFFERSIZE=n
Read the sourcefile(s) into a buffer of size n bytes.

/DELETE
Delete the nondirectory file in the dest-dir if it has the same filename as a source file.

/FLAT
Do not maintain tree structure, move all source files into dest-dir (see Examples).

/NACL
Give the new files the default ACL. See the DEFACL command for a discussion of default ACLs.

/RECENT
If there is a nondirectory file in dest-dir with the same filename as a source file, move that source file only if it is more recent than the existing file.

MOVE (continued)

/V
List the name of each file the system moves on @OUTPUT. When used with /DELETE or /RECENT, the /V switch also verifies each deletion.

/BEFORE/TLM=
Move only those files that were last modified before the specified time, date, or date:time.

/BEFORE/TLM=time
Move only those files that were last modified today before time. Time is in the format hh:mm:ss.

/BEFORE/TLM=date
Move only those files that were last modified before the specified date. date is in the format dd:mmm:yy

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

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

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

/AFTER/TLA=

Move only those files that were last accessed after the specified time, date, or date:time. See /BEFORE/TLM=date:time for format.

Specify a range of dates by using both the /BEFORE and /AFTER switches. Use the same modifier for both: either /TLM= or /TLA= but not /TLM= and /TLA= at the same time.

/TYPE=type

Select all files of the specified type. Types are provided in Table 2-3. Type can be in the form:

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

You can use more than one /TYPE= switch in a command line; for example:

/TYPE=64-68/TYPE=\66

selects types 64, 65, 67, and 68.

Argument Switches

None.

MOVE (continued)

Examples

```
) MOVE ↑DIR1↓
)
```

Move the entire subtree inferior to the working directory to DIR1 which is in the immediately superior directory. Note that since you specified no source files, the CLI assumes #.

```
) MOVE/FLAT ↑DIR2↓
)
```

Move each file that is inferior to the working directory into DIR2. Note that the system doesn't maintain the tree structure; instead, it lists each file directly in DIR2.

```
) MOVE/TYPE=64-68/TYPE=\66 UDD:BOB:DIR3 +↓
)
```

Format

XEQ MPL inputfile-pathname

Purpose

Invoke the macro processor for procedural languages (MPL).

MPL defines and references macros for use in programs written in high-level languages. It processes macros in a file that contains source code for the target language. Target languages include Data General's PL/I, FORTRAN 5, and DG/L languages. MPL can be used to process program text in almost any high-level procedural language.

Execute MPL before compiling a program to translate MPL code (source code including macro definitions and references) into source code acceptable to the compiler.

MPL first looks for the input file named **inputfile-pathname.MPL**. If the search fails, it looks for **inputfile-pathname**. An output file has the appropriate language extension: .DG (for DG/L), .PL1 (for PL/I), and .FR (for FORTRAN 5). Specify an output file with the /O switch (MPL adds the language extension if you omit it). Otherwise, MPL takes the name of the input file and adds the proper extension for the output file.

When MPL executes, it reads a file to determine which characters need special interpretation. Specify the proper file with the /LANGUAGE=language-file switch. If you omit the switch, MPL uses the default file, MPL.LANGUAGE. Some language files come with MPL: DGL.LANGUAGE, PL1.LANGUAGE, or F5.LANGUAGE.

MPL (continued)

Abbreviate the switches in the command line, as long as the abbreviation remains unique. For a complete discussion of this utility, see the *Macro Processor for Procedural Languages (MPL) User's Manual.*

MPL Switches

/E=pathname
See CLI Commands page.

/ERRORLIMIT =integer
Terminate processing if the number of errors exceeds integer, where integer is in the range 0 to 32767.

/LANGUAGE =language-file
Use the special characters defined in language-file. For possible values, see the discussion above.

/N
Produce no output. /N overrides /O.

/O=pathname
Write output to the file specified by pathname.

Example

```
) XEQ MPL/LANGUAGE=PL1.LANGUAGE IN_FILE)
```

Invoke MPL to process the file named IN_FILE.MPL. MPL will use the PL/I language file, named PL1.LANGUAGE.

!NEQUAL *Pseudo-Macro*

Format

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

Purpose

Include input conditionally.

End the sequence with the !END pseudo-macro, and the sequence may optionally include the !ELSE pseudo-macro.

The !NEQUAL pseudo-macro must always have two arguments. It compares these character by character. If they don't match, !NEQUAL executes the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, !NEQUAL does not execute the input following it up to the !END.

If the arguments match, !NEQUAL does not execute the commands up to the !ELSE or !END. If there is an !ELSE, it executes the input following the !ELSE up to the !END.

Macroname Switches

None.

Argument Switches

None.

Example

This macro,

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

will write NOT AN ASTERISK if you call it with any argument except *; otherwise, it writes AN ASTERISK.

Note that you can also code the macro as follows:

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

Notice that we used commas to separate the arguments in the !NEQUAL pseudo-macro, for the same reason as in the !EQUAL pseudo-macro.

!OCTAL *Pseudo-Macro*

Format

[!OCTAL decimal-number]

Purpose

Convert a decimal number to octal.

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

Macroname Switches

None.

Argument Switches

None.

Example

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

Format

[!OPERATOR]

Purpose

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

This pseudo-macro does not accept arguments. The operator makes his presence or absence known by appropriate control commands to EXEC.

Macroname Switches

None.

Argument switches.

None.

!OPERATOR (continued)

Example:

Given a macro including:

```
[!EQUAL, [!OPERATOR],ON]
QBATCH FILE1 FILE2
[!ELSE]
WRITE TRY AGAIN LATER
[!END]
```

!OPERATOR is most useful when you want to change macro behavior based on the presence of an operator. In the example above, the CLI will queue up FILE1 and FILE2 if the operator is on duty; otherwise, it will output the message to try again later.

Format

PATHNAME filename

Purpose

Display a complete pathname starting at the root directory.

This command returns the full pathname beginning from the root to the specified file. 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 page.

Argument Switches

None.

Examples

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

Display the pathname of =, the working directory.

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

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.

Examples

```
) 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.milliseconds }

Purpose

Delay the CLI.

seconds is a number between 0 and 65535.
milliseconds is a number between 0 and 999.
Delay the CLI for the specified number of seconds.

Command Switches

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

Argument Switches

None.

Example

```
) PAUSE 8.50↓
)
```

Delay the CLI for 8.5 seconds.

PED *Utility*

Format

XEQ PED

Purpose

Display the process environment.

This command calls the Process Environment Display program, which displays runtime data on all system processes and terminals. It is a privileged utility, and is described further in the *AOS Operator's Guide* and the *AOS/VS Operator's Guide.*

Format

PERFORMANCE

Purpose

Display information about the CLI.

This command displays the following information about the CLI:

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

Command Switches:

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

PERFORMANCE (continued)

Argument Switches

None.

Example

```
) PERFORMANCE↵
153/489 SYSTEM CALLS
SHARED: 18 PAGES
UNSHARED: CURRENT 2 PAGES,
POSSIBLE 14 PAGES, HIGHEST 4 PAGES
5 STACK FAULTS
)
```

Format

PERMANENCE pathname [ON / OFF]

Purpose

Set or display a file's permanence attribute.

A permanent file cannot be deleted unless you turn its permanence off. With the **PERMANENCE** command, you can protect key files from accidental deletion. You can use templates in the pathname argument.

Command Switches

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

/V
Display the filename with its PERMANENCE attribute.

Argument Switches

None.

PERMANENCE (continued)

Examples

```
) PERMANENCE ZONIS↵
OFF
) PERMANENCE ZONIS ON↵
) PERMANENCE/V ZONIS↵
ZONIS      ON
)
```

First, display the PERMANENCE status of file ZONIS. Second, set PERMANENCE for the file to ON. Last, display and verify the current PERMANENCE setting for the file.

Format

[!PID]

Purpose

Expand to your CLI's process ID.

This pseudo-macro does not accept arguments. This pseudo-macro always returns a three-digit number.

Macroname Switches

None.

Argument Switches

None.

Example

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

PL1 *Utility*

Format

XEQ PL1 source-pathname

Purpose

Compile a PL/1 source file.

PL/1 is a high-level language based on ANSI standard PL/1. PL/1 source files have the extension .PL1. The compiler first searches for pathname.PL1, and then for pathname. The compiler produces an object file named source-pathname.OB.

Under AOS/VS, PL/1 accepts abbreviations for switch names, provided the abbreviation is unique. Under AOS, however, switch names cannot be abbreviated.

For detailed information on the PL/1 programming language, see *Plain PL/1 (A PL/1 Primer)*, the *PL/1 Reference Manual (AOS)*, and the *PL/1 Reference Manual (AOS/VS)*.

PL/1 Switches (AOS only)

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

/CODE
Print a generated code listing on the list file. /L must accompany this switch.

/DEBUG
Output symbol and line information for use by the SWAT debugger.

/ERRORCOUNT=integer
Terminate compilation after integer errors. If you do not include this switch, the default is 100 errors.

/LINEID
Generate code to keep track of source line numbers at execution time.

/N
Do not produce an object file.

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

/NOINCLUDES
Suppress listing of all %INCLUDE files.

/NOLEF
Do not generate LEF instructions.

/NOMAP
Suppress listing of storage map.

/NOPROCID
Do not save procedure names in the static data area.

/NOWARNINGS
Suppress severity 1 error messages.

/O=pathname
Write object file to **pathname.OB**, or to **pathname** if it already ends in .OB

/OPT [=1 or 2 or 3]
Set compiler optimization to the specified level, ranging from 1 (lowest) to 3 (highest). If you include this switch but do not specify a level, the default level is 3.

/PARAMS=pathname
Include parameter file **pathname** in the present compilation. The parameter file must have been previously created by compiling with the /SAVEPARAMS= switch.

PL1 (continued)

/SAVEPARAMS [=*pathname*]
Compile only %REPLACE statements, and save the result in pathname for later use with the /PARAMS= switch. If you do not specify a pathname, the compiler names the file source-pathname.PAR.

/STAT
Write compilation statistics to @OUTPUT.

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

/TMPDIR=string
Add string as a prefix to the beginning of all temporary filenames.

/XREF
Append a cross-reference table to the listing file. /L must accompany this switch.

PL/1 Switches (AOS/VS only)

/E=pathname,/L,/L=pathname
/CODE
Print a generated code listing on the list file. /CODE overrides /CODEMAP. /L must accompany this switch.

/CODEMAP
Print a code offset map. If /CODE is also specified, the compiler ignores /CODEMAP. /L must accompany this switch.

/DEBUG
Output symbol and line information for use by the SWAT™ debugger.

/DECLARES =pathname
Use pathname, created by a previous compilation with /SAVEDECLARES=, as the initial symbol table.

/ERRORCOUNT =integer
Terminate compilation after integer errors. If you do not include this switch, the default is 100 errors.

/INCLUDES
Do not suppress listing of %INCLUDE files. This is the default.

/LEF
Do not suppress generation of LEF instructions. This is the default.

/LINED
Generate code to keep track of source line numbers at execution time. This switch includes the function of /PROCID.

/MAP
Include a storage map in the listing. This is the default. /L must accompany this switch.

/MAPCASE
Translate all identifiers into uppercase before compilation.

/N
Do not produce an object file.

PL1 (continued)

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

/NOCODE
Do not print a generated code listing on the listing file. This is the default.

/NOCODEMAP
Do not print a code offset map. This is the default.

/NODEBUG
Do not output symbol and line information for the SWAT debugger. This is the default.

/NOINCLUDES
Suppress listing of all %INCLUDE files.

/NOLEF
Do not generate LEF instructions.

/NOLINEID
Do not generate code to keep track of source file line numbers at runtime. This is the default.

/NOMAP
Suppress listing of storage map.

/NOMAPCASE
Do not translate identifiers into uppercase. This is the default.

/NONESTLEVEL
Do not include block and group nesting level numbers in listing. This is the default.

/NOPROCID
Do not generate code to keep track of procedure names at runtime. This is the default.

/NOREPLACES
Do not flag lines containing %REPLACE names. This is the default.

/NOSTATISTICS
Do not print compilation statistics. This is the default.

/NOSUBCHECK
Do not generate code to check out-of-bound subscripts and arguments to SUBSTR at runtime. This is the default.

/NOWARNINGS
Suppress severity-1 error messages.

/NOXREF
Do not generate a symbol cross-reference table. This is the default.

/O=pathname
Write object file to pathname.OB, or to pathname if it already ends in .OB.

PL1 (continued)

/OPT [=1 or 2 or 3]
Set compiler optimization to the specified level, ranging from 1 (lowest) to 3 (highest). If you include this switch but do not specify a level, the default level is 3.

/PROCID
Save all procedure names at runtime. The default on-unit reports the procedure in which a runtime error occurs.

/REPLACES
Flag listing lines containing %REPLACE names.

/SAVEDECLARES =pathname
Save symbol table in pathname. The source file may contain only DECLAREs and %REPLACEs.

/STATISTICS
Write compilation statistics to the listing file and @OUTPUT.

/SUBCHECK
Compile code into program to check for out-of-bounds subscripts and arguments to SUBSTR.

/SYMLIB =pathname
Use pathname, created by a previous compilation with /SAVEDECLARES=, as a symbol "library".

/TMPDIR=string
Add string as a prefix to the beginning of all temporary filenames.

/WARNINGS
Do not suppress severity - 1 error messages. This is the default.

/XREF
Append a cross-reference table to the listing file. /L must accompany this switch.

Argument Switches

None.

Example

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

Compile the PL/I source file MYPROG.PL1. The /L=LFILE switch directs the compiler to output the listing to file LFILE.

PL1LINK *Utility*

Format

PL1LINK main-objectmodule *[subprogram-objectmodule]...*

Purpose

Link object modules to form an executable PL/I program.

PL1LINK is a macro that invokes the Link utility, to make PL/I object modules into an executable program. The PL1LINK macro will accept the switches of the Link utility. For a list of these switches, see the *AOS Link User's Manual* or the *AOS/VS Link and Library File Editor (LFE) User's Manual.*

Format

POP

Purpose

Return to the previous environment level.

POP restores the previous environment's settings of LEVEL, SUPERPROCESS, SUPERUSER, SCREENEDIT, SQUEEZE, CLASS1, CLASS2, VAR0 through VAR9, TRACE, LISTFILE, DATAFILE, LOGFILE, DIRECTORY, SEARCHLIST, DEFACL, STRING, PROMPT, and CHARACTERISTICS. It preserves none of the current settings. If your current LEVEL is 0, the POP command will cause a CLASS1 exceptional condition.

Command Switches

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

/V
Verify new level.

Argument Switches

None.

POP (continued)

Examples

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

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

Format:

XEQ PREDITOR

Purpose

Create and edit user profiles.

You must have the SUPERUSER privilege in order to use PREDITOR, though it does not need to be activated.

See your AOS or AOS/VS system manager's guide for more information about PREDITOR.

PREFIX *Command*

Format

PREFIX *[argument]...*

Purpose

Set or display the prefix string.

The prefix string is the prompt the CLI displays to indicate its readiness to receive input. This command lets you display the current prefix string or specify up to 24 characters to be output at the prompt. The initial prefix is the right parenthesis. The other special characters are output in addition to the prefix: for example, & for line continuation, and * for SUPERUSER.

Any character is valid as part of the prefix string. To enter certain special characters, you must use the !ASCII pseudo-macro, with the most significant bit set (that is, add 200 octal to the octal value of the character). These special characters are the control characters (octal values 0-37), parentheses (octal values 50 and 51), and angle brackets (octal values 74 and 76).

Command Switches

/1,/2,/L,/L=PATHNAME,/Q
See CLI Commands page.

/I
Set the prefix to the initial value ,that is, a right parenthesis (no arguments allowed)

Example

```
) PREFIX [!ASCII 207] HELLO!
HELLO PREFIX/I
)
```

First, set the prompt to ring a bell (the bell character is octal 7), and to write "HELLO". The second command line shows the new prefix string. PREFIX with the /I switch resets the prefix string to its initial state, a right parenthesis.

PREVIOUS *Command*

Format

PREVIOUS

Purpose

Display the previous environment's settings.

Command Switches:

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

Argument Switches

None.

Example

```
) LEVEL↵
LEVEL 1
) PREVIOUS↵
LEVEL                    0 OFF
SUPERUSER                OFF
SUPERPROCESS             OFF
SCREENEDIT               OFF
SQUEEZE                  ABORT
CLASS1                   WARNING
CLASS2
TRACE                    0  0  0  0  0
VARIABLES                0  0  0  0  0
                         @LIST
LISTFILE                 @DATA
DATAFILE
LOGFILE                  :UDD:JOHN
DIRECTORY                :PER,:UTIL,:
SEARCHLIST               JOHN,OWARE
DEFACL
STRING
PROMPT                   /605X/LPP=24/CPL=80
CHARACTERISTICS          /ON/ST/EB0/ULC/WRP
                         /OFF/SFF/EPI/8BT/SPO
                         /RAF/RAT/RAC/NAS
                         /OTT/EOL/UCO/LT
                         /FF/EB1/PM/NRM
                         /MOD/TO/TSP/PBN
                         /ESC/FKT/NNL

)
```

First display the current level. Then, display all the previous environment's settings.

PRIORITY *Command*

Format

PRIORITY { username:procname / process-ID } *[new-priority]*

Purpose

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

You cannot set the priority to a value higher than the CLI's priority unless you have the SUPERPROCESS privilege. If SUPERPROCESS mode is on, you can change the priority of any process, not just a subordinate process.

new-priority is a decimal number greater than or equal to the priority at which the process was created. If you do not input a new-priority, the CLI displays the priority of the existing process. If you do specify a new-priority, the CLI will change the selected process's priority to the new value.

If you input PRIORITY with no arguments, the system will display the CLI's priority.

Command Switches

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

Argument Switches

None.

Example

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

Set the process SON1's priority to 1.

PROCESS *Command*

Format

PROCESS pathname *[argument-to-new-process]*

Purpose

Create a process.

PROCESS creates a son process with the program specified by pathname. You select the new process's type, priority, and privileges via command switches. Note that if you use the simple /IOC switch or the /STRING switch, you must also select /BLOCK.

Arguments to the new process are placed in the initital IPC message to the new process. The new process can access the arguments through the ?GTMES system call.

The list file and data file passed to the son process (with the /LIST and /DATA switches) are the CLI's generic list file and data file. CLIs that are the son of EXEC and are running on terminals have no generic list and data files. CLIs that are the son of EXEC and are running in a batch stream have a generic list file created by EXEC and named **username.LIST.sequence-number**. They do not have a generic data file.

The CLI first tries to run **pathname.PR**. If that fails, it tries **pathname**.

NOTE: If you omit all privilege switches, the new process has no privileges. If you use /DEFAULT, then the new process has the same privileges as the creating process.

Command Switches

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

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

/BLOCK
Block this CLI until the new process terminates. If you omit this switch, the CLI does not block; it displays the new process's ID. The CHECKTERMS command can display the termination message from the created process when it terminates. You must use /BLOCK if you use the /IOC or /STRING switches.

/BREAK
(AOS/VS only) Create a break file if an error trap or fatal termination occurs; the default is no break file.

/BSON
Block the son process until explicitly unblocked.

/CALLS=number
Maximum number of concurrent system calls for the new process; default is the same as for the creating process.

/CHLOGICALTYPE
(AOS/VS only) Allow the new process to change its logical type (32-bit or 16-bit).

/CHPRIORITY
Allow the new process to change its priority.

/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.

PROCESS (continued)

/CHWSS
(AOS/VS only) Allow the new process to change its working set size.

/CONSOLE
Make the new process's console the same as the creating process's; The default is no terminal.

/CONSOLE=consolename
Make consolename the new process's terminal.

/CPU=s
Limit the CPU time for a new process, where s is a number of seconds between 0 and 4,294,967.

/DACL
Do not pass the default ACL to the new process.

/DATA
Make the new process's generic @DATA filename the same as the creating process's; default is no @DATA.

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

/DEBUG
Start the new process in the debugger.

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

/DIRECTORY
Make the new process's initial directory the creating process's initial directory; default is the creating process's working directory.

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

/INPUT
Make the new process's generic @INPUT filename the same as the creating process's; default is no @INPUT.

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

/IOC
Make the new process's @INPUT, @OUTPUT, and @CONSOLE generic filenames the same as the creating process's. You must use /BLOCK if you use the simple /IOC switch.

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

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

/LIST
Make the new process's generic @LIST filename the same as the creating process's. The default is no @LIST filename.

/LIST=pathname
pathname is the new process's generic @LIST filename.

/MEMORY=pages
Make pages the maximum memory size of the new process in 2K-byte pages. The default for AOS is the same as the creating process's. For AOS/VS, the default is minimum of top of shared or 512MB. For 16-bit programs running under AOS/VS, the default is minimum of top of shared or 64KB.

/NAME=name
Make name the simple process name for the new process. (If you omit this switch, the system assigns the name.)

PROCESS (continued)

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

/OUTPUT
Make the new process's generic @OUTPUT filename the same as the creating process's; the default is no @OUTPUT.

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

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

/PRIORITY=number
Make number the new process's priority; the default is the same as creating process's priority.

/PREEMPTIBLE
Make the new process pre-emptible.

/RESIDENT
Make the new process resident.

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

/SONS
A son process may create the same number of processes as the creating process, minus one; default is zero.

/SONS=number
Make number the maximum number of son processes that the new process can create.

/STRING
Store the program termination IPC message in the current STRING instead of displaying it. You must use /BLOCK with this switch.

/SUPERPROCESS
Allow the son process to enter SUPERPROCESS mode.

/SUPERUSER
Allow the son process to enter SUPERUSER mode.

/UNLIMITEDSONS
Allow the new process the option of creating an unlimited number of son processes.

/USERNAME=name
Make name the new process's username; the default is the same as the creating process's default.

/WSMAX=pagenum
(AOS/VS only) Specify maximum number of pages allowed in main memory at one time; the default is dynamically set by the system.

/WSMIN=pagenum
(AOS/VS only) Specify minimum number of pages that must be in main memory; the default is dynamically set by the system.

Argument Switches

Use any argument switches appropriate for the program specified in pathname.

Examples

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

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

```
) PROCESS/BLOCK/IOC LAMBDA 1↵
)
```

Create a swappable son process with the same generic @INPUT, @OUTPUT, and CONSOLE as this CLI, and block the CLI until this son terminates. The new process's program is LAMBDA, and it has access to the argument 1 via the ?GTMES system call.

Format

PROMPT *[command]...*

Purpose

Set or display the current prompt setting.

PROMPT displays the prompt or specifies up to 8 CLI commands that the system will execute before it issues the prompt. When setting a PROMPT argument, you must enter only the CLI command name (no associated arguments or switches -- see Examples). You can only use commands that do not require an argument.

Command Switches

/1,/2./L,/L=pathname,/Q
See CLI Command page.

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

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

Argument Switches

None.

PROMPT (continued)

Examples

```
) PROMPT TIME DATE DIRECTORY↵
9:32:16
26-NOV-80
:UDD:USER:PHIL
) PROMPT↵
TIME
DATE
DIRECTORY
9:32:18
26-NOV-80
:UDD:USER:PHIL
) PROMPT/K↵
)
```

First, set PROMPT to the CLI commands that you want the system to execute before it issues the prompt character. (Note that there are no optional arguments or switches.) Then, display PROMPT; then set the PROMPT to null.

Format:

PRTYPE { username:procname / process-ID } [*PREEMPTIBLE* / *RESIDENT* / *SWAPPABLE*]

Purpose

Set or display the type of an inferior process.

You can't change your process type unless you have the privilege ?PVTY, or SUPERPROCESS mode is on. If SUPERPROCESS mode is on, you may change the type of any process, not just an inferior process.

Command Switches

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

Argument Switches

None.

PRTYPE (continued)

Example

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

First, create a pre-emptible son process with program PROGA. (The system assigns the new process a 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 may now change the environment settings LEVEL, SUPERPROCESS, SUPERUSER, SQUEEZE, CLASS1, CLASS2, TRACE, VAR0 through VAR9, LISTFILE, DATAFILE, LOGFILE, SCREENEDIT, DIRECTORY, SEARCHLIST, DEFACL, STRING, PROMPT, and CHARACTERISTICS, using the appropriate CLI commands.

Command Switches

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

/V
Display the new environment's level.

Argument Switches

None.

PUSH (continued)

Examples

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

First, display the current environment's level; then PUSH a level (thereby saving the current environment) and display the new level setting.

Format

QBATCH argument *[argument]...*

Purpose

Create and submit a batch job file.

The CLI creates a batch job file in your working directory and places an entry for it on the batch queue. The batch job file begins with commands that set the batch job's working directory, search list, and default ACL to their current setting. If you issue the QBATCH command without /I or /M, the remainder of the command line becomes the batch job. The EXEC utility deletes the job file after the job runs.

Command Switches

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

/I
Take the contents of the file from subsequent lines of the @INPUT file. You must terminate the input mode with a single right parenthesis), and a NEW LINE. (No arguments allowed).

/M
Take the contents of the file from subsequent lines of the current macro body. The last line of the macro file must contain a single right parenthesis). (No arguments allowed.)

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

QBATCH (continued)

/V
Display the name of the batch job file.

/AFTER=date:time
Date:time is in the form **dd-mmm-yy:hh:mm:ss**. Process this request after **date** and **time**. Note that the /AFTER switch effectively guarantees that the request will not be processed before a certain time. The request will remain in the queue while the /AFTER switch is in effect and will gain priority by virtue of its age. You can 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
Limit CPU time for batch jobs, where time specifies the maximum amount of CPU time that the request can use. This switch accepts time in the form **hh:mm:ss**, where minutes and seconds are optional. You must allow enough time for all processes created in the batch job. If you omit this switch, EXEC will assume one minute of CPU time.

This switch is acted on only if the operator has set a time limit for jobs in the stream; otherwise, the switch is ignored. If the limit is on, and the time specified by the switch exceeds the limit, EXEC rejects the command.

/DESTINATION=string
Print string in block letters at the top of any header or trailer pages. pages. The default destination string is **username**.

/HOLD
Hold the entry until you explicitly release it with the QUNHOLD command.

/JOBNAME=name
Name the entry name. You can use name to QHOLD, QUNHOLD, or QCANCEL the job. The jobname must contain at least one alphabetic character. The default jobname is null.

/NORESTART
If the system fails while processing this entry do not restart the job.

/NOTIFY
Cause EXEC to send a mesage back to your terminal when the queue request is completed.

/OPERATOR
Do not run this job if no operator is present. You should use this switch when submitting a batch job containing a MOUNT request.

/QLIST=pathname
Set the generic list file of the batch process to pathname. pathname may not be a queue name.

/QOUTPUT=pathname
Set the generic output file of the batch process to pathname. The pathname should not be a queue name.

/QPRIORITY=n
Give this job priority n ($0 < n < 255$). n cannot be less than the job priority specified in your user profile.

Argument Switches

Use any argument switches appropriate for batch job specified in argument.

QBATCH (continued)

Examples

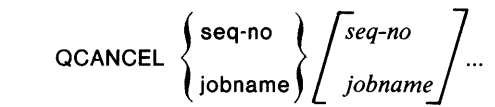
```
) QBATCH XEQ MASM FILE3↓
QUEUED, SEQ=65 QPRI=128
)

) QBATCH/I↓
))XEQ MASM FILE3↓
))XEQ BIND FILE3↓
))XEQ FILE3↓
)))↓
QUEUED, SEQ=66, QPRI=128
)

) QBATCH/V XEQ MASM FILE3↓
:UDD:CLJ: ?009.CLI.001.JOB
QUEUED, SEQ=67, QPRI=128
)
```

The 009 indicates the process ID of the issuing CLI. The 001 is included to make the filename unique; i.e., the next batch command you execute may generate file ?009.CLI.002.JOB.

Format



Purpose

Cancel a queue entry.

QCANCEL removes the specified entry from the queue to which it was submitted. Use the QDISPLAY command to find the seq-no that EXEC assigned to your entry. If your user process runs under EXEC, you can cancel only your own entries in the queue. You can cancel all jobs in a queue with a given jobname with one command by specifying the jobname. If the jobname is null, enter two commas as the argument. This will cancel all jobs with your username and a null jobname.

Command Switches

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

Argument Switches

None.

QCANCEL (continued)

Example

```
)QCANCEL JOB1)
)
```

Removes the entry for JOB1 from the BATCH_INPUT queue.

```
)QCANCEL ,,)
)
```

This cancels all jobs which you queued up that have a null jobname.

Format

QDISPLAY *[hostname]*

Purpose

Display queue information.

QDISPLAY displays the name and type of each queue maintained by the operating system. If you are allowed to place entries in a queue, this command will display the word OPEN with the queue name and type. If you do not provide an argument, QDISPLAY will display information from the local queues. Otherwise, the QDISPLAY will display information about the specified remote queues.

If you omit switches, QDISPLAY lists all queue names and their entries. Entries preceded by an asterisk are currently being processed; other entries are preceded by a letter that indicates status.

Status Letters

A Unexpired /AFTER switch in effect
B /BINARY switch in effect
C Cancelled by user (QCANCEL command)
D User /DELETE switch in effect
E Held by operator
F Cancelled by operator

QDISPLAY (continued)

G User /NORESTART switch in effect
H Held by user
N /NOTIFY switch in effect
O /OPERATOR switch in effect
R Restarted
S Queued by SUPERUSER

Command Switches

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

/QUEUE=queuename
Display only the queue name. This switch may appear more than once in a command.

Permanent queuenames are:

BATCH_INPUT
BATCH_OUTPUT
BATCH_LIST

Check with your operator for local queue names.

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

/TYPE=type
Display queues of type only. This switch may appear more than once. Queue types are

BATCH
PRINT
PUNCH (AOS only)
PLOT
HAMLET
RJE80
FTA

/V
Display appropriate column headings and more complete information for each queue that has entries and that is to be displayed. This switch has no effect if /SUMMARY is also present.

Argument Switches

None.

Examples

```
) QDISPLAY↓
```

Display information about all queues.

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

Display information about the batch and print queues.

```
) QDISPLAY/QUEUE=BATCH_INPUT↓
```

Display information about the queue named BATCH_INPUT.

Format

QFTA/DESTINATION=pathname source-pathname

Purpose

Place an entry on the FTA queue.

QFTA queues the file named in **pathname** to the file transfer agent (FTA) queue. Do not delete or modify the file until the system transfers it.

The source pathname may be local or remote. If it is remote, it must be a complete pathname, beginning from :NET:hostname.

The /DESTINATION=pathname switch is required. The pathname must be complete, beginning from :NET:hostname.

Command Switches

/1,/2,/L,/L=PATHNAME,/Q
See CLI Commands page.

/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 pathname of the queued file.

/AFTER=date:time
date:time is in the form **dd-mmm-yy:hh:mm:ss**. Process this request after date and time. Note that the **/AFTER** switch effectively guarantees that the result will not be processed before a certain time. The quest will remain in the queue while the **/AFTER** switch is in effect and will gain priority by virtue of its age. 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.

/APPEND
Append the source file to the destination file if it already exists.

/CHECKPOINT
Restart the transfer from the most recent checkpoint if a network error occurs during the transfer.

/COMPRESS[=integer]
Use the data compression algorithm integer. If you use this switch but do not supply an integer, the default value is 0. If you do not use this switch, the system transfers the data without compression.

NOTE: Currently, only algorithm 0 is available.

/DDELETE
Delete the destination file (if it exists) before the transfer occurs.

/DESTINATION=pathname
This switch is required. The pathname may be remote or local, but it must be fully qualified.

/HOLD
Hold the entry until you explicitly release it with the QUNHOLD command.

/NORESTART
Do not restart the transfer if the system fails during the transfer.

QFTA (continued)

/NOTIFY
Tell the FTA to send a message to your terminal upon completion of the transfer.

/OPERATOR
Do not run this job unless an operator is present.

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

If you omit this switch, the system calculates the priority as follows:

$n = (m + 255)/2$

/QUEUE=queuename
Submit the job to queuename rather than to the default queue. The queue type must be FTA.

/RECENT
Process the request only if the source file is more recent than the destination file.

/RMODE
Use record mode transfer. The default is block mode.

/SDELETE
Delete the source file after the transfer completes.

/STREAM=integer
Submit the job to stream n, where integer is in the range 0 to 7. The default stream is 0.

Argument Switches

None.

Example

```
) QFTA/DESTINATION=:NET:HOSTA:UDD:USER:&
&)FILEX/V FILE1
:UDD:USER:FILE1 QUEUED, SEQ=32, QPRI=127
)
```

Submit FILE1 to the FTA queue, and display the name of the queued file. The destination file is remote, and must be fully qualified with the :NET directory name and the :HOSTA hostname.

QHOLD *Command*

Format

QHOLD { seq-no / jobname } [*seq-no* / *jobname*] ...

Purpose

Hold a queue entry.

QHOLD allows you to temporarily suspend a queue entry. You can unhold the entry with the QUNHOLD command. With QHOLD, you can hold only your own entries and those entries which are not active. After a job becomes active, you must use the QCANCEL command.

To hold all jobs in a queue with the same jobname, specify the jobname. If the jobname is null, then enter two consecutive commas as the argument. This will hold all jobs with your username and a null jobname.

Command Switches

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

Example

```
) QSUBMIT/QUEUE=BATCH_INPUT/JOBNAME=&
&) JOVIAL FILE1
.
.
.
) QHOLD JOVIAL
)
```

Hold jobname JOVIAL until a subsequent QUNHOLD command releases it.

QPLOT *Command*

Format

QPLOT pathname *[pathname]...*

Purpose

Place an entry on a plotter queue.

QPLOT queues an entry on a digital plotter queue. Do not delete or modify the file until the system outputs it. The system always plots the exact data in the file. EXEC does not record billing parameters.

You may use templates in the pathname argument(s).

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Command page.

/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 names of the queued files.

/AFTER=date:time
date:time is in the form dd-mmm-yy:hh:mm:ss. Process this request after date and time. Note that the /AFTER switch effectively guarantees that the request will not be processed before a certain time. The request will remain in the queue while the /AFTER switch is in effect and will gain priority by virtue of its age. 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. 1<n<25. The default is n= 1.

/DELETE
Delete the pathnames after plotting them.

/FORMS=type
Specify that special forms type must be used. Check with your operator for local forms types. The default is standard forms.

/HOLD
Hold the entry until you explicitly release it with the QUNHOLD command.

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

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

/OPERATOR
Do not run this job if no operator is present. Use this switch when submitting a job that requires special forms.

/QPRIORITY=n
Give the entry 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 (m).

If you omit this switch, the system calculates the priority as follows:

$n=(m+255)/2$

QPLOT (continued)

/QUEUE=queuename
Submit job to queuename instead of to the default queue. The queue type must be PLOT.

Argument Switches

None.

Examples

```
) QPLOT FILE1 FILE2↵
QUEUED, SEQ=651, QPRI=127
QUEUED, SEQ=652, QPRI=127
)
```

Queue FILE1 and FILE2 to a digital plotter output queue.

```
) QPLOT/DELETE FILE3↵
QUEUED, SEQ=653, QPRI=127
)
```

Plot FILE3 then delete FILE3 when it is complete.

```
) QPLOT/COPIES=3/TITLES FILE4↵
QUEUED, SEQ=654, QPRI=127
)
```

Plot three copies of FILE4 and produce titles on each page.

Format

QPRINT pathname *[pathname]...*

Purpose

Place an entry on the print queue.

QPRINT queues the file named in pathname to the line printer output queue, but does not itself print the file. Do not delete or modify the file until the system outputs it.

You may use templates in the pathname argument(s).

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Command page.

/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 names of the queued files.

/AFTER=date:time
date:time is in the form dd-mmm-yy:hh:mm:ss. Process this request after date and time. Note that the /AFTER switch effectively guarantees that the request will not be processed before a certain time. The request will remain in the queue while the /AFTER switch is in effect and will gain priority by virtue of its age. 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.

/BEGIN=n
Start printing the file at page n. The default is n=1.

/BINARY
Print in binary mode. This switch is valid only for devices which have binary mode enabled. Check with your operator for local binary devices.

/COPIES=n
Produce n copies of the file. The default is n=1.

/DELETE
Delete files after printing them.

/DESTINATION=string
Print string in block letters at the the top of any header or trailer pages. The default destination string is username.

/END=n
Stop printing the file at page n. The default is n=last page.

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

/FORMS=type
Print on special forms type. Check with your operator for local forms types. If you omit this switch, the system will use standard forms.

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

/NORESTART
Do not restart the listing if the system fails while it is printing this file.

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

/OPERATOR
Do not run this job unless an operator is present. Use this switch when submitting a job that requires special forms.

/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 you will print. If you omit this switch, EXEC estimates the page limit as follows:

pages = (bytes-in-file)/1000 + 4

If the operator-set page limit is on, and the value specified by this switch exceeds the limit, then EXEC rejects the command.

/QPRIORITY=n
Give the entry 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 (m).

If you omit this switch, the system calculates the priority as follows:

$n = (m+255)/2$

/QUEUE=queuename
Submit the job to queuename instead of to the default queue. The queue type must be PRINT.

/TITLES
Print each page with a title line, consisting of pathname, date and time last modified, and page number. By default, the system prints no titles.

QPRINT (continued)

Argument Switches

None.

Examples

```
) QPRINT FILE1 FILE2↵
QUEUED, SEQ=655, QPRI=127
QUEUED, SEQ=656, QPRI=127
)
```

Queue FILE1 and FILE2 to the line printer output queue.

```
) QPRINT/DELETE/FOLDLONGLINES FILE3↵
QUEUED, SEQ=657, QPRI=127
)
```

Print FILE3 folding long lines. Delete FILE3 when it finishes printing.

```
) QPRINT/COPIES=3/PAGES=75/TITLES FILE4↵
QUEUED, SEQ=658, QPRI=127
)
```

Print three copies of FILE4 and produce titles on each listing page. Each listing begins on page 1. The total number of pages to print will not exceed 75.

Format

QPUNCH pathname *[pathname]...*

Purpose

Place an entry on the punch queue (AOS only).

QPUNCH places the file named in pathname on a paper tape punch queue. Note that this command does not punch the file, but merely queues it to the punch; so don't delete or modify the file until the system outputs it.

You may use templates in the pathname argument(s).

Command Switches

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

/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 names of the queued files.

/AFTER=date:time
date:time is in the form dd-mmm-yy:hh:mm:ss. Process this request after the date and time. Note that the /AFTER switch effectively guarantees that the request will not be processed before a certain time. The request will remain in the queue while the /AFTER switch is in effect and will gain priority by virtue of its age. You may use a plus sign (+) to specify a relative time for process delay. For example,

QPUNCH (continued)

/AFTER=+12

says don't process until at least 12 hours have passed.

/COPIES=n
Produce n copies of the file. The default is n=1.

/DELETE
Delete files after punching them.

/FEET=n
n is the maximum number of feet of tape that you will punch. If you omit this switch, the system estimates the limit by the size of the file. If you specify /COPIES, use this switch.

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

/HOLD
Hold the entry until you explicitly release it with the QUNHOLD command.

/NORESTART
Do not restart the listing if the system fails while it is punching this file.

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

/OPERATOR
Do not run this job unless an operator is present. Use this switch when submitting a job that requires special tape.

/QPRIORITY=n
Give the entry 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 (m).

If you omit this switch, the system calculates the priority as follows:

$n = (m+255)/2$

/QUEUE=queuename
Submit the job to queuename instead of to the default queue. The queue type must be PUNCH.

Argument Switches

None.

Examples

```
) QPUNCH FILE1 FILE2↵
QUEUED, SEQ=659, QPRI=127
QUEUED, SEQ=660, QPRI=127
)
```

Queue FILE1 and FILE2 to the paper tape punch output queue.

```
) QPUNCH/DELETE FILE3↵
QUEUED, SEQ=661, QPRI=127
)
```

Punch FILE3, then delete FILE3 when complete.

```
) QPUNCH/COPIES=3/FEET=75 FILE4↵
QUEUED, SEQ=662, QPRI=127
)
```

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

QSUBMIT *Command*

Format

QSUBMIT pathname *[pathname]...*

Purpose

Place an entry on a batch or spool queue.

QSUBMIT queues an entry to the specified queue for each pathname you supply in the argument list. If you omit the /QUEUE= switch, QSUBMIT assumes the batch input queue because you normally use QPLOT, QPRINT, or switches for other options you can specify.

Do not use QSUBMIT to submit batch jobs in stacked format. Stacked format commonly applies to jobs submitted on punched cards.

You may use templates in the pathname argument(s).

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI Command page.

/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 names of the queued files.

/AFTER=date:time
date:time is in the form dd-mmm-yy:hh:mm:ss. Process this request after date and time. Note that the /AFTER switch effectively guarantees that the request will not be processed before a certain time. The request will remain in the queue while the /AFTER switch is in effect and will gain priority by

virtue of its age. 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.

/BINARY
Print in binary mode. This switch is valid only for devices which have binary mode enabled. Check with your operator for local binary devices.

/CPU=time
Limit CPU time for batch jobs, where time specifies the maximum amount of CPU time that the request can use. This switch accepts time in the form hh:mm:ss, where minutes and seconds are optional. Allow enough time for all processes created in the batch job. The default assumed by the system is one minute of CPU time.

This switch is acted on only if the operator has set a time limit for jobs in the stream; otherwise, the switch is ignored. If the limit is on, and the time specified by the switch exceeds the limit, EXEC rejects the command.

/DELETE
Delete the pathname(s) after processing.

/DESTINATION=string
Print string in block letters at the top of any header or trailer pages. The default destination string is username.

/HOLD
Hold the entry until you explicitly release it with the QUNHOLD command.

/JOBNAME=name
Batch queues only. Name the entry name. Then you can use name to QHOLD, QUNHOLD, or QCANCEL the job. (The jobname must contain at least one alphabetic character.) The default jobname is null.

QSUBMIT (continued)

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

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

/OPERATOR
Do not run this job if no operator is present. Batch jobs requiring the MOUNT feature should be submitted with this switch.

/QPRIORITY=n
Give the job priority n $1<n<255$. n cannot be less than the priority specified in your user profile (m).

If you omit this switch, the system calculates the priority as follows:

$n = (m+255)/2$

/QUEUE=queuename
Submit job to queuename. The default is BATCH_INPUT.

/QOUTPUT=pathname
Set the generic output file of the batch process to pathname. The pathname should not be a queue name.

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

/XW0=n
Place the value n in the ?XXW0 word of the ?EXEC system call packet. n must be a decimal number.

/XW1=n
Place the value n in the ?XXW1 word of the ?EXEC system call packet. n must be a decimal number.

/XW2=n
Place the value n in the ?XXW2 word of the ?EXEC system call packet. n must be a decimal number.

/XW3=n
Place the value n in the ?XXW3 word of the ?EXEC system call packet. n must be a decimal number.

Argument Switches

None.

Examples

```
) QSUBMIT FILE1 FILE2↵
QUEUED, SEQ=663, QPRI=127
QUEUED, SEQ=664, QPRI=127
)
```

Submit FILE1 and FILE2 to the BATCH_INPUT queue.

```
) QSUBMIT/NORESTART/HOLD FILE3↵
QUEUED, SEQ=665, QPRI=127
)
```

Submit FILE3 to the BATCH_INPUT queue. Hold it until a subsequent QUNHOLD command releases it. Do not restart job FILE3 if the system fails while it is processing the file. The system calculates the job's priority as described above.

```
) QSUBMIT/AFTER=12:30 BATCHJOB↵
QUEUED, SEQ=667, QPRI=127
)
```

The system will not process this job before 12:30.

QUNHOLD *Command*

Format

QUNHOLD { seq-no / jobname } [*seq-no* / *jobname*] ...

Purpose

Free a held queue entry.

QUNHOLD negates a previous QHOLD command on a queue entry. You cannot QUNHOLD an entry that the operator has on hold. If you QHOLD a BATCH_INPUT entry by jobname, QUNHOLD it by jobname; but you can always QUNHOLD any BATCH_INPUT entry by sequence number.

You can release all jobs in a queue with a null jobname by entering two consecutive commas as the argument to the QUNHOLD command. This releases all jobs with your username and a null jobname.

Command Switches

/1, /2, /L, /L=pathname, /Q
See CLI Command page.

Argument Switches

None.

Example

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

First hold, and then release, the batched job with the jobname MYJOB.

RDOS *Utility*

Format

XEQ RDOS command *[argument] ...*

Purpose

Read or write an RDOS dump file or disk.

RDOS is the Real-time Disk Operating System. The RDOS utility reads an RDOS dump file or disk to, or writes it from, your AOS or AOS/VS working directory. Use AOS or AOS/VS templates, but not RDOS templates, in the five RDOS utility commands: LOAD, DUMP, GET, PUT, and LIST.

The LOAD command loads an RDOS dump file on 9 track magnetic tape into your working directory. The DUMP command dumps files from your working directory onto a 9 track magnetic tape in RDOS dump format. The GET, PUT, and LIST commands assume that an RDOS disk is on your system. To execute these commands, you must have previously specified the disk drive as part of your system during AOSGEN or VSGEN. Do not try to initialize the RDOS disk.

LOAD

Format

XEQ RDOS LOAD rdos-dumpfile *[rdos-filename]...*

LOAD Switches

/D
Load the new file after deleting any file with the same filename.

/N
Do not load: just verify filenames.

DUMP

FORMAT

XEQ RDOS DUMP rdos-dumpfile *[aos-filename]... [templates]...*

DUMP Switches

None.

GET

Format

XEQ RDOS GET/disk=rdos-diskunit
[rdos-filename]...[templates]...

GET Switches

/N
Do not load; just verify filenames.

/T
Move the contents of all RDOS directories designated in the command line. Without this switch, the directories will be created, but no files will be placed in them.

RDOS (continued)

PUT

Format

XEQ RDOS PUT/DISK=rdos-diskunit
[/DIR=rdos-subdirectory]
[aos-pathname]...[templates]...

PUT Switches

/DIR=rdos-subdirectory
Put files into the specified RDOS subdirectory.

LIST

(Corresponds to the RDOS CLI LIST/E/A command.)

Format

XEQ RDOS LIST/DISK=rdos-diskunit *[RDOS-filename]...*
[template]

LIST Switches

/T
List the contents of all directories designated in the command line. Without this switch, LIST will list the specified directories but not the files that they contain.

RDOS Switches

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

/A
Abort on an ABORT condition.

/V
Verify each file transferred.

Argument Switches

/C
When used with LOAD and GET commands, convert carriage returns to NEW LINES. When used with DUMP and PUT commands, convert NEW LINES to carriage returns.

/N
Do not transfer files matching this template.

Examples

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

Load all the files ending in .SR and .RB from file 0 on MTA0 into the working directory. Convert all carriage returns in the source files (.SRs) to NEW LINEs; do not convert them in the relocatable binary files (.RBs).

```
) XEQ RDOS LOAD/V @MTA0:1 +.SV/N↵
```

Load all files on @MTA0:1 except files with an .SV extension.

Examples (continued)

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

Dump all files in the working directory to file 1 of MTA0. Convert all of the source files' NEW LINEs to carriage returns, and list their filenames on @OUTPUT. The dump file will be in RDOS format. All NEW LINEs will be converted, even those in .OB and .PR files.

```
) CREATE/DIR X↵
) XEQ RDOS GET/V/DISK=@DPD5 X:+.SR/C↵
```

Copy all files that have .SR extensions from RDOS subdirectory X of disk @DPD5. The directory X must exist in the working directory for this command to work.

```
) XEQ RDOS PUT/V/DISK=@DPD5/DIR=Y A.FR/C↵
```

Copy file A.FR from the AOS working directory to RDOS subdirectory Y on RDOS disk @DPD5 and convert NEW LINEs to CRs.

```
) XEQ RDOS LIST/DISK=@DPD5 Z:-.-↵
```

List all files matching the template -.- from RDOS subdirectory Z on RDOS disk @DPD5. Notice that the template -.- only matches files with extensions. The AOS and AOS/VS template -.- corresponds to the RDOS template -.*-.

Format

[!READ argument *[argument]...*]

Purpose

Display text on @OUTPUT and expand to argument from @INPUT.

Macroname Switches

None.

Argument Switches

None.

Example

Given a macro including

```
.
.
.
XEQ MASM [!READ WHICH FILE TO ASSEMBLE?]
.
.
.
```

The CLI writes the messages on the terminal and instructs the Macroassembler to assemble the filenames you type in response. The filename that you supply effectively replaces the pseudo-macro in the command line.

RELEASE *Command*

Format

RELEASE logical-disk *[logical-disk]...*

Purpose

Release a logical disk from the working directory.

RELEASE releases a previously initialized logical disk (LD). See INITIALIZE for more information about LDs.

Command Switches

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

Argument Switches

None.

Example

```
) RELEASE ALPHA
)
```

Release an LD named ALPHA that you previously initialized.

Format

RENAME pathname newname

Purpose

Change a file's name.

newname must be a simple filename.

Command Switches

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

Argument Switches

None.

Examples

```
) FILESTATUS↵
DIRECTORY :UDD:USER
FILEU CODEA
) RENAME FILEU FILEME↵
) FILESTATUS↵
DIRECTORY :UDD:USER
FILEME CODEA
)
```

REPORT *Utility*

Format

XEQ REPORT *[pathname] ...*

Purpose

Generate a report on the contents of the SYSLOG log file.

REPORT is described in the *AOS Operator's Guide* and the *AOS/VS Operator's Guide.*

Format

REVISION pathname *[field1[.field2[.field3[.field4]]]]*

Purpose

Set or display a program's revision number.

You may use templates in the pathname argument. *field1* and *field2* numbers can range from 0 to 255. You set the revision level with the .REV pseudo-op in an assembly-language source program. Note that files of type PRG have only two field numbers whereas files of type PRV have four field numbers.

Command Switches

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

/V
Display the filename with the revision number.

Argument Switches

None.

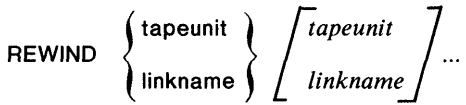
REVISION (continued)

Examples

```
) REVISION JOHN↵
00.00
) REVISION/V JOHN 0.1↵
JOHN 00.01
) REVISION TED↵
00.00.01.23
)
```

First, display the revision number of a program file named JOHN in the working directory, then change the revision and display the new one.

Format



Purpose

Rewind one or more tapes.

Specify either the same linkname you used to mount the volume, or the device on which it is mounted (tapeunit).

You may use templates in the tapeunit and linkname arguments.

Command Switches

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

Argument Switches

None.

REWIND (continued)

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 the required operations, rewind TAPE10.

Format

RIC source-pathname

Purpose

Compile an RPG II source file using the RPG II Interpretive Compiler (DG/RIC).

DG/RIC contains a debugger, analyzer, formatted dump, and high-speed compiler that produces interpretive code. Use the RPG II Interactive Editor to enter and correct the source program. Then use DG/RIC for a fast compile. Finally, when the program is debugged, compile the program using DG/ROC.

For a complete description of the RPG II programming language, see the *RPG II Reference Manual.* For additional discussion of DG/RIC, see the *Data General/RPG II Optimizing Compiler (DG/ROC) User's Manual (AOS and AOS/VS).*

RIC Switches

/C
Write the compiler-generated object code to the output file. The code follows the source listing (if one is specified) and the numbers of the lines for which DG/RIC generated specific code.

/CHECK
Compile the source program to a temporary file and delete the .IC, .DL, and .PR files. Use /CHECK with the /D/SOURCE/L=pathname switches for an error check and listing.

RIC (continued)

/D
Include debugger and analyzer information in the object file.

/E
Suppress both page ejection and new headers when the system encounters a comment specification having asterisks in columns 7 and 8. /E lets you run RPG II programs that use comments with lines of asterisks as separators.

/I
Inhibit conditioning indicator code optimization. Use for debugging, where repetitive patterns of conditioning indicators are replaced by a single test and branch.

/L=pathname
Write errors and source listing (if one is requested) to the file specified by pathname, instead of to @OUTPUT.

/N
Suppress printing of notes on the compiler listing. /N does not suppress warning, error, or fatal-error messages.

/O=pathname
Change the names of the output files to pathname.IC, pathname.PR, and pathname.DL.

/SOURCE
Write the source program to the default @OUTPUT or to the file specified in /L=pathname.

Argument Switches

None.

Examples

) RIC MYPROG↵

Compile MYPROG.RG, creating the object file MYPROG.IC. The system does not produce a source listing, but it sends error messages to @OUTPUT.

) RIC/D/O=NEWNAME MYPROG↵

Compile MYPROG.RG with debugger and analyzer information in the object file. The created files are named NEWNAME.IC, NEWNAME.PR, and NEWNAME.DL.

) RIC/SOURCE/L=PROG.LS MYPROG↵

Compile MYPROG.RG, and send a source listing and error messages to the output file PROG.LS.

ROC *Utility*

Format

ROC source-pathname

Purpose

Compile an RPG II source file using the RPG II Optimizing Compiler (DG/ROC).

DG/ROC produces machine instructions rather than interpretive code. DG/ROC programs run considerably faster than the same programs compiled with DG/RIC.

The ROC macro first searches for source-pathname.RG. If that is not found, it searches for source-pathname.

For a complete description of the RPG II programming language, see the *RPG II Reference Manual.* For more information on the ROC command line, see the *Data General/RPG II Optimizing Compiler (DG/ROC) User's Manual.*

ROC Switches

/L=pathname
See CLI Commands Page.

/BUFFERS=0
Suppress buffering.

/BUFFERS=integer
Multiply the number of buffers in the file description specifications by integer (SAM files only).

/BUFFERS=-1
Allocate as many buffers as will fit. The default is -1.

/CHECK
Execute a fast optimization check without generating optimized code.

/DSPLYRTN
Remove the suppression of DSPLY carriage return. Use this switch if DG/ROC may change the DSPLY operation, used in calculation specifications, to end with a carriage return.

/INTVAR
Store certain variables as hardware integers.

/L=pathname
Write the compiler output messages to the file specified by pathname.

/OPT=integer
Set optimization to the specified level:

0 Produce DG/RIC interpretive code.

1 Produce DG/ROC optimized machine code.

2 Increase optimization by removing redundant code.

3 Turn on all the compiler tuning switches (/DSPLYRTN, /INTVAR, and /OPT=2).

The default level is 1.

/P
Generate code for a formatted dump.

/SOURCE
Write the source program to the default @OUTPUT or to the file specified in /L=pathname.

/TMPDIR=prefix
Assign high volume compiler temporary files to the directory. Use the :TEMP: prefix to go to directory :TEMP.

ROC (continued)

/TRACE
Generate code that, at execution time, will output each statement number after it is executed. If the statement changes a variable, the variable name and contents are printed.

/TRACE= line#-line# [!line#-line#] ...
Generate code that, at execution time, will trace a specified range of lines. You may specify up to five ranges of lines. Separate each range by an exclamation point: /TRACE=25-50!75-100!125-150

/WARNOK
Ignore warnings and optimize the program.

Argument Switches

None.

Examples

```
) ROC MYPROG↵
```

Compile MYPROG.RG and produce optimized machine code.

```
) ROC/SOURCE/L=PROG.LS MYPROG↵
```

Compile MYPROG.RG, and send a source listing and error messages to the output file PROG.LS.

```
) ROC/P/TRACE/L=PROG.LS MYPROG↵
```

Compile MYPROG.RG, and send a listing to PROG.LS. Also, generate code that, at execution time, will write a trace to @OUTPUT and, in case of abnormal termination, will write a formatted dump to @OUTPUT.

RUNTIME *Command*

Format

RUNTIME [*username:procname* | *process-ID*] ...

Purpose

Display a process's runtime information.

RUNTIME displays the following runtime information about the specified process:

ELAPSED	Real-time elapsed since this process was created.
CPU	Central processor time used by this process.
I/O BLOCK	Number of blocks of data read or written by this process.
PAGE MSECS (AOS) PAGE SECS (AOS/VS)	Number of memory pages (2K bytes) used by this process, multiplied by CPU time used (in milliseconds for AOS, and in seconds for AOS).

If you include no argument, the CLI displays its own runtime information.

Command Switches

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

Argument Switches

None.

Example

```
) RUNTIME 7⤶
ELAPSED 21:44:09, CPU 0:01:47.008
I/O BLOCKS 927, PAGE MSECS 1728224
)
```

Note that, under AOS/VS, RUNTIME displays a PAGE SECS value rather than a PAGE MSECS value.

SCOM *Utility*

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 Command Switches). 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 the matchsize). The default matchsize is automatically set to four.

SCOM Switches

/L
Write the list of differences to current @LIST file. If you omit this switch, the program doesn't list the differences. Instead, it outputs the message FILES DIFFER STARTING AT LINE xxx/xxx if the files differ or a CLI prompt if the files match.

/L=filename
Write differences to filename.

/EOL
The end-of-line character is treated as significant. If you omit this switch, SCOM ignores EOL characters and blank lines.

/MS=number
Set matchsize to the specified value (number). If you omit this switch, the default is 4.

Argument Switches

None.

Example

```
) XEQ SCOM MYFILE YOURFILE
```

SCREENEDIT *Command*

Format

SCREENEDIT [ON / OFF]

Purpose

Set or display the current SCREENEDIT mode.

If SCREENEDIT is on (the default in interactive mode), you can modify input into your terminal by using cursor control characters. In batch jobs, the default is SCREENEDIT OFF. SCREENEDIT ON is only valid for display terminals.

Control Characters

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 to the right of the cursor.
CTRL-X	Move to the right one character. (The → key on the function keypad has the same effect.)
CTRL-Y	Move to the left one character. (The ← key on the function keypad has the same effect.)

Command Switches

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

/P
Set the current SCREENEDIT mode to the previous environment's SCREENEDIT mode (no arguments are allowed to this switch).

Argument Switches

None.

Example

```
) SCREENEDIT ON↓
) ANT CHARACTER STRING. (CTRL-H) (CTRL-X)&↓
&)(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 you've corrected the string by replacing the T with Y, it reads:

```
) ANY CHARACTER STRING.
```

SEARCHLIST *Command*

Format

SEARCHLIST *[pathname]...*

Purpose

Set or display the search-list setting.

If you use the /P command switch, the CLI displays the previous environment's search list; otherwise, the CLI displays the current search list.

Command Switches

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

/K
Delete the current search list, if any exists (no arguments allowed).

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

Argument Switches

None.

Examples

```
) SEARCHLIST↵
:PER,:UTIL,:
) PUSH↵
) SEARCHLIST      :UDD :HENRY,:PER,:UTIL,:↵
) SEARCHLIST↵
:UDD:HENRY,:PER,:UTIL,:
) SEARCHLIST/P↵
) SEARCHLIST↵
:PER,:UTIL,:
)
```

Display the current search list and then push a level. Change the current search list and display it. Then set the search list to its original value and display it again.

!SEARCHLIST *Pseudo-Macro*

Format

[!SEARCHLIST]

Purpose

Expand to the search list.

This pseudo-macro does not accept arguments.

Macroname Switches

/P
Expands to the previous environment's search list.

Argument Switches

None.

Example

```
) SEARCHLIST :UDD:MDIR,[!SEARCHLIST]
)
```

Evaluate the !SEARCHLIST pseudo-macro, then set the search list to the resulting argument string. The effect is the addition of :UDD:MDIR to the current search list.

Format

XEQ SED *[pathname]*

Purpose

Edit an ASCII text file.

SED calls the SED text editor program. If the file you specify in pathname does not exist, SED asks:

DO YOU WANT pathname TO BE CREATED?

Answer Y) to create the file. SED then displays its prompt (*).

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

See the AOS/VS *SED Text Editor User's Manual* for more information.

SED Switches

/ED=pathname
Store the .ED file, which contains formatting settings, in the directory specified by pathname.

/NO_ED
Do not create an .ED file.

/NO_SCREEN
Do not automatically update the screen. This switch is useful for a terminal with a low baud rate or for a hard-copy terminal.

SED (continued)

/PROFILE=pathname
Begin the editing session by executing the SED commands contained in pathname.

/WORK=pathname
Store all temporary SED files in the directory specified by pathname.

Argument Switches

None.

Example

```
) XEQ SED/ED=NEAT/WORK=:FIX_HEAD REPORT
```

Invoke SED to edit the file REPORT. Place the file containing formatting setting in the directory NEAT, and place all temporary files on the fixed-head disk.

Format

SEND { process-ID / username:procname / consolename } message

Purpose

Send a message to a terminal.

Use **SEND** to send a message to a process's terminal. The target process can be any process with a terminal. A **procname** can be either a simple or complete process. A complete **procname** is in the form

username:process

for example, BOOTHBY:CON7. **consolename** may begin with either :PER: or the @ prefix. 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 cannot send control characters.

If you send a message and the target process doesn't receive it, that process may have disabled message reception. You can use templates for the destination argument.

Command Switches

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

/I
Send each of the following input lines as a separate message until you reach a line containing a single).

SEND (continued)

/M
Send each line of the current macro file as a separate message. You must end the macro sequence with a single).

Argument Switches

None.

Examples

```
) SEND 2 PLEASE BRING UP LPT1↵
)
```

Send the message to the system operator.

```
) SEND @CON- SYSTEM WILL SHUT DOWN &↵
&)AT MIDNIGHT.↵
)
```

Send the 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.↵
)
```

Reply to a message received from process 8.

Format

[!SIZE pathname]

Purpose

Expand to a file's length in bytes.

!SIZE expands to the byte length of the file you specify (by its pathname). If the pathname does not exist, the pseudo-macro returns zero.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE THE LENGTH OF FOO IS [!SIZE FOO]
THE LENGTH OF FOO IS 14
)
```

SLB (AOS only) *Utility*

Format

XEQ SLB/O=shared-routine-name library-number
programname *[programname]...*

Purpose

Build a shared library (AOS only).

Call the Shared Library Builder utility to build a shared library routine from one or more executable programs. The SLB searches for programnames with the .PR extension, but you need not type the extension. The SLB always appends the extension .SL to shared-routine-name. library-number is a number in the range 2 through 63 that you want the new routine to have (numbers 0 and 1 are system reserved). See the *AOS Shared Library Builder User's Manual* for more on the SLB.

SLB Switches

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

Argument Switches

/O
This is the output filename; /O is a mandatory switch.

Example

```
) XEQ SLB/L=@LPT/O=GEOM5 SINE COSINE
TANGENT
```

Create shared library GEOM.SL from executable programs SINE.PR, COSINE.PR, and TANGENT.PR. Because GEOM.SL is a shared routine, many users have access to it from their own programs.

SORT/MERGE *Utility*

Format

{ SORT } { *[INTO outfile-pathname [FROM infile-pathname]...]* }
{ MERGE }

Purpose

Invoke the Sort/Merge utility.

This general-purpose utility manipulates record order and content. It sorts and copies records; merges multiple files into a single file; edits records in files; deletes duplicate records during a sort or merge operation; and deletes records you specify.

Use a command file to tell SORT/MERGE where to find the records to be sorted or merged, where to send the sorted or merged records, and how to perform the sort or merge. Save the command file on disk.

You must declare the input and output files in the command file, or use the INTO FROM phrase to declare them in the command line. Declare the output file in the command line. Declare some or all of the input files in the command file.

For a complete discussion of the Sort/Merge utility, see the *AOS Sort/Merge with Report Writer User's Manual.*

Sort/Merge Switches

/C
Indicate that you intend to enter a command file from your terminal.

/C=pathname
Use the file specified by pathname as the command file.

/L
Write statistical output and any error messages to the current list file.

/L=pathname
Write statistical output and any error messages to the file specified by pathname.

/N
Suspend execution of the imperative. The utility still checks the syntax of the command file statements.

/O
Delete the output file, if it exists, and recreate it with the results of the Sort/Merge process.

/S
Suppress statistical output.

/T=pathname
Save the command file that will be typed in at the terminal. The command file is saved in the file specified by pathname.

Example

```
) SORT/C=REORDER INTO TEACHERS FROM&)
&)REGISTER_6)
```

Invoke Sort/Merge to execute the command file REORDER. Sort/Merge sorts a copy of the records in input file REGISTER_6, and sends the sorted records to output file TEACHERS.

SPACE *Command*

Format

SPACE [*control-point-directory [new-max-size]* | *logical-disk*]

Purpose

Set or display the amount of disk space in a control-point directory or logical disk.

SPACE returns the maximum, current, and remaining disk space in 512-byte disk blocks. See the appropriate programmer's manual for more information about control point directories and logical disks.

You may use templates for the directory argument.

Command Switches

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

/V
Display the *control-point-directory* or *logical-disk* name.

Argument Switches

None.

Examples

```
) SPACE/V↵
= MAX 370889, CUR 304097, REM 66792
)
```

Display disk space in =, the working directory.

```
) SPACE :↵
MAX 37000, CUR 18000, REM 19000
)
```

Display disk space in :, the root directory.

SPEED *Utility*

Format

XEQ SPEED *[pathname]*

Purpose

Edit an ASCII text file.

Use SPEED to create a new or edit an existing source file. If *pathname* is a new file, SPEED asks:

Create new file?

Answer Y↓ to create it. Then SPEED prompts (!).

For more information, consult the *SPEED Text Editor (AOS and AOS/VS) User's Manual.*

SPEED Switches

/D
Display text automatically. If you include both /D and /I, /D is ignored.

/I=pathname
Take SPEED commands from file pathname not from @INPUT (the keyboard.)

Argument Switches

None.

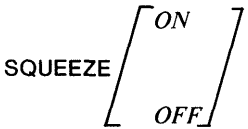
Example

```
) XEQ SPEED/I=PROCESS_REP.SCF REPORT1)
```

Invoke SPEED to edit the file REPORT1. The editing is not an interactive session, but is performed by executing the SPEED commands contained in the file PROCESS_REP.SCF.

SQUEEZE *Command*

Format



Purpose

Set or display the SQUEEZE setting.

When SQUEEZE mode is *ON*, the CLI outputs each sequence of two or more tabs or spaces as a single space. (Exception: output from the TYPE command.) Turn SQUEEZE mode *ON* to process any single CLI command by appending the /Q switch to the command.

Command Switches

/1,/2,/L,/L=pathname,/Q
See CLI commands page.

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

Argument Switches

None.

Examples

```
) SQUEEZE↵
OFF
) SQUEEZE ON↵
) SQUEEZE↵
ON
)
```

Display the current SQUEEZE setting, then set SQUEEZE to ON, and finally display the new SQUEEZE setting.

STRING *Command*

Format

STRING *[argument]...*

Purpose

Set or display STRING setting.

The STRING buffer can hold up to 127 characters.

Command Switches

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

/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. Kill the current STRING, and display STRING again (a null line).

!STRING *Pseudo-Macro*

Format

[!STRING]

Purpose

Expand to the STRING setting.

This pseudo-macro does not accept arguments.

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 [*ON* / *OFF*]

Purpose

Set or display the SUPERPROCESS setting.

Only privileged users can set SUPERPROCESS to *ON*.

The CLI precedes each prompt with a plus sign (+) when SUPERPROCESS is *ON*. If you enable both SUPERPROCESS and SUPERUSER, the CLI displays a number sign (#) before the prompt. The initial default prefix is a right parenthesis. The prompts are:

PROMPT	SUPERPROCESS	SUPERUSER
)	OFF	OFF
*)	OFF	ON
+)	ON	OFF
#)	ON	ON

Command Switches

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

SUPERPROCESS (continued)

/P

Set the current SUPERPROCESS mode to previous environment's SUPERPROCESS mode (no arguments allowed).

Argument Switches

None.

Examples

```
) SUPERPROCESS↵
OFF
) SUPERPROCESS ON↵
+)SUPERPROCESS↵
ON
+)
```

First, display the current SUPERPROCESS setting. Second, turn SUPERPROCESS ON. (Note that with SUPERPROCESS turned ON, the CLI outputs the prompt +) in the left margin.) Last, display the current SUPERPROCESS setting.

Format

SUPERUSER [*ON* / *OFF*]

Purpose

Set or display the SUPERUSER setting.

Only privileged processes can set SUPERUSER to *ON*.

The CLI precedes each prompt with an asterisk (*) when SUPERUSER is *ON*. If you enable both SUPERUSER and SUPERPROCESS, the CLI displays a number sign (#) before the prompt. The initial default prefix is a right parenthesis. The prompts are:

PROMPT	SUPERPROCESS	SUPERUSER
)	OFF	OFF
*)	OFF	ON
+)	ON	OFF
#)	ON	ON

Command Switches

/1, /2, /L, /L=pathname, /Q
See CLI Command page.

SUPERUSER (continued)

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

Argument Switches

None.

Examples

```
) SUPERUSER
OFF
) SUPERUSER ON
*)SUPERUSER
ON
*)
```

First, display the current SUPERUSER setting. Second, turn SUPERUSER ON. (Note that with SUPERUSER turned ON, the CLI outputs the prompt *) in the left margin.) Finally, display the current SUPERUSER setting.

Format

XEQ SWAT program-pathname *[program-argument]...*

Purpose

Invoke the SWAT™ interactive debugger.

The SWAT debugger is a high-level, interactive symbolic debugging system for high-level language programs. Use the SWAT debugger to check a program's correctness at the level of the source language, rather than at the assembly or machine-language level.

To use AOS SWAT under AOS/VS, invoke the program called SWAT16.PR. SWAT16 accepts the same switches as SWAT.

For a complete description of the SWAT debugger, see the *SWAT™ Debugger User's Manual.*

SWAT Switches

/AUDIT[=pathname]
Maintain an audit of this SWAT session. If you specify a pathname, the debugger writes the audit information to that file. Otherwise, the debugger writes the information to file program-pathname.AU.

/CONSOLE=consolename
Assign a new terminal for the program being debugged.

/DATA=pathname
Associate a new filename with @DATA.

/DEBUG
(AOS only) Begin execution in the Symbolic Debugger then move into the SWAT debugger.

SWAT (continued)

/INPUT=pathname
Associate a new filename with @INPUT.

/LIST=pathname
Associate a new filename with @LIST.

/OUTPUT=pathname
Associate a new filename with @OUTPUT.

Argument Switches

Use any argument switches appropriate for the program specified in program-pathname.

Example

```
) XEQ SWAT/DATA=DEBUG.DATA/LIST=&↵
&)DEBUG.LIST MYPROG↵
```

Invoke the SWAT utility to debug program MYPROG. For debugging, use DEBUG.DATA where the program calls for the generic @DATA file, and DEBUG.LIST where it calls for the generic @LIST file.

Format

SYSID *[argument]...*

Purpose

Set or display the unique system identifier.

Only PID 2 can set the system identifier.

Command Switches

/1, /2, /L, /L=pathname, /Q
See CLI Command page.

Argument Switches

None.

Example

```
) SYSID
REMULAC
)
```

This displays the current system identifier.

SYSINFO (AOS only) *Command*

Format

SYSINFO

Purpose

Display system information (AOS only).

SYSINFO displays current system environment information: revision, memory, master logical disk, identifier.

Command Switches

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

Argument Switches

None.

Example

```
) SYSINFO
SYSTEM REVISION: 03.30
SYSTEM MEMORY: 1023 PAGES
MASTER LOGICAL DISK: ROOT.7.25.79
R E M U L A C
)
```

This displays current system information.

Format

SYSLOG *[filename]*

Purpose

Set or display the SYSLOG setting.

The system normally writes information to the log file:

- System users: log on time, log off time, devices used, CPU use, and size of main memory allocated.
- Peripheral devices: type and number of errors encountered.

SYSLOG is privileged: only the initial CLI process (PID 2) may issue the SYSLOG call.

Command Switches

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

/ERROFF
(AOS only) Inhibit sending soft device error messages to the operator's terminal.

/ERRON
(AOS only) Enable sending of soft device error messages to the operator's terminal.

/START
Start system log file.

/STOP
Stop system log file.

SYSLOG (continued)

Argument Switches

None.

Example

```
) SYSLOG/START↓
)
```

Start recording in the system log file.

```
) SYSLOG↓
ON
)
```

SYSLOG with no arguments returns the current state of the log file.

Format

[!SYSTEM]

Purpose

Expand to the name of the operating system.

This pseudo-macro does not accept arguments.

Macroname Switches

None.

Argument Switches

None.

!SYSTEM (continued)

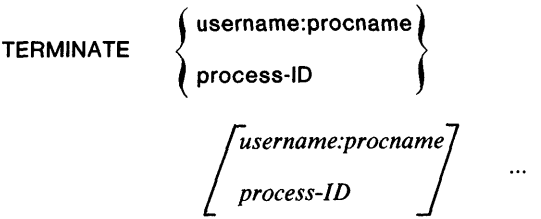
Example

Given a macro including

```
.
.
.
[!EQUAL,[!SYSTEM],AOS]
.
.
.
[!END]
[!EQUAL,[!SYSTEM],AOS/VS]
.
.
.
[!END]
```

The [!EQUAL,[!SYSTEM],AOS] and the first !END enclose commands that will be executed only when the system is running AOS. The [!EQUAL,[!SYSTEM],AOS/VS] and the second !END enclose commands that will be executed only when the system is running AOS/VS.

Format



Process

Terminate an inferior process.

You must supply the process-ID or the name of an inferior process unless SUPERPROCESS is ON. procname can be either a simple or complete process name.

Command Switches

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

/BREAKFILE[=pathname]
Produce a breakfile in the working directory at the time of termination. If you call TERMINATE with the simple /BREAKFILE switch, the system creates a break file with a default name. If you call TERMINATE with /BREAKFILE=*pathname*, the break file will have the specified name.

Argument Switches

None.

TERMINATE (continued)

Example

```
) PROCESS SMITH:PROGA↵
PID 17
        .
        .
        .
) TERMINATE 17↵
)
```

First, create a swappable son process that runs concurrently with the CLI. (The CLI displays the PID of the new process.) Then terminate the process.

Format

TIME *[new-time]*

See CLI Commands page.

Only the operator (PID2) can set the time. new-time is in the following format:

hours minutes seconds

Minutes and seconds are optional. You can use spaces or colons to separate entries.

Command Switches

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

Argument Switches

None.

Examples

```
) TIME↵
19:30:45
) TIME 8 45↵
)
```

Display the system time, then set the time to 8:45 A.M.

!TIME *Pseudo-macro*

Format

[!TIME]

Purpose

Expand to current system time.

!TIME does not accept arguments.

Macroname Switches

None.

Argument Switches.

None.

Example

```
) WRITE IT IS NOW [!TIME]↓
IT IS NOW 03:47:21
```

Format

TRACE

Purpose

Set or display the current trace mode.

Each trace mode can be turned on or off independently using switches. When trace mode is on, you can see the actual command line as the CLI processes it. The trace modes are

mode	symbol
command trace	***
macro trace	#
pseudo-macro trace	+++

If you use the /LOG switch, and a log file is open, trace output goes to the log file. The default is to @OUTPUT.

Command Switches

/1,/2,/L,/L=pathname, /Q

/COMMAND
Specify command trace.

/LOG
Send trace output to log file.

/KILL
Turn all trace modes off. (No arguments allowed.)

TRACE (continued)

/MACRO
Specify macro trace.

/PREVIOUS
Set to previous trace mode. (No arguments allowed).

/PSEUDO
Specify pseudo-macro trace.

/ON
Turn the following trace modes on.

/OFF
Turn the following trace modes off.

Argument Switches

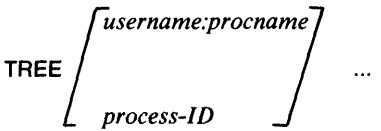
None.

Examples

```
) TRACE/MACRO↵
) TRACE/COMMAND↵
) TRACE↵
***TRACE
/COMMAND/MACRO
)
```

First turn on macro trace, then turn on command trace, then display both trace modes. The *** are the command trace symbols.

Format



Purpose

Display a process's family tree.

Command Switches

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

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
)
```

Display PID 7's tree. Then, display EXEC's tree.

TYPE *Command*

Format

TYPE pathname *[pathname]...*

Purpose

Type the contents of a file. SQUEEZE mode does not compress output from TYPE.

You may use templates for the pathname argument(s).

Command Switches

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

/V
Display the title 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↵
        .
        .
        .
) TYPE/2=ERROR FILE1 FILE2 FILE3↵
        .
        .
        .
)
```

Display MYFILE on the terminal. Then, set CLASS2 exceptional conditions to ERROR and type FILE1, FILE2, and FILE3. If any of the named files do not exist, the CLI will display an ERROR message and will cease typing. It will not type files whose names appear to the right of the nonexistent filename.

!UADD *Pseudo-Macro*

FORMAT

[!UADD argument$_1$ argument$_2$]

Purpose

Expand to the sum of two numbers.

The pseudo-macro requires two arguments, which must evaluate to unsigned decimal integers. Both arguments may be double precision. Double precision is in the range 0 to 4,294,967,295.

The value returned is the sum of argument$_1$ plus argument$_2$. The sum must be within the double precision range. If the two arguments produce a sum greater than this range, the value returned will be equivalent to the actual sum modulo 4,294,967,296.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE [!UADD 5 6]
11
)
```

Add two integers and display the sum.

Given a macro containing

```
.
.
.
[!ULE,[!UADD,[!SIZE FILE1],[!SIZE FILE2],10000]
.
.
.
```

Evaluate the two !SIZE pseudo-macros, and then the !UADD pseudo-macro, which returns the sum of the sizes of the two files. If the size does not exceed the indicated number of bytes,execute the code between the !ULE pseudo-macro and the next !ELSE or !END.

!UDIVIDE *Pseudo-Macro*

Format

[!UDIVIDE argument$_1$ argument$_2$...]

Purpose

Expand to the quotient of two numbers.

The pseudo-macro requires two arguments, which must evaluate to unsigned decimal integers. argument$_1$ may be double precision, which is in the range 0 to 4,294,967,295. argument$_2$ must be single precision, which is in the range 1 to 65,535.

The value returned is the integer result of argument$_1$ divided by argument$_2$. Note that argument$_2$ cannot equal zero.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE [!UDIVIDE 10 3]
3
)
```

Divide the first integer by the second, and display the quotient.

The macro TIP.CLI, which contains the following lines, prompts for input and computes 15% of the input amount:

```
PUSH
VAR0 [!READ TOTAL CHECK IN CENTS?,,,,]
WRITE TIP IS [!UDIVIDE,[!UMULTIPLY,15,[!VAR0]],100]
POP
```

Push a level, prompt for input, and assign the value to a variable. Multiply the amount by .15: that is, multiply the amount by 15, and then divide the product by 100. Output this final quotient.

!UEQ *Pseudo-Macro*

Format

[!UEQ argument$_1$ argument$_2$]

Purpose

Include input conditionally.

!UEQ begins a sequence of text that the CLI executes conditionally. The sequence must end with !END, and can also include !ELSE.

The !UEQ pseudo-macro must always have two arguments that evaluate to unsigned decimal integers. Both arguments must be in the range 0 to 4,294,967,295.

!UEQ compares the arguments: if they are equal, the CLI executes the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI does not execute the input between the !ELSE and the !END.

If the arguments are not equal, the CLI does not execute the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI executes the input between the !ELSE and the !END.

Macroname Switches

None.

Argument Switches

None.

Examples

Given a macro containing

```
.
.
.
[!UEQ,[!VAR0],17]
WRITE VAR0 = 17
[!ELSE]
WRITE VAR0 = [!VAR0]
[!END]
```

This macro will write VAR0 = 17 if the current value of VAR0 is 17. Otherwise, it will write VAR0 = n, where n is the current value of VAR0.

You can also code the macro as follows:

```
WRITE VAR0 =[!UEQ,[!VAR0],17]17
[!ELSE][!VAR0][!END]
```

!UGE *Pseudo-Macro*

Format

[!UGE argument$_1$ argument$_2$]

!UGE begins a sequence of text that the CLI executes conditionally. End the sequence with !END; you can also include !ELSE.

!UGE must have two arguments that evaluate to unsigned decimal integers. Both arguments must be in the range 0 to 4,294,967,295.

If argument$_1$ is greater than or equal to argument$_2$, the CLI executes the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI doesn't execute the input between the !ELSE and the !END.

If argument$_1$ is less than argument$_2$, the CLI does not execute the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI executes the input between the !ELSE and the !END.

Macroname Switches

None.

Argument Switches

None.

Examples

Given a macro containing.

```
.
.
.
[!UGE,[!VAR7],10]
WRITE VAR7 IS GREATER THAN OR EQUAL TO 10
[!ELSE]
WRITE VAR7 IS LESS THAN 10
[!END]
```

This macro will write the first message if the current value of VAR7 is greater than or equal to 10. Otherwise, it will write the second message.

You can also code the macro as follows:

```
[!UGE,[!VAR7],10]WRITE VAR7 IS GREATER THAN OR
EQUAL TO 10
[!ELSE]WRITE VAR7 IS LESS THAN 10[!END]
```

Note that there are no spaces between the bracketed !UGE statement and its WRITE command, nor between the !ELSE statement and its WRITE command.

!UGT *Pseudo-Macro*

Format

[!UGT argument$_1$ argument$_2$]

Purpose

Include input conditionally.

!UGT begins a sequence of text that the CLI executes conditionally. End the sequence with !END; you can also include !ELSE.

The !UGT pseudo-macro must always have two arguments that evaluate to unsigned decimal integers. Both arguments must be in the range 0 to 4,294,967,295.

If argument$_1$ is greater than argument$_2$, the CLI executes the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI doesn't execute the input between the !ELSE and the !END.

If argument$_1$ is less than or equal to argument$_2$, the CLI does not execute the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI executes the input between the !ELSE and the !END.

Macroname Switches

None.

Argument Switches

None.

Examples

Given a macro containing.

```
.
.
.
[!UGT,[!VAR1],11]
WRITE VAR1 IS GREATER THAN 11
[!ELSE]
WRITE VAR1 IS LESS THAN OR EQUAL TO & 11
[!END]
```

This macro will write the first message if the current value of VAR1 is greater than 11. Otherwise, it will write the second message.

You can also code the macro as follows:

```
[!UGT,[!VAR1],11]WRITE VAR1 IS GREATER THAN 11
[!ELSE]WRITE VAR1 IS LESS THAN OR EQUAL TO&
11[!END]
```

Note that there are no spaces between the bracketed !UGT statement and its WRITE command, nor between the !ELSE statement and its WRITE command.

Format

[!ULE argument$_1$ argument$_2$]

Purpose

Include input conditionally.

!ULE begins a sequence of text that the CLI executes conditionally. End the sequence with !END; you can also include !ELSE.

The !ULE pseudo-macro must always have two arguments that evaluate to unsigned decimal integers. Both arguments must be in the range 0 to 4,294,967,295.

If argument$_1$ is less than or equal to argument$_2$, the CLI executes the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI doesn't execute the input between the !ELSE and the !END.

If argument$_1$ is greater than argument$_2$, the CLI does not execute the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI executes the input between the !ELSE and the !END.

Macroname Switches

None.

Argument Switches

None.

Examples

Given a macro containing

```
.
.
.
[!ULE,[!VAR5],2]
WRITE VAR5 IS LESS THAN OR EQUAL TO 2
[!ELSE]
WRITE VAR5 IS GREATER THAN 2
[!END]
```

This macro will write the first message if the current value of VAR5 is less than or equal to 2. Otherwise, it will write the second message.

You can also code the macro as follows:

```
[!ULE,[!VAR5],2]WRITE VAR5 IS LESS THAN OR &
EQUAL TO 2
[!ELSE]WRITE VAR5 IS GREATER THAN 2[!END]
```

Note that there are no spaces between the bracketed !ULE statement and its WRITE command, nor between the !ELSE statement and its WRITE command.

!ULT *Pseudo-Macro*

Format

[!ULT argument$_1$ argument$_2$]

Purpose

Include input conditionally.

!ULT begins a sequence of text that the CLI executes conditionally. End the sequence with !END; you can also include !ELSE.

The !ULT pseudo-macro must always have two arguments that evaluate to unsigned decimal integers. Both arguments must be in the range 0 to 4,294,967,295.

If argument$_1$ is less than argument$_2$, the CLI executes the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI does not execute the input between the !ELSE and the !END.

If argument$_1$ is greater than or equal to argument$_2$, the CLI does not execute the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI executes the input between the !ELSE and the !END.

Macroname Switches

None.

Argument Switches

None.

Examples

Given a macro containing

```
.
.
.
[!ULT,[!VAR2],13]
WRITE VAR2 IS LESS THAN 13
[!ELSE]
WRITE VAR2 IS GREATER THAN OR EQUAL TO 13
[!END]
```

This macro will write the first message if the current value of VAR2 is less than 13. Otherwise, it will write the second message.

You can also code the macro as follows:

```
[!ULT,[!VAR2],13]WRITE VAR2 IS LESS THAN 13
[!ELSE]WRITE VAR2 IS GREATER THAN OR EQUAL &
TO 13[!END]
```

Note that there are no spaces between the bracketed !ULT statement and its WRITE command, nor between the !ELSE statement and its WRITE command.

!UMODULO *Pseudo-Macro*

Format

[!UMODULO argument$_1$ argument$_2$]

Purpose

Expand to the result of the modulus operation on two numbers.

!UMODULO requires two arguments that must evaluate to unsigned decimal integers. argument$_1$ may be double precision, which is in the range 0 to 4,294,967,295. argument$_2$ must be single precision, which is in the range 1 to 65,535.

The value returned is the integer result of argument$_1$ modulo argument$_2$. For unsigned integers, this value is equivalent to the remainder produced by dividing argument$_1$ by argument$_2$. Argument$_2$ cannot equal zero.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE [!UMODULO 10 3]↵
1
)
```

Perform the modulus operation on two integers, and display the results.

The macro DIVIDE.CLI, which contains the following command lines, divides the first argument by the second argument, and displays the quotient and the remainder (that is, the result of the modulus operation):

```
WRITE THE QUOTIENT IS [!UDIVIDE,%1%,%2%]
WRITE THE REMAINDER IS [!UMODULO,%1%,%2%]
```

Evaluate and display the !UDIVIDE pseudo-macro, then evaluate and display the !UMODULO pseudo-macro.

!UMULTIPLY *Pseudo-Macro*

Format

!UMULTIPLY argument$_1$ argument$_2$]

Purpose

Expand to the product of two numbers

!UMULTIPLY requires two arguments that must evaluate to unsigned decimal integers. argument$_1$ may be double precision, which is in the range 0 to 4,294,967,295. argument$_2$ must be single precision, which is in the range 0 to 65,535.

The value returned is the result of argument$_1$ multiplied by argument$_2$. The product must be within the double precision range. If the two arguments create a product greater than this range, the value returned will be equivalent to the actual product modulo 4,294,967,296.

Macroname Switches

None.

Argument Switches

None.

Examples

```
) WRITE [!UMULTIPLY 66777 92]↓
6143484 )
```

Multiply two integers and display the product.

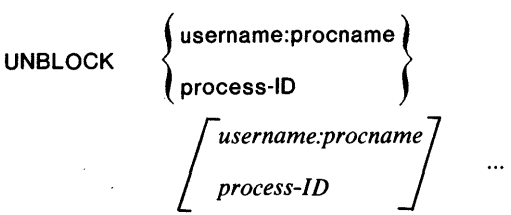
The macro CENT.CLI, which contains the following command lines, takes as its argument an integer value in degrees Fahrenheit (the value cannot be less than 32). It returns a rough equivalent in degrees centigrade. The macro is based on the formula, C = 5/9(F - 32):

```
WRITE CENTRIGRADE = [!UDIVIDE,[!UMULTIPLY,5,
[!USUBTRACT,%1%,32]],9]
```

Subtract 32 from the Fahrenheit argument. Multiply the difference by 5/9: that is, multiply the amount by 5, and then divide the product by 9. Finally, output this quotient.

UNBLOCK *Command*

Format



Purpose

Unblock a previously blocked inferior process.

Supply the processID or the process name. The process name can be for either a simple or a full process.

The process must be previously blocked and inferior, unless you have SUPERPROCESS privilege.

Command Switches

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

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. Block the new process, perform the required operations, then unblock the new process.

!UNE *Pseudo-Macro*

Format

[!UNE argument$_1$ argument$_2$]

Purpose

Include input conditionally.

!UNE begins a sequence of text that the CLI executes conditionally. End the sequence with !END; you can also include !ELSE. pseudo-macro.

!UNE must always have two arguments that must evaluate to unsigned decimal integers. Both arguments must be in the range 0 to 4,294,967,295.

!UNE compares the two arguments. If they are not equal, the CLI executes the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI doesn't execute the input between the !ELSE and the !END.

If the arguments are equal, the CLI does not execute the input up to the !ELSE or !END pseudo-macro. If there is an !ELSE, the CLI executes the input between the !ELSE and the !END.

Macroname Switches

None.

Argument Switches

None.

Examples

Given a macro containing

```
.
.
.
[!UNE,[!VAR0],17]
WRITE VAR0 = [!VAR0]
[!ELSE]
WRITE VAR0 = 17
[!END]
```

if the current value of VAR0 is not 17, this macro will write VAR0 = n, where n is the current value of VAR0. Otherwise, it will write VAR0 = 17.

You can also code the macro as follows:

```
WRITE VAR0 =[!UNE,[!VAR0],17][!VAR0]
[!ELSE]17[!END]
```

!USERNAME *Pseudo-Macro*

Format

[!USERNAME]

This pseudo-macro does not accept arguments.

Purpose

Expand to the CLI username.

Macroname Switches

None.

Argument Switches

None.

Example

```
) WRITE CALL ME [!USERNAME]
CALL ME ISHMAEL.
)
```

Format

[!USUBTRACT argument$_1$ argument$_2$]

Purpose

Expand to the difference of two numbers.

!USUBTRACT requires two arguments that must evaluate to unsigned decimal integers. Both arguments may be double precision. Double precision is in the range 0 to 4,294,967,295.

The value returned is the result of argument$_1$ minus argument$_2$. The pseudo-macro cannot return a negative number. If argument$_1$ is less than argument$_2$ the value returned is equivalent to the absolute value of the actual difference, modulo 4,294,967,296.

Macroname Switches

None.

Argument Switches

None.

!USUBTRACT (continued)

Examples

```
) WRITE [!USUBTRACT 17 5]↓
12
)
```

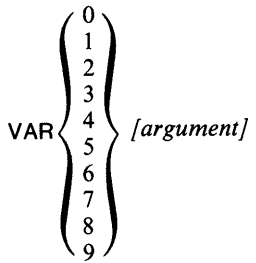
Subtract one integer from another and display the difference.

The macro DIFF.CLI, which contains the following command lines, takes as its arguments two integers. It returns their difference, either positive or negative:

```
[!UGE,%1%,%2%]
WRITE REMAINDER IS [USUBTRACT,%1%,%2%]
[!ELSE]
WRITE REMAINDER IS -[!USUBTRACT,%2%,%1%]
[!END]
```

First, determine if the first argument is greater than the second. If it is, subtract the second from the first and display the result. If the second argument is greater than the first, subtract the first from the second, put a negative sign before the difference, and display the result.

Format



Purpose

Set or display the valuue of variable VAR0.

VAR0 displays the current value of variable VARn or sets its current value to the specified value. The CLI provides 10 variables, VAR0 through VAR9. *argument* must evaluate to a decimal integer in the range 0 to 4,294,967,295.

Command Switches

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

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

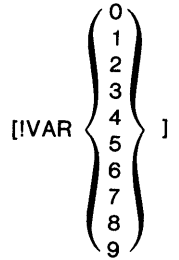
VARn (continued)

Example

```
) VAR0↓
0
) VAR0 34573↓
) WRITE [!UADD [!VAR0] 2]
34575
)
```

First, display the current value of VAR0. Then reset the value. Finally, include the pseudo-macro !VAR0 in a command line to use the current value of VAR0.

Format



Purpose

Display the current value of variables VAR0 through VAR9.

The ten pseudo-macros !VAR0 through !VAR9 display the current value of the ten CLI variables VAR0 through VAR9. These pseudo- macros do not accept arguments.

Macroname Switches

/P
Expand to previous environment's value for VAR0.

Argument Switches

None.

!VARn (continued)

Examples

```
) WRITE THE CURRENT VALUE OF VAR0 IS [!VAR0]↓
THE CURRENT VALUE OF VAR0 IS 0
) PUSH↓
) VAR0 39↓
) WRITE NOW THE CURRENT VALUE OF VAR0 IS &↓
&)[!VAR0]↓
NOW THE CURRENT VALUE OF VAR0 IS 39
WRITE [!VAR0/P]↓
0
)
```

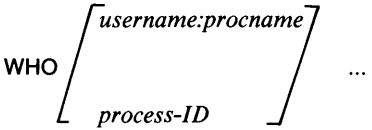
First, evaluate [!VAR0] and write the current value of VAR0. Then change environment, and give VAR0 a new value. Evaluate and write [!VAR0] for the current environment, and then, using the /P switch, for the previous environment.

Generate a new AOS/VS operating system (AOS/VS only).

VSGEN is the system generation utility for AOS/VS. It creates an AOS/VS system specifically designed to manage the hardware and software configuration at your installation. VSGEN requests information about the peripheral devices the system will need to manage. It also allows you to adjust system performance by specifying values for certain system parameters.

For a complete description of VSGEN and its operation, consult *Managing AOS/VS*.

Format



Purpose

Display process information.

WHO displays the PID, username, process name, and program name of a process. Without argument(s), the command applies to the CLI.

Command Switches

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

Argument switches.

None.

Examples

```
) WHO↓
PID: 17 COSTLEY CON13 :CLI.PR
)
```

The current process's ID is 17, its username is Costley, its process name is CON13, and the program it runs is CLI.PR.

```
) WHO 007↓
PID: 7 JAMES_B CON7 :SPY.PR
)
```

WRITE *Command*

Format

WRITE *[argument]...*

Purpose

Display arguments.

WRITE is useful in macros for either writing to the terminal explaining what is happening, or writing a record/log to a LISTFILE explaining what has happened or is happening.

Command Switches

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

Argument Switches

None.

Examples

```
) WRITE (A B)_<X Y>↓
A_X A_Y
B_X B_Y
)
```

The WRITE command is useful when you want to experiment with the CLI command line operators (parentheses and angle brackets).

Also use the WRITE command in macros to write to your terminal, explaining what is happening. For example:

```
) WRITE/L START AT [!DATE][!TIME]↓
```

XEQ *Command*

Format

XEQ pathname *[argument-to-new-program]...*

Purpose

Execute a program.

The CLI is blocked until the subordinate process terminates. The subordinate's termination IPC message (if any) goes to STRING (if you used the /S switch), to the current list file (if you used the /L switch), the temporary list file (if you used the /L= switch), or to @OUTPUT (if you didn't use either of these switches). The CLI may take exceptional action depending on whether or not the subordinate process returns exceptional condition flags.

Command Switches

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

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

/M
Create input for the program from the macro body. The last line of the macro body must contain a single).

/S
Return the program's termination message to STRING; default is @OUTPUT.

Argument Switches

Use as needed by the new program.

Examples

) XEQ LINK OBJ1↵

Call the LINK utility; that is, create a subordinate process whose program is the LINK utility. Have LINK produce an executable program file from the object file OBJ1.

) XEQ MYPROG 0 1↵

Execute a program named MYPROG. Note that MYPROG can get the arguments 0 and 1 for use in whatever way you choose.

) XEQ/S PROG2↵

Execute PROG2; divert PROG2's termination message (if any) to STRING instead of to @OUTPUT.

SED Commands

To open a file

EXECUTE SED *[switches] [pathname]*

Open a file for editing; control which directories store .ED and temporary files; customize the editing sessions's characteristics (/PROFILE switch).

To add text

APPEND *[FROM pathname] [range]*

Add text to end of page from terminal or from source file.

DUPLICATE *[range] destination*

Copy text from one location in page to another or onto another file.

INSERT *[address] [FROM pathname] [range]*

Insert text - from terminal or source file - before location in page. Specify range of lines.

To change text

MODIFY *[address]*

Revise a line or range of lines of existing text, using cursor and line control characters, screen position keys.

MOVE *[range] destination*

Move text from one location in page to another, or onto another file.

REPLACE *[range]*

Delete text; replace with new text typed from terminal.

SUBSTITUTE "searchstring" *[FOR]* "searchstring" *[[IN]range]*

Substitute word or phrase for another word or phrase throughout a range of text.

To delete and restore text

DELETE *[range]*

Remove a range of text from file.

UNDO_LAST_DELETE (may be abbreviated as UNDO)

Restore most recently deleted text to file.

Display information

HELP *[word]*

Display information on terminal about commands and keywords.

DISPLAY_STATUS

Display file status information: edit filename, current page and line, last page and last line of current page, view range and display mode, whether TYPER and BLANK modes are ON or OFF, and if line numbers are being displayed.

LIST *[range]*

List a range of text on terminal.

PRINT *[range]*

Print a range of text on @LIST (usually lineprinter); @ LIST must be set beforehand.

VIEW

Refresh the screen. Use SET_VIEW [range] to specify number of lines viewed.

SET_VIEW *[range]*

Specify number of lines to be viewed.

To format text

CLEAR BLANK_MODE

Remove ability to pad lines with trailing blanks established by SET BLANK_ MODE command.

CLEAR FUNCTION_KEY number

Remove definition of function key established by SET FUNCTION_KEY MODE command.

CLEAR LINE_NUMBER_DISPLAY

Remove line number display established by SET_LINE_NUMBER_DISPLAY command.

CLEAR_TYPER_MODE

Remove vertical column feature established by SET_TYPER_MODE command.

CUT_LINE address column.

Split a line into two lines;(hard-copy terminals only).Current line must be first line on the page.

CUT_WHILE_EDITING_KEY

Split a line into two lines;(not hard-copy terminals). Current line must be first line on the page.

JOIN *[address]*

Remove a page break in file.

PASTE_LINES range *[OR address]*

Merge a range of lines into a single line, or one line (address) onto next line.

SET BLANK_MODE

Allow padding of lines with trailing blanks.

SET DISPLAY_MODE number

Set relative brightness of lines displayed, and/or display of current line. Mode numbers are: 0 (all dim); 1 (all bright); 2 (all bright and * marks the current line); 3 (current line dim, all else bright); 4 thru 11+, same as 0.

SET FUNCTION_KEY number "string"

Set function keys 4-8 to execute one or more commands defined by "string". Separate commands with semicolons.

SET LINE_NUMBER_DISPLAY

Redisplay line numbers cleared with CLEAR LINE_NUMBER-DISPLAY command.

SET TYPER_MODE

Cause vertical arrow keys to respect column position.

SET VIEW_RANGE*[TO number]*

Set number of lines to display before and after current line.

SPELL "string"

Open and search SED.dictionary: alphabetical list of words in lower case, followed by New Line; "string" can be either lower or upper case.

SPLIT*[address]*

Set page break in file before address or current line.

To locate text

BACKFIND *"searchstring"*

Find word or phrase in a page of text beginning at line before current line, working backwards.

FIND "searchstring" *[[IN]range]*

Find word or phrase in range of text beginning at any line on page, working forwards.

POSITION address

Move current line position to address in file.

Miscellaneous

DIRECTORY pathname

Change working directory to pathname.

DO cli-command

Execute a CLI command and return to editing session.

EXECUTE pathname

Execute SED commands from previously created file and return to session.

To Close or update a file

ABANDON

Abort editing session without updating file with new changes.

BYE

Conclude editing session; update file with new changes.

BYE PERM OFF

Turn PERMANENCE off on old version of file being edited, allowing exit.

SAVE

Update a copy of file, including changes, without concluding session.

Cursor and Line Control Characters

CTRL-A

Move cursor to end of line. Repeats last command isued in command mode.

CTRL-B

Move cursor backward to last letter in each word.

CTRL-E

Insert one or more characters before cursor; followed by), terminates insert mode and displays edited line.

CTRL-F

Move cursor forward to first letter in each word.

CTRL-H

Move cursor to beginning of line. (Same as HOME key.)

CTRL-J

Terminate modification of line. (Same as NEW LINE key.)

CTRL-K

Erase all characters to right of cursor.(Same as ERASE EOL key.)

CTRL-M

Erase all characters to right of cursor and terminate modification of line, placing cursor at column one of next line. (Same as CR key.)

CTRL-X

Move cursor to right one character. (Same as → key.)

CTRL-Y

Move cursor to left one character. (Same as ← key.)

CTRL-U

Delete all characters in line.

CTRL-I TAB key

Move cursor to next tab stop: columns 9, 17, 25, 33, 41, 49, 57, 65, 73.

DEL

Erase character to left of cursor and close up line.

ESC

Terminate APPEND, INSERT, MODIFY or REPLACE mode. Press ESC before NEW LINE to enter line with no changes.

Function Keys

To set key functions for keys numbered 4 thru 8, see the SET_FUNCTION_KEY command.

For fixed key functions see the SED Template for D2 or D200 terminals.

A

Append a page or window from input file to current edit file.

: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 zero (0) if the Append failed.

nBCx, except for 0BCx

Copy text from the current position (CP) up to the nth NEW LINE. Copy the next n lines of text, but first line copied will only contain text from CP forward.

1BCx

Copy text from CP to the end of the line.

0BCx

Copy text from the beginning of the current line up to CP.

-nBCx

Copy text from the beginning of the nth line preceding the current line through n NEW LINEs up to CP. Copy text from previous n lines and, if CP is not at beginning of current line, text on the current line up to CP.

-1BCx

Copy text from the previous line and current line up to CP.

m,nBCx

Copy text from the character after the mth character up to and including the nth character. Simple numbers refer to positions in current buffer. Allow numerical expressions containing arithmetic operators, or variables and pseudo-variables such as V0 and VN.

#BCx

Abbreviate 0,ZBCx, copying the entire current buffer to buffer x.

BFB,BFC,BFNR,BFNW,BFO,BFR,BFU,BFW

Execute an FB,FC,FNR,FNW,FO,FR,FU, or FW command local to current buffer.

BGx

Get a line from the terminal and copy it to a buffer.

BKx

Deactivate (kill) buffer x.

BSx

Make buffer x the current 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 status of all current buffers. Type buffer status of buffer x.

&B?

Display octal character count in alternate radix. Use WR command for decimal value.

^Bx

Insert contents of buffer x into command string in place of ^Bx command.

C[text1] $[text2]$

Search entire buffer for text1 and replace with text2.

0C[text1] $[text2]$

Search from beginning of current line up to CP.

1C[text1] $[text2]$

Limit search from CP through the end of current line (NEW LINE).

nC[text] 1$[text2]$ except for 0Ctext1$text2$

Limit search for text1 from CP up to the nth NEW LINE forward. Search through the next n lines forward, but search only the first line from CP forward.

-1C[text] 1$[text2]$

Limit search to the preceding line and the current line up to CP.

-nC[text1] $[text2]$

Limit search to n lines preceding the current line; if CP is not at beginning of current line, search current line up to CP.

m,nC[text1] $[text2]$

Limit search from character after the mth character up to and including the nth character.

[-]nC[text1] $[text2]$

Search next n carriage returns for text1 and replace with text2. If you include minus sign, search preceding nlines up to CP for text1 and replace with text2.

nD

Delete n characters starting at CP.

/D

Display 20 lines of text automatically. Format:) X SPEED/D filename)

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. For files opened by FO, create backup with .BU file extension. Clear buffer.

FC

Close all current global input and output files. FC does not write text in edit buffer to output file before closing it.

FNR*[pathname]*

Close current input file. If you include *pathname*, close current input file, and open another input file with the specified name.

FNW*[pathname]*

Close current output file. If you include *pathname*, close current output file, and open a new output file with the specified name.

FOpathname

Open pathname for input, pathname.TM for output, and Yank (Y) page into buffer. Set update mode ON.

FRpathname

Open pathname for input. Do not use FR if update mode is ON.

FWpathname

Create and open 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.

H

Normal exit from SPEED, return to SPEED's parent process, usually CLI.

/I=filename

Invoke a command file. Format:) X SPEED/I=command.file text.file↵

I[text]$

Insert text into buffer at CP.

↑I[text]$

Insert text into buffer at CP with leading tab.

nI

Insert ASCII decimal equivalent of n at CP.

n\

Insert ASCII representation of decimal n at CP.

J,OJ,1J

Move CP to beginning of buffer.

nJ

Position CP at beginning of line n.

[-]nK

Delete characters from CP through next n carriage returns. If you include minus sign, delete preceding n lines up to CP.

m,nK

Delete characters (M+1) through n in 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.

nM

Move CP across N characters. If N is positive, CP moves to right, if N is negative, CP moves to left.

Ntext

(Nonstop search). Search current buffer and rest of input file for text. Copy buffer to output file if text is not found.

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 form feed appended. If you include minus sign, copy preceding n lines plus characters on current line up to CP to the output file with form feed appended.

0P

Copy current line from beginning through CP to output file with form feed appended.

1P

Copy text in buffer to output file from CP to end of line.

nP, except for 0P

Copy text in current buffer to output file text from CP up to nth NEW LINE. Copy next n lines of text, but first line copied contains only text from CP forward.

-1P

Copy text in buffer to output file from previous line and current line up to CP.

-nP

Copy text in buffer to output file from beginning of nth line preceding current line through n NEW LINEs up to CP. Copy text in buffer to output file from previous n lines; if CP not at beginning of current line, copy text on current line up to CP.

m,nP

Copy characters (M+1) through n in buffer to output file with form feed appended.

#P

Abbreviate 0,ZP. Copy entire current buffer to buffer x.

0PW

Copy current line from beginning through CP to output file without form feed.

1PW

Copy text in buffer to output file from beginning of line to CP.

nPW, except for 0PW

Copy text in buffer to output file text from CP up to end the nth NEW LINE. Copy next n lines of text; first line copied contains only text from CP forward.

-1PW

Copy text in buffer to output file from previous line and current line up to CP.

-nPW

Copy text in buffer to output file from beginning of nth line preceding current line through n NEW LINES up to CP. Copy text in buffer to output file from previous n lines; if CP is not at beginning of current line, copy text on current line up to CP.

m,nPW

Copy text in buffer to output file from character after the mth character up to and including nth character. Numbers refer to positions in current buffer. Also use numerical expressions containing arithmetic operators, variable and pseudo variables such as V0 and VN.

#PW

Abbreviate 0,ZPW. Copy entire current buffer to buffer x.

Qtext

(Quick search.) Same as Ntext except SPEED does not copy the buffer to the output file if text is not found.

R

(Roll). Copy current buffer to output file, clear buffer, and Yank (Y) 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 minus sign, search preceding n lines plus characters up to CP on current line.

0Stext

Search from beginning of current line up to CP.

1Stext

Search from CP to next NEW LINE character: through next n lines, including current line.

nStext, except 0Stext

Search from CP forward to nth NEW LINE character: through next n lines, including current line.

-1Stext

Search preceding line and current line up to CP.

-nStext

Search from nth NEW LINE preceding CP up to CP: preceding n lines and current line up to CP.

m,nStext

Search from character following the mth character up to and including nth character. Specify simple numbers, and SPEED counts m and n from the beginning of buffer. Use more complicated numerical expressions and specify m and n using variable and pseudo-variables.

#Stext

Abbreviate 0.ZStext. Specify entire current buffer as search range.

@S%text%

Set first character following command name as delimiter. Here, %.

:Stext

Inhibit error messages; receive +1 for success, 0 for failure.

T

Type current line, indicating CP, (↑).

0T

Display current line from beginning of line to CP.

1T

Display current line from CP to first NEW LINE character.

nT, except 0T

Display contents of current buffer from CP up to nth NEW LINE following CP; show n lines, counting line where CP resides.

[-]nT

Type n lines of curent buffer. If you include minus sign, type n lines preceding and including current line up to CP.

-1T

Display immediately preceding line and current line up to CP.

-nT

Display contents of n preceding lines and the current line up to CP.

m,nT

Display contents of current buffer from character following mth character up to and including nth character. Simple numbers specify positions from beginning of buffer. Specify complex numerical expressions and with variables and pseudo-variables.

ZT

Display text from CP to end of buffer. If text exceeds display characteristics of screen, text will roll to bottom of buffer.

#T

Type entire buffer. Abbreviates 0,ZT. If length exceeds screen characteristics, text will roll to bottom of buffer.

Vv

Represent current value of variable v.

VC

Represent decimal value of ASCII character following CP.

VC=

Display decimal value of ASCII character following CP.

VDv

Decrement value of variable v; represent decremented value.

VIv

Increment value of variable v; represent incremented value.

VL

Represent number of current line.

VL=

Display number of current line.

VL=

Display number of current line.

VN

Represent number of lines in current buffer.

VN=

Display number of lines in current buffer.

VP

(Value position.) Get CP position before last search.

nVSv

Set variable v to value n and return that value.

WA

(Window argument.) Set new value for default argument of deletion and movement commands.

0WA

Sets window argument value at 0. Commands:

D (no effect)

J Move CP to beginning of buffer.

K Kill characters from beginning of current line up to CP.

L Move CP to beginning of current line.

M Leave CP in current position.

nWA

Sets window argument at +1. Commands:

D Delete one character to right of CP.

J Move CP to beginning of buffer.

K Kill characters from CP through end of line, including NEW LINE character.

L Move CP to right of next NEW LINE: one line forward.

M Move CP one character to right.

WC=

Display current value of case control mode:

0 Case control off.

1 Up-shift.

2 Down-shift.

0WC

Turn case control off.

nWCx

If n is positive, shift up any character preceded by x; if negative, shift down any characters preceded by x.

nWCy

If n is positive, shift up using x as shift character and y as shift-lock character; if negative, shift down using x as shift character and y as shift-lock character.

WD

(Window display). Set automatic display mode value.

0WD

Turn window display mode off. Issue T commands for display.

nWD

Turn window display off; n can be positive from 1 to 10; set at n SPEED displays n lines preceding and n lines following the CP, and shows the CP as a blinking asterisk (on 6052, 6053, D100 and D200 models), or caret within parentheses (other models.)

WM

(Window mode.) Represent value of data input mode.

WM=

Display current state of data input mode (0 is page, n is window).

0WM

Set SPEED to read text into buffer mode in page mode, from form feed to form feed. SPEED treats -nWM as 0WM.

nWM

Set SPEED to read text into buffer in window mode, n lines at a time.

WP

(Window position.) Set position of CP after unsuccessful searches. SPEED treats all nonzero arguments as 1.

0WP

CP positioned at:

- beginning of buffer for default (numerically unmodified) C and S commands.
- n lines beyond previous position for nC and nS commands.
- position before search for -nC and -nS commands.

nWP, when n is not 0

CP positioned at:

- position before search for default C and S commands.
- position before search for nC and nS commands.
- n lines before previous position for -nC and -nS commands.
- after mth character for m,nC and m,nS commands.

WR

(Window radix.) Set new alternate radix.

nWR

Set new radix at n (positive numbers from 2 to 36.) Default: 8 (octal).

WS

(Window shifts.) Represent current status of case mode. Default=0.

0WS

Permit case-independent search: u in command matches U and u in text; L in command matches both L and l in text. Let search command(s) match text, regardless of whether upper- or lowercase.

nWS, n not =0

Force case-independent search; u in command matches u but not U in text; L in command matches L but not l in text. Case must match text in search command; n is decimal from 0 to 65535.

WS=

Display current status of case mode.

Xclicommand

Execute CLI command without exiting SPEED.

Y

(Yank.) Clear buffer and read in one page from input file.

Z

(Last character.) Represent total number of characters in current buffer.

Z=

Display total number of characters in current buffer.

↑Z

Accept any character in his position; use with search commands S,C,N and Q.

$(ESC)

(Standard delimiter.) Echo after ESC or BREAK ESC. Do not use before CTRL-D.

!

(Prompt.) SPEED is ready for input from terminal.

!label!

Skip this material.

n"Xcommand-string"

(Conditional execution.) Carry out commands if numerical argument has specified value:

n"Gx'y If n >0, do x; otherwise, do y.

n"Lx'y If n <0, do x; otherwise, do y.

n"Ex'y If n =0, do x; otherwise, do y.

n"Nx'y If n not=0, do x; otherwise, do y.

#command

Entire contents of buffer as the command states.

$

Switch to alternate redix.

*

Multiplication operator: m*n.

*(blinking)

Position of CP.

+

Addition operator: m+n.

-

Unary minus operator: -n.

.(period)

Current position of CP.

.=

Display current CP position.

/

Division operator: m/n.

:

Multipurpose modifier: :command-name.

- Modify Search and file input commands A, Y, R, S, C, N and Q to return a value of 1 if command succeeds, 0 if it fails.
- Modify output commands P, nP, m,nP, PW, m,nPW to delete all characters in the buffer after output.
- Modify execute command X to execute program from SPEED.

;

(Conditional termination.) Terminate execution of command line upon failure or success of last command. Jump command loop if last search 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<x>

(Command loop.) Execute bracketed command line n times. Skip if n <0.

n=

(Equals.) Display or type value of numeric argument n.

Use these commands:

VC= ASCII equivalent of next character to right.

VL= Current line number.

VM= Number of moves from beginning of line to CP.

VN= Number of NEW LINEs in current buffer.

VP= Previous position of CP.

WA=Default argument mode.

WC=Case control mode.

WD=Number of lines in display at one time.

WM=Number of lines SPEED reads into buffer at one time.

WP=Mode for positioning CP after unsuccessful search.

WR=Alternate radix.

WS=Mode for matching case letters during search.

Z= Characters in current buffer.

.=Characters from beginning of buffer to CP.

n@=

Suppress line break between displayed value and next SPEED prompt.

n&=$$

Express numerical value in alternate radix.

?

(Trace mode toggle.) Turn trace mode on, if off; off, if on. Default: off.

@

Set temporary delimiter. Modify INSET and SEARCH commands to change text delimiter. @ follows command, defining delimiter.

#

Equals double argument 0,Z (edit entire buffer.)

n\

(Backslash.) Insert string of ASCII digits of decimal number n into buffer at CP.

-x

Place previous command string in buffer x. This must be the first command after the prompt.

↑*

(Logical AND. Boolean intersection). Format: x↑*y

↑+

(logical Inclusive OR. Boolean union.) Format: x↑+y

↑-

(Logical NOT. Boolean complement.) Format: ↑-x

↑/

(Logical Exclusive OR. Boolean symmetric difference.) Format: x↑/y

_x

Save previous command line in buffer. Use ↑Bx command to reissue.

↑SHIFT-N (↑↑) or ↑SHIFT-G(↑↑)

Position CP at SHIFT-N (ASCII 036 octal) in search string if search is successful.

↑SHIFT-O(↑_)

Interpret next character literally. (ASCII 037 octal).

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 not =0.

!string!

Define a label named string in the command string.

The DEBUG Utility AOS only

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

DEBUG Commands

Do not apply these DEBUG commands to DEDIT.

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

Display contents of floating point accumulator.

SFP expression 1; expression 2

Set floating point accumulator.

P*[breakpoint-count]*

Start user-program execution.

DEBUG/DEDIT Commands

STAB)

FILENAME? symbol-table name)

Append a symbol table.

DSTAB *[<symbol file number>]*

Undo STAB command; close specified symbol file.

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

Change display and address mode.

CLOSE

Close dialog file.

expression *[;mode charcter-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.

SLIST

Clarify order of symbol search: local and global symbols recognized.

BYE

Terminate debugging/editing.

The DEBUG Utility AOS/VS only

Memory access commands

address/

Display one-word value starting at word location address and open that word for modification.

address\

Display two-word value starting at word location address and open those two words for modification.

)!

Close the open location.

(carriage return)

Close the open location and open the subsequent location.

↑(uparrow)

Close the open location and open the previous location.

$S

Display or search a range of memory locations.

Commands that Access MV/8000 Machine State Registers.

$A

Display contents of the four fixed-point accumulators and the carry bit.

n$A

If 0<=n<=3, open fixed point accumulator n; if n=4, open the carry bit

$E

Display the four stack registers.

n$E

Open the stack pointer (n=0), frame pointer (n=1), stack limit (n=2), or stack base (n=3) register.

$F

Display the four floating point accumulators and the floating point status register (FPSR).

n$F

If 0<=n<=3, open floating point accumulator n; if n=4, open the first 32 bits of the FPSR; if n=5, display the last 32 bits of the FPSR (i.e.,floating point PC).

$L

Open the program counter (PC).

$V

Open the process status register (PSR).

Commands that Access Debugger Registers.

$G

Open the ring register.

$N

Open the output radix register.

n$Q

Open the proceed count register for breakpoint n.

$T

Open the global display mode register.

Commands that Control Program Execution.

$B

Display all breakpoints

address$B

Set a breakpoint at location address.

address,condition$B

Set a conditional breakpoint at location address.

$D

Delete all breakpoints.

n$D

Delete breakpoint n.

n$P

Continue program execution at the last breakpoint encountered and set the proceed count for that breakpoint to n.

n$Q

Open the proceed count register for breakpoint n.

$R

Start or resume program execution at the current program counter (PC).

address$R

Start or resume program execution at location address.

Commands Related to Symbol Use

$I

Display the currently defined temporary symbols.

symbol,value$I

Definer a temporary symbol.

$J

Delete all temporary symbols.

symbol$J

Delete a specific temporary symbols.

$X

Disable the current symbol table file.

file$X

Disable the current symbol table file (if any) and enable a new one.

Display Mode Commands

(f-key)

Redisplay the last value in the specified display mode (for terminals with function keys).

CTRL-(f-key)

Modify the global display mode (for terminals with function keys).

TAB-n

Redisplay the global display mode (for terminals without functon keys).

TAB-TAB-n

Modify the global display mode (for terminals without function keys)

General Use Commands

$C

Push to the CLI.

$H

Help command: list various topics on DEBUG.

keyword$H

Help command: list information on keyword.

$Y

Disable current log file.

file$Y

Disable current log file (if any) and enable new one.

$Z

Terminate DEBUG session.

$?

Display diagnostic error message for 1st error.

;comment

Enter character string comment in current log file.

093-000150-02

