

DTIC FILE COPY

2

David Taylor Research Center

Bethesda, Maryland 20084-5000

AD-A232 370

CISD-90/02 30 September 1990

Computer & Information Services Department
Departmental Report

COMPUTER CENTER REFERENCE MANUAL, VOLUME 2

David V. Sommer
Sharon E. Good

Approved for Public Release:
Distribution Unlimited



DTIC
ELECTE
MAR 14 1991
S B D

CISD-90/02 COMPUTER CENTER REFERENCE MANUAL, VOLUME 2

91 3 13 064

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION/AVAILABILITY OF REPORT		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			Approved for Public Release; Distribution Unlimited		
4. PERFORMING ORGANIZATION REPORT NUMBER(S) CISD-90/02					
5. MONITORING ORGANIZATION REPORT NUMBER(S)					
6a. NAME OF PERFORMING ORGANIZATION DTRC		6b. OFFICE SYMBOL (If applicable) 3511		7a. NAME OF MONITORING ORGANIZATION	
6c. ADDRESS (City, State, and ZIP Code) Bethesda, MD 20084-5000		7b. ADDRESS (City, State, and ZIP Code)			
8a. NAME OF FUNDING/SPONSORING ORGANIZATION		8b. OFFICE SYMBOL (If applicable)		9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c. ADDRESS (City, State, and ZIP Code)		10. SOURCE OF FUNDING NUMBERS			
		PROGRAM ELEMENT NO.		PROJECT NO.	TASK NO.
				WORK UNIT ACCESSION NO.	
11. TITLE (Include Security Classification) Computer Center Reference Manual, Volume 2					
12. PERSONAL AUTHOR(S) David V. Sommer, Sharon E. Good					
13a. TYPE OF REPORT Final		13b. TIME COVERED FROM 093090 TO indef		14. DATE OF REPORT (Year, Month, Day) 90/09/30	
15. PAGE COUNT 487					
16. SUPPLEMENTARY NOTATION					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP			
			CDC NOS Control statements Programming		
			CDC NOS/VE Hardware		
			Computer Interactive		
19. ABSTRACT (Continue on reverse if necessary and identify by block number) The Computer Center in the Computer and Information Services Department of the David Taylor Research Center has installed an Integrated Supercomputer Network. This manual provides an introduction to the Network. Some information has been distilled from many individual documents and augmented to reflect usage at DTRC. Control statement examples and descriptions of hardware and software are included, as is information on moving files among the CDC CYBER 860 (with the Mass Storage System), the DEC VAXcluster, the secure DEC VAX, and the CRAY X-MP, creating and executing batch jobs, and using the interactive systems. Volume 1 describes the CRAY X-MP, the Mass Storage System and the DEC VAXes. Volume 2 describes the CDC CYBER 860.					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE INDIVIDUAL David V. Sommer			22b. TELEPHONE (Include Area Code) (301) 267-3343		22c. OFFICE SYMBOL 3511

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

18 (continued)

Software Documentation

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

David Taylor Research Center
Bethesda, Maryland 2084-5000

* *
* *
* Computer Center *
* Reference Manual *
* Volume 2: CDC *
* *
* *

by
David V. Sommer
Sharon E. Good

Scientific and Engineering User Support Branch
Code 3511

	Carderock	Annapolis
Phone	(301) 227-1907	(301) 267-3343
Autovon	287-1907	281-3343

For recorded message on computer status (301) 227-3043

Questions and requests for more detailed information
should be directed to Code 3511, Bldg. 17, Rm. 226

Computer and Information Systems Department
Departmental Report

September 1990

CISD-90/02

..

Through Revision 0 (Sept 1990)

..

*** Revision Record ***

Revision	Description
0 (Sep 90)	Original printing.

Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

*** List of Effective Pages ***

Pages	Rev	Pages	Rev
Title	0		
i thru vii	0		
8-1-1 thru 8-1-6	0		
8-2-1 thru 8-2-15	0		
8-3-1	0		
8-4-1 thru 8-4-13	0		
8-5-1 thru 8-5-9	0		
8-6-1 thru 8-6-4	0		
8-7-1 thru 8-7-15	0		
8-8-1 thru 8-8-3	0		
8-9-1	0		
8-10-1	0		
9-1-1 thru 9-1-10	0		
9-2-1 thru 9-2-12	0		
9-3-1 thru 9-3-7	0		
9-4-1 thru 9-4-6	0		
9-5-1 thru 9-5-6	0		
9-6-1 thru 9-6-8	0		
9-7-1 thru 9-7-2	0		
H-1 thru H-176	0		
I-1 thru I-138	0		
G1-1	0		
Index-1 thru Index-38	0		

Contents

Volume 2

Preface

Revision Record	i
List of Effective Pages	ii

8	CDC CYBER 180/860 -- NOS/VE	8-1-1
	NOS/VE Version 1.5.1	8-1-1
	Accessing NOS/VE on the CDC CYBER 860	8-1-2
	Terminal Keys	8-1-3
	Permanent Files	8-1-4
	Batch Jobs	8-1-5
	Following Batch Job Progress	8-1-6
	Killing Batch Jobs	8-1-6
	Batch Job Classes	8-1-6
	NOS/VE SCL Commands	8-2-1
	Flow Control	8-2-1
	Job Control	8-2-1
	Interactive	8-2-4
	Terminal Control	8-2-4
	Interactive Status	8-2-5
	Job Processing	8-2-5
	File Management	8-2-7
	Permanent File	8-2-8
	Mass Storage System File	8-2-9
	Cray	8-2-10
	Load/Dump Memory	8-2-10
	Tape Management	8-2-10
	Procedures	8-2-11
	System Utilities	8-2-11
	Library Maintenance	8-2-12
	Programming Languages	8-2-13
	Loader and Loader-related Control Statements	8-2-13
	Combination	8-2-13
	Variables	8-2-13
	Miscellaneous	8-2-14
	Editing Files	8-3-1
	Message & Help Libraries	8-4-1
	The CREMM Command	8-4-1
	CREMM Subcommands	8-4-2
	The Message Text	8-4-7
	Parameter Substitution	8-4-7
	Horizontal Positioning	8-4-7
	Miscellaneous	8-4-7
	Status Record Fields	8-4-8

Sample Helps	8-4-9
Procedures and Functions	8-5-1
Procedure and Function Commands	8-5-1
Procedure and Function Libraries	8-5-2
Subcommands	8-5-2
DTRC Procedure / Function Library	8-5-3
Sample Procedures	8-5-5
Sample Function	8-5-8
Source Libraries	8-6-1
SCU Subcommands	8-6-1
NOS/VE Source Library Commands	8-6-3
Embedded SCU Directives	8-6-3
Other Subcommands	8-6-4
Object Libraries	8-7-1
Subcommands	8-7-1
DTRC Object Libraries	8-7-14
Examples	8-7-14
Program/Command/Procedure Execution	8-8-1
Command Lists	8-8-1
Search Modes	8-8-2
Examples	8-8-3
On-line Manuals	8-9-1
Other Software	8-10-1
Accessing Other Software	8-10-1
.NSYS	8-10-1
9 CDC CYBER 180/860 -- NOS	9-1-1
NOS Version 2.7.1	9-1-1
Accessing NOS on the CDC 860	9-1-2
Terminal Keys	9-1-3
Direct versus Indirect Files	9-1-6
Batch Jobs	9-1-7
Killing Batch Jobs	9-1-8
Batch Job Classes	9-1-8
Accessing Other Networks	9-1-9
Connecting to Another System	9-1-9
Transferring Files	9-1-9
Some FTP Commands	9-1-10
NOS CCL Commands	9-2-1
Flow Control	9-2-1
Job Control	9-2-2
Interactive	9-2-4
Terminal Control	9-2-4
Subsystem Selection	9-2-4
Interactive Status	9-2-4
Job Processing	9-2-5
File Management	9-2-6
Permanent File	9-2-8

Dump Memory	9-2-3
Tape Management	9-2-9
Checkpoint/Restart	9-2-9
Procedures	9-2-10
System Utilities	9-2-10
Library Maintenance	9-2-10
Programming Languages	9-2-10
Loader and Loader-related Control Statements	9-2-11
Combination	9-2-11
Miscellaneous	9-2-12
 Procedures	 9-3-1
Procedure Directives	9-3-1
DTRC Procedure Library	9-3-5
Sample Procedure	9-3-7
 Program Libraries	 9-4-1
UPDATE	9-4-1
UPDATE Directives	9-4-1
DECK and COMDECK	9-4-1
Compile File	9-4-2
Modification	9-4-2
File Manipulation	9-4-3
Input Stream Directives	9-4-3
Special	9-4-4
Examples	9-4-5
 Object Libraries	 9-5-1
LIBEDIT Directives	9-5-1
DTRC Object Libraries	9-5-5
Examples	9-5-5
 Loader	 9-6-1
Types of Loading	9-6-1
Loader Control Statements	9-6-3
Segmentation	9-6-4
SEGLOAD Directives	9-6-4
Sample Tree Diagram	9-6-5
Segmentation Cautions	9-6-6
Compile, Load and Catalog Absolute Program	9-6-7
Simple Load	9-6-7
SEGLOAD	9-6-7
Interactive Simple Execution	9-6-7
Object Program Execution	9-6-8
COBOL5	9-6-8
Fortran	9-6-8
 Other Software	 9-7-1
Accessing Other Software	9-7-1
UN=APLLIB	9-7-1
UN=LIBRARY	9-7-1
UN=NSYS	9-7-2

Value Descriptors	H-2
Condition Handling	H-3
Some Common Parameters	H-4
Summary of CDC NOS/VE SCL Commands	H-9

I Appendix I	I-1
CDC NOS JCL Commands	I-1
Strings	I-1
Some Common Parameters	I-2
Summary of CDC NOS JCL Commands	I-4
Interactive Cray Facility Commands	I-134

G1 Glossary	G1-1

Index	Index-1

Abstract

The Computer Center in the Computer and Information Services Department of the David Taylor Research Center has installed an Integrated Supercomputer Network. This manual provides an introduction to the Network. Some information has been distilled from many individual documents and augmented to reflect usage at DTRC. Control statement examples and descriptions of hardware and software are included, as is information on moving files among the CDC CYBER 860A (with the Mass Storage System), the DEC VAXcluster, the secure DEC VAX, and the CRAY X-MP, creating and executing batch jobs, and using the interactive systems. Volume 1 describes the CRAY X-MP, the Mass Storage System, and the DEC VAXes. Volume 2 describes the CDC CYBER 860A.

Administrative Information

The work described in this report was performed in the Scientific and Engineering Support Branch (3511) of the Computer and Information Services Department, David Taylor Research Center, under the sponsorship of the DTRC Computer Center (351).

***** CDC CYBER 180/860 -- NOS/VE *****

The Control Data Corporation (CDC) CYBER 180 model 860 when operating under NOS/VE has a single central processing unit (CPU) and 1,048,576 64-bit words (8 Mbytes) of physical memory. Each user job has a maximum virtual address space of 4096 segments of up to 4 Gbytes (4 billion bytes) per segment.

The CPU has 32 registers for operating on information: 16 address (A), and 16 operand (X) registers. The CYBER 860 has a buffer of 12 central memory (CM) words of instructions, called an instruction stack, and a 2048-word, high-speed cache memory for operands.

All system input and output (I/O) is handled by 25 small, independent computers called peripheral processors (PPs). Each PP has 4096 12-bit words of memory, and is able to access all 8 megabytes of CM without help from the CYBER 860 CPU.

Connected to this bank of PPs are 28 normal and 5 concurrent I/O channels which provide data and control signal paths to the CYBER 860 peripheral equipment. System I/O consists of data transfers between CM and the peripheral equipment under control of the PPs via these channels. This I/O scheme allows the CPU to dedicate itself to computation, providing high performance.

*** NOS/VE Version 1.5.1 ***

One operating system for the CDC CYBER 860 at DTRC is called the Network Operating System / Virtual Environment, version 1.5.1 (NOS/VE 1.5 - level 739) and differs only slightly from the standard NOS/VE system. The interactive subsystem, when entered via NOS, is called VEIAF (NOS/VE InterActive Facility). Medium-speed remote batch terminals are accessed via NOS.

Permanent files (user programs and data files retained for frequent use) reside on model 895 disk drives and the Mass Storage System. User files, if not specifically requested on a tape, will be assigned to available disk areas.

NOS/VE operates in a "dual-state" with NOS (see chapter 9).

*** Accessing NOS/VE on the CDC CYBER 860 ***

To access NOS/VE on the CDC CYBER 860:

- . dial (301) 227-5200 <-- this will connect you with DTNET
 at 1200 baud (see page 1-1-4 for
 higher speeds)
- . press the RETURN key until it displays the DTNET> prompt
- . enter "connect cdc860" (or "c cdc860") to connect to NOS
- . in response to the Family: prompt, enter either
 - . ,xxxx,pw,VEIAF (xxxx is your User Initials,
 pw is your login password
 VEIAF is the NOS/VE InterActive Facility)
 - or-
 - . RETURN, then the rest of the information one item at a time
 as prompted
- . in response to the Enter Account Number: prompt, press RETURN
- . in response to the Enter Project Number: enter a dollar sign
(\$) followed by your Job Order Number
- . when you receive the "/" prompt, you are in NOS/VE

*** Terminal Keys ***

NOS/VE supports screen formatting for most display terminals. Many commands use a full-screen mode when the interaction style is SCREEN. When these commands show function keys, they are shown for the terminal mode you have defined (default is CDC Viking 721 terminal).

"CHANGE_TERMINAL_ATTRIBUTE TERMINAL_MODE=DEC_VT100_GOLD" before EDIT_FILE defines the following keypad:

PF1 UNMARK mark	PF2 MRK BX mrk ch	PF3 LOC NXT locate	PF4 WIDTH loc all
7 DEL CH ins ch	8 DEL LIN ins lin	9 LIN UP move	- JOIN break
4 COPY help	5 undo	6 EXIT quit	, CLR EL skp el
1 FIRST bkw	2 LAST fwd	3 HOME back	ENTER
0 <gold>		.	REFRSH middle

After pressing one or more of the keypad keys, the RETURN key must be used to perform the requested functions.

The arrow keys may be used to position the cursor.

*** Permanent Files ***

Unlike NOS, NOS/VE supports only one type of file.

Disk space is allocated and charged by PRU (physical record unit) with one PRU holding 64 words (512 8-bit characters).

*** Batch Jobs ***

A batch job is created and submitted to the Input Queue from an interactive session. This may be done interactively, or the batch job may be put into a file and the file executed (putting the job into the input queue). Two commands are available for submitting batch jobs: JOB (creates the job and submits it) and SUBMIT_JOB (submits an existing file containing a batch job). SUBMIT_JOB requires that the file contain a LOGIN statement as the first statement of the job. Since the LOGIN statement contains your password, SUBMIT_JOB is not recommended and should be used only if you are executing as another user.

1) Interactively:

```
/job job_name=compile
job/fortran i=$user.myprog l=list_file
job/print_file list_file
job/lgo
job/jobend
/
```

-or-

```
/Collect_Text $local.submit_file
ct? login login_user=xxxx ..
ct? password=pw
ct? fortran i=$user.myprog l=list_file
ct? print_file list_file
ct? lgo
ct? logout
ct? **
/Submit_Job $local.submit_file
```

2) From a file:

File \$user.myjob contains:

```
job job_name=compile
fortran i=$user.myprog l=list_file
print_file list_file
lgo
jobend
```

To submit the job file:

```
/Include_File $user.myjob
/
```

-or-

File \$user.mysjob contains:

```
login login_user=xxxx password=pw
fortran i=$user.myprog l=list_file
```

```
print_file list_file
lgo
logout
```

To submit the job file:

```
/Submit_Job $user.mysjob
/
```

**** Following Batch Job Progress ****

The DISPLAY_JOB_STATUS (disjs) command will display information about your batch job. By default, all information is shown. See Appendix H for available parameters.

**** Killing Batch Jobs ****

The TERMINATE_JOB (terj) command is used to terminate a job. See Appendix H for available parameters.

**** Batch Job Classes ****

Batch jobs in NOS/VE have only one service classes or priority. SECURE processing is not available on the CYBER 860.

For the current rates:

```
On NOS/VE:  RATES
             -or-
             RATES,file    <-- puts a copy of the rates into a file

On NOS:     BEGIN,RATES

On VMS:     HELP  RATES
```

***** NOS/VE SCL Commands *****

The NOS/VE System Control Language (SCL) statements are grouped by function in this section. See Appendix H for a description of the syntax for each command. (DTRC) indicates a command or program added at DTRC.

*** Flow Control ***

" Comment.

BLOCK Group a sequence of statements into a block.

CONTINUE Exit the current WHEN statement.

CYCLE Execute the next iteration, if any, of a repetitive command.

EXIT Transfer control out of a structured statement, command or function procedure, or command utility.

FOR Control repetition of a list of statements.

IF Conditional execution of one or more statement lists.

INCLUDE_COMMAND

Include a string containing SCL statements into the command stream.

INCLUDE_FILE

Insert a text file of SCL commands into the command stream.

INCLUDE_LINE

Insert a string containing SCL statements into the command stream.

LOOP Repeat forever a statement list.

REPEAT Conditionally repeat a statement list.

*** Job Control ***

BLOCK Group a sequence of statements into a block.

CANCEL Cancel the most recently selected conditions.

CAUSE Cause a condition to occur.

CHANGE_COMMAND_SEARCH_MODE

Change the command list search mode.

CHANGE_INPUT_ATTRIBUTE

Change an input job's attributes -- resubmit a job.

CHANGE_JOB_ATTRIBUTE

Change the current job's attributes.

CHANGE_JOB_LIMIT

Change a job's resource limits.

CHANGE_LOGIN_PASSWORD

Set or change your login/batch password.

CHANGE_MESSAGE_LEVEL

Select the NOS/VE message display format.

CHANGE_OUTPUT_ATTRIBUTE

Change an output file's attributes.

CHANGE_SCL_OPTION

Change the options provided by the SCL interpreter.

CHANGE_UTILITY_ATTRIBUTES

Change the attributes of a utility command (UTILITY/
UTILITYEND).

COMMAND Declare an entry in utility command table.**CREATE_COMMAND_LIST_ENTRY**

Add entries to the beginning of end of the command list.

DEFINE_PRIMARY_TASK

Designate the requesting task as the primary task for the job.

DELETE_COMMAND_LIST_ENTRY

Delete entries from the command list.

DETACH_JOB

Explicitly disconnect your terminal from the current job.

DISPLAY_JOB_ATTRIBUTE

Display the attributes of your current job.

DISPLAY_LOG

Display the job log for the requesting job.

DISPLAY_MESSAGE

Display a message in one or more logs.

DISPLAY_OUTPUT_ATTRIBUTE

Display attributes of an output file.

DISPLAY_OUTPUT_HISTORY

Display job log entries for output files.

DISPLAY_OUTPUT_STATUS

Display the status of a file in the output queue.

DISPLAY_PROGRAM_ATTRIBUTES

Display the current job library list and default execution option values.

DISPLAY_SCL_OPTIONS

Display the options set by CHANGE_SCL_OPTIONS.

DISPLAY_TASK_STATUS

Display the status of one or more tasks.

EXECUTE_COMMAND

Execute a single command asynchronously in a new task.

EXECUTE_INTERSTATE_COMMAND

Execute a NOS command.

JOB Generate and submit a batch job.

RESUME_COMMAND

Resume a job activity interrupted by a pause break.

SET_SENSE_SWITCH

Sets sense switches on or off.

STATUS_CRAY

(DTRC) Display the status of Cray jobs.

SUBMIT_CRAY_JOB

(DTRC) Submit a job to the Cray.

SUBMIT_DETACHED_JOB

Start a disconnected interactive job.

SUBMIT_JOB

Submit a NOS/VE batch job.

TASK

Delimit a sequence of SCL statements to be as a synchronous or an asynchronous task.

WAIT

Suspend command processing for a time period or until an event.

WHEN

Delimit SCL statements to be executed when a specified condition occurs.

WHILE

Conditionally repeat a statement list.

*** Interactive ***

** Terminal Control **

CHANGE_CONNECTION_ATTRIBUTES

Change the terminal file's connection attributes.

CHANGE_INTERACTION_STYLE

Change from line mode to screen mode or vice versa.

CHANGE_LINK_ATTRIBUTES

Change individual link attributes for communication between dual-state partners.

CHANGE_TERMINAL_ATTRIBUTES

Define and change an interactive terminal's attributes.

CHANGE_TERM_CONN_DEFAULTS

Change a terminal's default connection attributes.

CREATE_INTERSTATE_CONNECTION

Establish a NOS batch control point on a dual-state system.

DEFINE_TERMINAL

Compile your terminal definition file.

DELETE_INTERSTATE_CONNECTION

Terminate the interstate connection.

DESIGN_SCREEN

Enter the Screen Design Facility (SDF).

DISPLAY_CONNECTION_ATTRIBUTES

Display the terminal file's connection attributes.

DISPLAY_LINK_ATTRIBUTES

Display individual link attribute values.

DISPLAY_TERMINAL_ATTRIBUTES

Display interactive terminal attribute values.

DISPLAY_TERM_CONN_DEFAULTS

Display the defaults for terminal connection attributes.

INITIALIZE_TERMINAL

Change the terminal settings.

REQUEST_TERMINAL

Associate a file with a terminal in an interactive job.

SET_LINK_ATTRIBUTES

Use CHANGE_LINK_ATTRIBUTES.

** Interactive Status **

%A Same as DISPLAY_ACTIVE_TASKS .

%D Immediately detach a terminal job from the terminal (same as DETACH_JOB).

%J Immediate detailed job status (same as DISPLAY_JOB_STATUS).

%S Immediate abbreviated job status.

** Job Processing **

DISPLAY_ACTIVE_TASKS
Display tasks executing within the current job.

DISPLAY_COMMAND_INFORMATION
Display information about a command.

DISPLAY_COMMAND_LIST_ENTRY
Display information about one or more entries in a command list.

DISPLAY_COMMAND_LIST
Display information about the command list.

DISPLAY_JOB_ATTRIBUTE_DEFAULT
Display the current defaults for the job attributes.

DISPLAY_JOB_LIMIT
Display your job's limits.

DISPLAY_JOB_STATUS
Display current status of queued or executing jobs.

DISPLAY_TOPICS_FILE
Open a Topics online manual.

ECHO (DTRC) Control echoing of SCL statements to a file.

EXPLAIN Open an on-line manual.

EXPLAIN_MESSAGE
Request a description of a system message.

GENERATE_COMMAND_TABLE
Generate command and function tables for a command environment.

HELP Open an on-line manual.

KERMIT Begin a KERMIT file transfer session.

LOGIN Provide access to NOS/VE batch services.

LOGOUT Terminate a batch or interactive job.

MANAGE_JOB

Begin a utility that manages the selection and control of one or more jobs.

MANAGE_OUTPUT

Begin a utility that manages the selection and control of one or more output files.

PUSH_COMMANDS

Move the command list entry containing the procedure that call this statement to the top of the command list.

ROUTE_JOB

(for input from cards, or a PC or terminal supporting HASP)
Name a job and the output destination.

SET_COMMAND_LIST

Add to and/or delete from the current command list, and/or alter the state of the search mode indicator.

SET_DEBUG_LIST

Add or delete debug object libraries to/from the job debug library list.

SET_DEBUG_RING

Specify the ring in which Debug is to execute.

SET_DEFAULT_LOGIN_PROJECT

(DTRC) Set the default login project number (job order number at DTRC) so that it need not be entered at future logins.

TERMINATE_COMMAND

Terminate the program interrupted by a pause break (%1).

TERMINATE_JOB

Terminate queued or executiong jobs.

TERMINATE_TASK

Delete asynchronous tasks (and all tasks generated by those tasks).

XMODEM_RECEIVE

Receive a file send from a PC using XMODEM.

XMODEM_SEND

Send a file to a PC using XMODEM.

*** File Management ***

CHANGE_WORKING_CATALOG

Establish the default working catalog.

COLLECT_TEXT

Copy lines of text from the command list to a specified file.

COMPARE_FILE

A binary comparison of two files.

COPY_FILE

Copy a file.

CREATE_FILE_CONNECTION

Create a connection between two files so that any data access request against the subject file is pass to the target file.

DELETE_FILE_CONNECTION

Delete the connection between files.

DISPLAY_FILE

Display a file in hexadecimal and/or ASCII.

DISPLAY_FILE_CONNECTIONS

Display the names of the files currently connected to the subject files.

DISPLAY_INPUT_ATTRIBUTE

Display the input attributes of a queued or processing job.

ENTER_FILE_MANAGER

Begin a File Manager utility session.

FILE_MANAGEMENT_UTILITY

Execute the File Management Utility.

GET_FILE

Copy a NOS file to NOS/VE.

MERGE

Merge sorted records from one or more files and write them in sorted order to a single output file.

OPEN_FILE_MIGRATION_AID

Begin the file migration environment.

PRINT_FILE

Print one or more files.

PUT_LINE

Write lines to a file.

REPLACE_FILE

Copy a NOS/VE file to NOS direct file, replacing any existing file.

REWIND_FILE

Position a file at BOI (beginning-of-information).

SHOW_FILE

Display a file in a pop up window that provides editing options.

SORT

Sort records from one or more files and write them in sorted order to a single output file.

TERMINATE_OUTPUT

Delete output files.

*** Permanent File ***

BACKUP_PERMANENT_FILES

Begin a utility session to back up permanent files and catalogs.

CHANGE_CATALOG_CONTENTS

Delete damage condition notices from files in the specified catalog and all its subcatalogs, and delete catalog entries for files with no cycle data.

CHANGE_CATALOG_ENTRY

Change a file's catalog entry item(s).

CHANGE_FILE_ATTRIBUTES

Change an existing file's attributes.

CREATE_CATALOG

Create a new catalog.

CREATE_CATALOG_PERMIT

Grant or deny permission to use a catalog.

CREATE_FILE

Create a new file (or cycle of a file) and attach it to the job.

CREATE_FILE_PERMIT

Establish or modify an access control entry for a file.

DELETE_ALL_CYCLES

(DTRC) Delete all cycles of one or more files.

DELETE_CATALOG

Delete a catalog and, optionally, its contents.

DELETE_CATALOG_PERMIT

Delete an access control entry for a catalog.

DELETE_FILE

Delete a file or cycle of a file.

DELETE_FILE_PERMIT

Delete an access control entry.

DETACH_FILE

Detach one or more files from a job.

DISPLAY_CATALOG

Audit files and catalogs in a specific catalog or about the access control list at the catalog level.

DISPLAY_CATALOG_ENTRY

Audit a file, its usage, or its access control list.

DISPLAY_FILE_ATTRIBUTES

Display selected attributes of one or more files.

DISPLAY_WORKING_CATALOG

Display your working catalog.

EDIT_CATALOG

Begin an EDIT_CATALOG utility session.

PURGE (DTRC) Delete cycles of one or more files keeping the specified number of high cycles.

RESTORE_PERMANENT_FILES

Begin a RESTORE_PERMANENT_FILES utility session.

SET_FILE_ATTRIBUTES

Set the attributes of a file.

*** Mass Storage System File ***

MSACCES (DTRC) Access the Mass Storage System (MSS).

MSAUDIT (DTRC) Obtain a sorted audit of your MSS files.

MSCATLIST (DTRC) The NOS CATLIST command.

MSCHANG (DTRC) Change the attributes of a Mass Storage System file.

MSFETCH (DTRC) Fetch a file from the Mass Storage System.

MSFILES (DTRC) Obtain a sorted list of indirect and direct Mass Storage System file names.

MSPASSW (DTRC) Change your Mass Storage System access password.

MSPURGE (DTRC) Purge a file on the Mass Storage System.

MSSTORE (DTRC) Store a file on the Mass Storage System.

*** Cray ***

STATUS_CRAY
(DTRC) Display the status of Cray jobs.

SUBMIT_CRAY_JOB
(DTRC) Submit a job to the Cray.

*** Load/Dump Memory ***

*** Tape Management ***

CHANGE_170_REQUEST
Change the 170 tape file description in a temporary NOS/VE file
formed in a preceding CREATE_170_REQUEST command.

CHANGE_BACKUP_LABEL_TYPE
Change the job default label type for backup to tape.

CREATE_170_REQUEST
Create a NOS/VE temporary file to be associated with a 170
(NOS) tape file.

CREATE_VAX_REQUEST
Create a NOS/VE temporary file to be associated with a VAX
tape file used for future references to the VAX tape file.

DISPLAY_BACKUP_LABEL_TAPE
Display the current job default label type for a permanent
file backup file on tape.

DISPLAY_TAPE_LABEL_ATTRIBUTES
Display the current magnetic tape label attributes.

RELEASE_RESOURCE
Release tape reservations.

REQUEST_MAGNETIC_TAPE
Associate a file with a magnetic tape.

RESERVE_RESOURCE
Specify the number of tape units to be reserved.

RESTORE_FOREIGN_FILES
Begin a RESTORE_FOREIGN_FILES utility session to migrate
NOS tape files to NOS/VE.

SKIP_TAPE_MARK

Position a tape backward or forward.

*** Procedures ***

PROCEDURE

First statement of an SCL procedure defining the procedure's names, attributes, and parameters.

*** System Utilities ***

ANALYZE_OBJECT_LIBRARY

Begin an ANALYZE_OBJECT_LIBRARY utility session.

ANALYZE_PROGRAM_DYNAMICS

Measure program execution characteristics and restructure program as a single load module.

ANY_MAIL (DTRC) Check for mail.**CHANGE_UTILITY_ATTRIBUTES**

Change the attributes of a utility command (UTILITY/UTILITYEND).

CHECK_FOR_MAIL

Check for unread mail.

COMPRESS_MAIL_DATABASE

Create a new copy of your mailbox file (MVFS\$ELECTRONIC_MAIL_DATABASE) in your master catalog, with unused space removed and any data faults corrected.

CONVERT_UPDATE_TO_SCU

Convert an UPDATE source library to SCU format.

CREATE_MANUAL

Read a source file of directives and text to create an on-line manual for use by EXPLAIN.

CREATE_TOPICS_FILE

Create a Topics manual from a text file using binding.

DECRYPT_FILE

Decrypt a file encrypted using ENCRYPT_FILE.

DISPLAY_FUNCTION_INFORMATION

Display information about a function.

EDIT_FILE

Begin a file editor (EDIT_FILE utility) session.

EMAIL

Begin a Mail/VE session.

FUNCTION

First statement of an SCL function defining the function's names, attributes, and parameters.

FUNCTION

(UTILITY subcommand) Declare an entry in a utility function table.

MANAGE_FORM

Begin a MANAGE_FORMS utility session.

MEASURE_PROGRAM_EXECUTION

Begin a program measurement utility session.

RESTORE_LOG

Begin a RESTORE_LOG utility session.

TABLEND

(UTILITY subcommand) End the collection of utility command table definitions.

UTILITY

Delimit the definition and execution of a command utility.

*** Library Maintenance ***

COMPARE_OBJECT_LIBRARY

Compare two object libraries or two object files.

CONVERT_UPDATE_TO_SCU

Convert an UPDATE source library to SCU format.

CREATE_OBJECT_LIBRARY

Begin a CREATE_OBJECT_LIBRARY utility session.

CREATE_PROGRAM_PROFILE

Generate a program profile.

DISPLAY_OBJECT_LIBRARY

Display information about an object library, object file, or procedure file.

DISPLAY_OBJECT_TEXT

Display object records of an object library or file.

EDIT_DECK

(SCU subcommand) Open a deck in the working library for editing while maintaining your current position in other decks.

EXPAND_SOURCE_FILE

Expand a text file as though it were a deck in an SCU library.

EXTRACT_SOURCE_LIBRARY

Extract decks from a base library to use as a separate library.

FORMAT_SCL_PROCEDURE

Format a file of SCL commands to improve readability.

GENERATE_SCU_EDIT_COMMANDS

Compare a deck and a text source file and generate editing commands to make the deck match the source file.

SOURCE_CODE_UTILITY

Begin an SCU utility session.

*** Programming Languages ***

BASIC Compile a BASIC source program.

COBOL Compile a COBOL source program.

FORTTRAN Compile a Fortran program.

FTN (same as FORTTRAN)

PASCAL Compile a PASCAL program.

*** Loader and Loader-related Control Statements ***

EXECUTE_TASK

Execute a program.

\$LOCAL.LGO Load and execute the default compiler binary output file.

name Load and execute binary program or procedure in your command list.

pathname Load and execute binary program or procedure in the specified file

*** Combination ***

FLR (DTRC) Compile FTN5, load with optional libraries, and run.

*** Variables ***

CREATE_DEFAULT_VARIABLE

Create an environment variable to be used as a default.

CREATE_FILE_VARIABLE

Create an environment variable of type FILE.

DELETE_VARIABLE

Delete variables accessible from the current block.

DISPLAY_VALUE

Display the value of an expression.

DISPLAY_VARIABLE_LIST

Display variables accessible to the current block.

GET_LINE

Read a line from a file into a string variable.

POP

Delete the current version of an environment object or variable and restore to its former value.

PUSH

Temporarily change environment objects or variables in an SCL procedure.

SHOW_VALUE

Display a value in a pop up window.

TYPE

Create one or more user-defined data types.

VAR

Create one or more variables.

*** Miscellaneous ***

AUXPRINT (DTRC) Print a file on your auxiliary printer (the one attached to your terminal).

ENCRYPT_FILE

Encrypt a file using the National Bureau of Standards Data Encryption Standard algorithm.

ENTER_PROGRAMMING_ENVIRONMENT

Access the Programming Environment.

FICHE

(DTRC) Print a NOS/VE file on microfiche (COM).

NEWS

(DTRC) Display the current news file or copy it into a user file.

NEW_ROUTINES

(DTRC) Display a description of recently-added routines or copy the information into a user file.

OLD_NEWS

(DTRC) Display the old news file or copy it into a user file.

RATES

(DTRC) Display the current Computer Center rates or put them into a user file.

SELECT_USER_MENU

Begin the NOS/VE user menu system.

SEND_OPERATOR_MESSAGE

Send a message to an operator.

WHO (DTRC) List the executing jobs, including users currently
logged in.

WHOAMI (DTRC) Display the executing user's username and job order
number.

XEROX (DTRC) Print a NOS/VE file on the Xerox 8700.

***** Editing Files *****

Files are created and edited through the EDIT_FILE utility. This is a screen or line mode editor depending on the interaction style set for your session. It is in line mode by default. There are various commands available that will allow you to make changes to the text you are working with. The keypad layout for the DEC_VT100_GOLD terminal definition is on page 8-1-3.

The zero key followed by the number key on the keypad performs the functions indicated by capital letters. The remaining functions are performed by pressing the keypad numeric key by itself. The cursor keys will position the cursor where you want. EDIT_FILE is like FSE in that the 'home line' is where line mode commands are entered. Also, this is where you enter the 'QUIT' command. Unlike other editors, QUIT exits the editor and saves the file (keypad 6 by itself). In order to exit without saving, you must enter QUIT WF=T (or keypad 0 + keypad 6). All commands entered from the keypad must be followed the ENTER key.

***** Message & Help Libraries *****

A message module contains information for use during command entry and error processing. The CREATE_MESSAGE_MODULE utility is used to maintain message modules. When used within CREATE_OBJECT_LIBRARY (see Section 8-6), you can create command helps, parameter helps, and status messages.

*** The CREMM Command ***

CREATE_MESSAGE_MODULE

Begin a CREATE_MESSAGE_MODULE utility session.

Syntax: CREATE_MESSAGE_MODULE -or-
 CREMM
 N or NAME=program_name

 M or MANUAL=program_name
 NL or NATURAL_LANGUAGE=keyword or name
 MO or MERGE_OPTION=keyword
 STATUS=status variable

Parameters: N - the name of the message module in the form
 name\$language
 where name is the name of the procedure
 language is US_ENGLISH

M - the name of the on-line manual which
describes the command or function

NL - the natural language of the message
(default: US\$ENGLISH)

MO - merge options

ab	option	meaning
A	ADD	add to the library
R	REPLACE	replace existing module
C	COMBINE	replace if existing, else add

(default: C)

Subcommand prompt: CMM/

Similar commands: NOS:
 VMS: LIBRARY

Examples: /creol
 COL/cremm
 CMM/

*** CREMM Subcommands ***

The following CREMM subcommands are used to create messages.

CREATE_BRIEF_HELP_MODULE

Create a brief description of a command.

Syntax: CREATE_BRIEF_HELP_MODULE -or-
 CREBHM

 CTU or COLLECT_TEMPLATE_UNTIL=string
 STATUS=status variable

Parameters: CTU - the string which terminates the message
 (default: '**')

See also: CREATE_FULL_HELP_MODULE

Examples: CMM/crebhm
 ? The MSFETCH command fetches a file from
 ? the Mass Storage System.
 ? **
 CMM/

CREATE_FULL_HELP_MESSAGE

Create a full description of a command.

Syntax: CREATE_FULL_HELP_MODULE -or-
 CREFBHM

 CTU or COLLECT_TEMPLATE_UNTIL=string
 STATUS=status variable

Parameters: CTU - the string which terminates the message
 (default: '**')

See also: CREATE_BRIEF_HELP_MODULE

Examples: CMM/crefbhm
 ? The MSFETCH command fetches a file from
 ? the Mass Storage System. You can give it
 ? a different name on NOS/VE or let it have
 ? its NOS name.
 ? **
 CMM/

CREATE_PARAMETER_ASSIST_MESSAGE

Create a message to be displayed when the user enters an incorrect parameter value.

Syntax: CREATE_PARAMETER_ASSIST_MESSAGE -or-
 CREPAM
 N or NAME=name

 CTU or COLLECT_TEMPLATE_UNTIL=string
 STATUS=status variable

Parameters: N - the name of the parameter
 (if the command/procedure has aliases, this
 must be the first name listed in the PDT or
 PROCEDURE statement)

CTU - the string which terminates the message
 (default: '***')

See also: CREATE_PARAMETER_HELP_MESSAGE

Examples: CMM/crepam name=mfn
 ? Enter the name of the MSS file you are
 ? fetching. It must be 1-7 alphanumeric
 ? characters.
 ? **
 CMM/

CREATE_PARAMETER_HELP_MESSAGE

Create a message to be displayed when the user requests help for the specified parameter.

Syntax: CREATE_PARAMETER_HELP_MESSAGE -or-
 CREPHM
 N or NAME=name

 CTU or COLLECT_TEMPLATE_UNTIL=string
 STATUS=status variable

Parameters: N - the name of the parameter
 (if the command/procedure has aliases, this
 must be the first name listed in the PDT or
 PROCEDURE statement)

CTU - the string which terminates the message
 (default: '***')

See also: CREATE_PARAMETER_ASSIST_MESSAGE

Examples: CMM/crephm mfn
 ? The MFN parameter is the name of the MSS file
 ? you are fetching. It must be 1-7 alphanumeric
 ? characters.
 ? **
 CMM/

CREATE_PARAMETER_PROMPT_MESSAGE

Create the prompt message for the specified parameter.

Syntax: **CREATE_PARAMETER_PROMPT_MESSAGE** -or-
 CREPPM
 N or NAME=name

 CTU or COLLECT_TEMPLATE_UNTIL=string
 STATUS=status variable

Parameters: N - the name of the parameter
 (if the command/procedure has aliases, this
 must be the first name listed in the PDT or
 PROCEDURE statement)

CTU - the string which terminates the message
 (default: '**')

See also: **CREATE_PARAMETER_ASSIST_MESSAGE**

Examples: CMM/creppm ..
 CMM../name=mfn
 ? MSS filename?
 ? **
 CMM/

CREATE_STATUS_MESSAGE

Create a status message for the specified status condition code.

Syntax: **CREATE_STATUS_MESSAGE** -or-
 CRESM
 N or NAME=name
 C or CODE=integer

 I or IDENTIFIER=string
 S or SEVERITY=keyword
 CTU or COLLECT_TEMPLATE_UNTIL=string
 STATUS=status variable

Parameters: N - the condition identifier

C - the status condition code
 (range: 10000-19999; numbers above and
 below this range are reserved for
 NOS/VE)

I - 2-character product identifier

S - the severity level of the status condition
 ab option
 -- -----
 NS NON_STANDARD
 D DEPENDENT

I INFORMATIVE
W WARNING
E ERROR
F FATAL
C CATASTROPHIC
(default: ERROR)

CTU - the string which terminates the message
(default: '**')

Examples: Change the message for CLE\$UNKNOWN_COMMAND from
 "<text> is not a command." to "The command <text>
 is not in your command list."

```
/display_value ..  
../$status_code_string(cle$unknown_command)  
CL 790  
/creol  
COL/cremm show  
CMM/cresm name=cle$unknown_command code=790 ..  
CMM/i='CL' s=error  
? The comand +P is not in your command list.  
? **  
CMM/quit  
COL/genl my_messages  
COL/quit  
/create_command_list_entry my_messages ..  
../p=before  
  
/what_am_i_doing  
--ERROR-- The command WHAT_AM_I_DOING is not  
in your command list.  
/
```

END_MESSAGE_MODULE

End the CREMM session.

Syntax: END_MESSAGE_MODULE ~or-
 ENDM ~or-
 QUIT ~or-
 QUI

 CM or CREATE_MODULE=boolean
 STATUS=status variable

Parameters: CM - specify whether message module is to be
 created
 (default: YES)

See also: CREATE_MESSAGE_MODULE

Examples: /creol
 COL/cremm
 CMM/<do your thing>
 CMM/endmm
 COL/quit
 /

*** The Message Text ***

The message text follows each CREATE_*_MESSAGE subcommand and terminates when <ctu> is encountered on a separate line. The text is really a template which is processed into the message that is displayed. Unless otherwise indicated, the text is displayed as a single paragraph with the last non-blank concatenated with the first position of the next line. Thus, if you end with a complete word, you must start the next line with a blank. To put two blanks between sentences, start the sentence on a new line preceded by two blanks. Otherwise, multiple blanks are compressed to a single blank.

Control sequences are used to format the message. A control sequence is indicated by a plus sign (+) and is used for parameter substitution, horizontal positioning, and insertion of portions of the condition record.

** Parameter Substitution **

Control sequence	Description
+Pn	Substitute the nth command parameter (1 ≤ n ≤ 128) (for status messages, the parameters are taken from the text field of the status record).
+P	Substitute the next command parameter.
+Fn	Substitute the nth message parameter and expand it to a filepath.
+F	Substitute the next message parameter.

** Horizontal Positioning **

Control sequence	Description
+N	Start a new line.
+Nn	Start a new line and indent n spaces.
+E	If the current output line becomes too long, start a new line.
+En	Same, but indent n spaces.
+Xn	Put n spaces into the text (default: 1). This is required to insert two or more contiguous spaces.
+Ktext+K	The bracketed text is put onto a single line, if possible.
+Hn	Horizontal tab to column n. If line becomes too long, start a new line with no leading blanks.
+H	Tab to the next default tab (1, 9, 17, 25, ...).

** Miscellaneous **

Control sequence	Description

++	Insert a single plus character. (not needed if the character following the plus is not a control character)
+-	Do nothing -- used to concatenate a message parameter with following text without intervening blanks -- most often used with a number might be interpreted as part of the control sequence, e.g., "+H9+-9) some text".
+R	The remainder of the template is repeated until there are no more +P or +F control sequences. +R may be used only once in a template.

** Status Record Fields **

The following control sequences are used only with status messages:

Control sequence	Description
+C	Insert the condition field in the format "XX n" where XX is the product identifier and n is the numeric part (0-16,777,215)
+I	Insert the product identification part of the condition code
+S	Insert the security level
+T	Insert the entire text field

*** Sample Helps ***

- 1) The procedure Fortran_Load_Run has the following header:

```
PROCEDURE (help_flr) Fortran_Load_Run, flr (
  input,          i : file      = $required
  library,        libraries, l : list of file = $optional
  parameter,      parameters, p : list of file = $optional
  status )
```

Create HELP_FLR:

```
/creol
COL/cremm help_flr$us_english
CMM/ Create_Brief_Help_Message
CMM/This commands compiles, loads with optional libraries,
CMM/ and executes a Fortran program.
CMM/**
CMM/ Create_Full_Help_Message
CMM/This commands compiles, loads, and executes a Fortran program.
CMM/ You may specify one or more object libraries to be searched
CMM/ while loading the program.
CMM/**
CMM/ Create_Parameter_Prompt_Message input
CMM/Input file?
CMM/**
CMM/ Create_Parameter_Help_Message input
CMM/The INPUT parameter specifies the file with the input Fortran
CMM/ source program.
CMM/**
CMM/ Create_Parameter_Assist_Message input
CMM/Enter the name of the file with the input Fortran source
CMM/ program.
CMM/**
CMM/Create_Parameter_Prompt_Message library
CMM/Library file(s)?
CMM/**
CMM/Create_Parameter_Help_Message library
CMM/The LIBRARY parameter specifies the libraries to be used in
CMM/ loading the program.
CMM/+NOIf omitted, no libraries will be used.
CMM/**
CMM/Create_Parameter_Assist_Message library
CMM/Enter the name of the file to receive the output.
CMM/+NOOr, press RETURN to use no libraries.
CMM/**
CMM/Create_Parameter_Prompt_Message parameter
CMM/Parameters?
CMM/**
CMM/Create_Parameter_Help_Message parameter
CMM/The PARAMETER parameter specifies the parameters for the
CMM/ program (to replace files in PROGRAM statement list).
CMM/+NOIf omitted, no parameters will be passed to the program.
CMM/**
CMM/Create_Parameter_Assist_Message parameter
CMM/Enter the parameters for the program.
```

```
CMM/+NOIf omitted, no parameters will be passed to the program.  
CMM/**  
CMM/Quit  
COL/genl my_help_library  
COL/quit  
/
```

Alternatively, the lines above beginning with CMM/ could be put into a file (without "CMM/") and the file (e.g., FLR_HELP) included:

```
/creol  
COL/include_file flr_help  
COL/genl my_help_library  
COL/quit  
/
```

2) The procedure MSFETCH has the following header:

```

procedure (help_msfetch) MSFETCH (
  from, f, mfn      : name = $required
  to, t             : file = $optional
  data_conversion, dc : key B60 B56 A6 A8 D63 D64
                    : keyend = B56
  status )

```

The help for MSFETCH is called HELP_MSFETCH and is in file MSFETCH_HELP:

```

( 1) Create_Message_Module help_msfetch$us_english
( 2) " MSFETCH
( 3) Create_Brief_Help_Message
( 4) This fetches a file from the MSS.
( 5) **
( 6) Create_Full_Help_Message
( 7) This fetches one of your files from the Mass Storage System.
( 8) **
( 9) " FROM
(10) Create_Parameter_Prompt_Message from
(11) MSS Filename?
(12) **
(13) Create_Parameter_Help_Message from
(14) The FROM parameter specifies the MSS file to be fetched.
(15) **
(16) Create_Parameter_Assist_Message from
(17) Enter the name of the MSS file to be fetched.
(18) **
(19) " TO
(20) Create_Parameter_Prompt_Message to
(21) NOS/VE filename?
(22) **
(23) Create_Parameter_Help_Message to
(24) The TO parameter specified the NOS/VE name for the file.
(25) **
(26) Create_Parameter_Assist_Message to
(27) Enter the NOS/VE name for the file.
(28) Or press return to use the MSS filename.
(29) **
(30) " DATA_CONVERSION
(31) Create_Parameter_Prompt_Message data_conversion
(32) DC=
(33) **
(34) Create_Parameter_Help_Message data_conversion
(35) The DATA_CONVERSION (DC) parameter specifies the character
(36) code conversion to be performed during the transfer (B60,
(37) [ B56 ], A6, A8, D63, D64).
(38) **
(39) Create_Parameter_Assist_Message data_conversion
(40) Enter one of the following data conversion codes:
(41) +NOB60 (NOS word --> rightmost 60 NOS/VE bits;
(42) leftmost 4 NOS/VE bits set to 0)
(43) +NOB56 (rightmost 56 NOS bits --> contiguous NOS/VE bits;
(44) leftmost 4 NOS bits ignored (use for NOS/VE object libraries.

```



```

(45) SCU libraries, permanent file backup files))
(46) +NOA6+X2(NOS 6/12 display code --> 7-bit ASCII)
(47) +NOA8+X2(NOS 8/12 display code --> 7-bit ASCII)
(48) +NOD63 (NOS 63-char display code --> 7-bit ASCII)
(49) +NOD64 (NOS 64-char display code --> 7-bit ASCII)
(50) +N00r, press RETURN for B56.
(51) **
(52) Quit

```

where

```

1      - specifies the name and language
2      - a comment to help locate things while editing
3-5    - the brief help message
6-8    - the full help message
9      - a comment to help locate the FROM parameter helps
10-12  - the prompt message for the FROM parameter
13-15  - the help message for the FROM parameter
16-18  - the assist message for the FROM parameter
19-29  - like 9-18 for the TO parameter
30-51  - like 9-18 for the DATA_CONVERSION parameter
52     - terminate the CREMM

```

Create HELP_MSFETCH:

```

/creol
COL/addm my_help_library    <-- if MY_HELP_LIBRARY already
                             has modules in it
COL/include_file msfetch_help <-- replaces or adds HELP_MSFETCH
COL/delcde my_help_library  <-- remove the library from your
                             command list
COL/genl my_help_library.$next<-- creates a new cycle of your
                             library ($NEXT is needed if the
                             library is in your command list)

COL/quit
/crecle my_help_library    <-- restore to your command list
/

```

3) SUBMIT_CRAY_JOB displays the message

Job sent to NOS Cray Station at hh:mm:ss

when the job has been sent. The status message module which does this is:

```
1 - CREATE_MESSAGE_MODULE NAME=CRM$MESSAGE_TEMPLATE_MODULE ..
2 - NATURAL_LANGUAGE=US_ENGLISH

3 - CREATE_STATUS_MESSAGE NAME=CRE$JOB_SENT_TO_NOS_CRAY_STA ..
4 - IDENTIFIER='CR' CODE=10004 SEVERITY=INFORMATIVE ..
5 - COLLECT_TEMPLATE_UNTIL='**'
6 - Job sent to NOS Cray Station at +P
7 - **

8 - END_MESSAGE_MODULE CREATE_MODULE=YES
```

The procedure SUBCJ terminates with

```
EXIT procedure WITH $status(false, 'CR', 10004, $time(hms))
```

The +P in line 6 of the module is replaced by the value of \$time(hms) in the \$STATUS function.

***** Procedures and Functions *****

A procedure or function is a group of control statements separate from the job control statement file. Calling a procedure provides a simplified way to process that group of control statements. A procedure may be called by a job repeatedly, by another procedure, or by itself. A function returns a value to the caller.

*** Procedure and Function Commands ***

A procedure begins with a PROCEDURE statement, which defines the procedure's names, attributes, and parameters, and ends with PROCEND.

A function begins with a FUNCTION statement, which defines the function's names, attributes, and parameters, and ends with FUNCEND. Functions names begin with a dollar sign (\$).

All commands, functions, utilities, and statements in the command list are available in procedures and functions. The following commands are used in writing them (see Appendix H for syntax):

COLLECT_TEXT	Read text lines from a file or the command stream into a file.
DELETE_VARIABLE	Delete variables from the current block.
DISPLAY_VALUE	Display the value of an expression.
DISPLAY_VARIABLE_LIST	Display the variables available to the job.
FORMAT_SCL_PROCEDURE	Format a procedure and detect errors.
GET_LINE	Read lines from a file into a string variable.
INCLUDE_COMMAND	Create and execute a command.
INCLUDE_FILE	Execute a file of commands.
INCLUDE_LINE	Execute a string of commands.
PUT_LINE	Write lines from string variables into a file.
TYPE/TYPEND	Create an SCL data type.
VAR/VAREND	Create an SCL variable.
WHEN/WHENEND	Execute statements on specified condition.
WAIT	Suspend processing for a specified time or until an event or set of events occurs.

*** Procedure and Function Libraries ***

CREATE_OBJECT_LIBRARY (creol) is a utility for creating and maintaining object libraries of load modules, function modules, SCL procedure modules, and message modules. CREOL uses a module list while processing its subcommands. Your library is not modified until you generate a new library.

This section discusses SCL procedure and functions libraries. See section 8-6 for object libraries; and 8-3 for message/help libraries.

*** Subcommands ***

The following CREOL subcommands are used to maintain libraries of procedures and functions. They are described in detail in Section 8-6.

ADD_MODULE

Add one or more modules to the module list.

COMBINE_MODULE

Add or replace modules in the module list.

DELETE_MODULE

Delete one or more modules from a module list.

DISPLAY_NEW_LIBRARY

Display the contents of the module list.

GENERATE_LIBRARY

Generate a new object library containing the modules in the module list. It can also write an object file, SCL procedure text file, form source file, form variable file, or CREATE_MESSAGE_MODULE subcommands.

QUIT

End a CREATE_OBJECT_LIBRARY utility session.

REORDER_MODULE

Reorder modules in the module list.

REPLACE_MODULE

Replace modules in the module list.

SET_DISPLAY_OPTION

Change and display the default display options for DISPLAY_NEW_LIBRARY.

*** DTRC Procedure / Function Library ***

Public-access procedure/function library PROCLIB has been added to each user's command list at DTRC.

The following are the public procedures and functions available at DTRC as of the date of this page. For the current list, type "HELP CONTENTS CCRM". See Appendix H for further information on individual procedures and functions.

ANY_MAIL	Check for mail and, optionally, list the letters.
AUXPRINT	Print a file on your auxiliary printer.
DELETE_ALL_CYCLES	Delete all cycles of one or more files.
ECHO	Turn echo file on or off.
FICHE	Route a copy of a local file to the microfiche (COM).
MSACCES	Access the Mass Storage System.
MSAUDIT	Sorted list of MSS files.
MSCATLIST	The complete NOS CATLIST command.
MSCHANG	Change attributes of an MSS file.
MSFETCH	Fetch a file from the MSS.
MSPASSW	Change your MSACCES password.
MSPURGE	Purge an MSS file.
MSSTORE	Store a file on the MSS.
NEWS	Display the current news.
NEW_ROUTINES	Announcement of new routines.
OLD_NEWS	Display the old news.
PURGE	Delete all cycles except the specified number of highest cycles.
RATES	Display the current computer rates.
SET_DEFAULT_LOGIN_PROJECT	Set your Default Login Project Number so it can be omitted in future logins.
STATUS_CRAY	Display the status of the Cray queues.
SUBMIT_CRAY_JOB	

Submit a job to the Cray via the NOS Cray Station.

WHO Display a list of the executing jobs (including logged-in users.

WHOAMI Display your User Initials and account number.

XEROX Route a copy of a local file to the Xerox 8700.

*** Sample Procedures ***

- 1) Compile and optionally execute a Fortran program, optionally using libraries of subprograms:

```

( 1) PROCEDURE (help_flg) Fortran_Load_Run, flr (
( 2)   input,           i : file           = $required
( 3)   library,   libraries, l : list of file = $optional
( 4)   parameter, parameters, p : list of file = $optional
( 5)   status )

( 6) "FLR Compile Fortran, load with optional libs and run"

( 7) " Echo execute statement with substitution
( 8) "
( 9)   Collect_Text output=$output substitution_mark='?' ..
(10)           until='***'
(11) FLR I=?i? LIBRARY=?library? PARAMETER=?parameter?
(12) **

(13) " Compile Fortran
(14) "
(15)   Include_Line 'Fortran I='//$string(i)//' L=$NULL'

(16) " Link with optional libraries and execute
(17) "
(18)   Execute_Task P=parameters L=library

(19)   Exit Fortran_Load_Run

(20) " created 1990/05/21 by AMDS
(21) " last modified 1990/05/21 @ 1310 by AMDS

(22) PROCEND Fortran_Load_Run

```

Explanation:

- (1-5) procedure header
 - (1) Defines the help module name, and procedure name and aliases
 - (2-4) Defines the parameters, their aliases and types
 - (5) The STATUS parameter should always be the last parameter
- (6) A comment describing what the procedure does
- (7-12) Echoing the command to the terminal substituting user values for the parameters
 - (7-8) Comments
 - (9-10) The COLT command
 - (11) The display with the strings delimited by "?" to be replaced by the execution values
 - (12) The end-of-file for COLT
- (13-15) Creates the FORTRAN command and executes it (INCLUDE_COMMAND could also have been used here)
- (16-18) Loads and executes the object module created by the compiler
- (19) Leave the procedure (if the procedure name is specified,

- it must be the first name in line 1)
- (20-21) The history of the procedure
- (22) The end of the procedure (if the procedure name is specified, it must be the first name in line 1)

2) Fetch a file from the Mass Storage System:

```

( 1) procedure (help_msfetch) MSFETCH (
( 2)   from, f, mfn           : name = $required
( 3)   to, t                  : file = $optional
( 4)   data_conversion, dc : key  B60 B56 A6  A8  D63  D64
( 5)                               keyend = B56
( 6)   status )

( 7) "MSFETCH Fetch a file from the MSS"

( 8) " Echo execute statement with substitution
( 9) "
(10)   Collect_Text o=$output sm='?' u='**'
(11) MSFETCH FROM=?from? TO=?to? DC=?data_conversion?
(12) **

(13) " Validate FROM=
(14) "
(15)   if $specified(from) then
(16)     from_file = $string(from)
(17)     if $size(from_file) > 7 then
(18)       Put_Line ' MSFETCH FROM= filename too long '//..
(19) '- 7 alphanumeric characters maximum \'//from_file//\'
(20)       exit MSFETCH
(21)     ifend
(22)   ifend

(23) " Get the file
(24) "
(25)   Get_File to=$parameter_value(to) ..
(26)           from=$parameter_value(from) ..
(27)           data_conversion=$parameter_value(data_conversion)

(28) " Closing message
(29) "
(30)   Collect_Text $output  '?'  'close_end'
(31) File ?from? fetched from MSS as file ?to?
(32)   using DC=?data_conversion?
(33) close_end

(34)   Exit MSFETCH

(35) " created 1990/03/27 by AMDS
(36) " modified 1990/03/27 @ 1112 by AMDS
(37) " modified 1990/03/28 @ 1348 by AMDS (remove fault trap)
(38) " last modified 1990/03/30 @ 0908 by AMDS (add MFN as
(39) "                                     synonym for FROM;
(40) "                                     remove MFN as
(41) "                                     synonym for TO)

```


(42) PROCEND MSFETCH

Explanation:

- (1-6) Procedure header
 - (1) Defines the help module and procedure names
 - (2-5) Defines the parameters, their aliases, types, and acceptable values
 - (6) The STATUS parameter should always be the last parameter
- (7) A comment describing what the procedure does
- (8-12) Echoing the command to the terminal substituting user values for the parameters
 - (8-9) Comments
 - (10) The COLT command
 - (11) The display with the strings delimited by "?" to be replaced by the execution values
 - (12) The end-of-file for COLT
- (13-22) Code to validate the FROM parameter
 - (13-14) Comments
 - (15/22) Encloses code to be executed if the FROM parameter was specified
 - (16) Convert the parameter to a string for later use
 - (17/21) Encloses code to be executed if the value is too long
 - (18-19) Display the error message
 - (20) Leave the procedure
- (23-27) Code to get the file from NOS
- (28-33) Closing message
- (34) Leave the procedure
- (35-41) The history of the procedure
- (42) The end of the procedure

*** Sample Function ***

- 1) The following function is used by procedure FILES to analyze the user's specified DEPTH parameter and return 0 (error), a valid depth value, or the default value.

```

( 1) function $files_set_depth (
( 2)   depth      : integer 0..3 = $required
( 3)   max_depth   : integer 2..3 = $required
( 4)   default_depth : integer 2..3 = $required
( 5) )

( 6) "$FILES_SET_DEPTH - set the depth to specified value"
( 7) "                               or default"

( 8)   IF depth > max_depth THEN
( 9)     the_depth = 0
(10)   ELSEIF depth = 0 THEN
(11)     the_depth = default_depth
(12)   ELSE
(13)     the_depth = depth
(14)   IFEND

(15)   EXIT $files_set_depth WITH the_depth

(16) " created 1990/05/16 by AMDS
(17) " last modified 1990/05/16 @ 0804 by AMDS

(18) FUNCEND $files_set_depth

```

Explanation:

- (1-5) function header
 - (1) defines the function name
 - (2) defines DEPTH parameter as a required integer with valid values of 0, 1, 2, 3
 - (3) defines MAX_DEPTH as a required integer with valid values 2 or 3
 - (4) defines DEFAULT_DEPTH similarly
 - (5) end of function header
- (6-7) A comment describing what the function does
- (8-14) Define variable THE_DEPTH as 0 (error), the default (input value is 0), or the specified depth)
- (15) Exit from the procedure returning THE_DEPTH as the function's value
- (16-17) The history of the procedure
- (18) The end of the function (if the function name is specified, it must be the first name in line 1)

The FILES procedure uses this function as follows:

```

(1) the_depth = $files_set_depth(depth,2,2)
(2) IF the_depth = 0 THEN
(3)   EXIT files WITH $status(false, 'US', 1, ..
(4) 'invalid depth when F= is specified - use 1 or 2')
(5) IFEND

```

Explanation:

- (1) The function is evaluated (it tests the value of DEPTH against a maximum of 2 and with a default of 2) and its value is stored in THE_DEPTH (this could have been some other name).
- (2-5) If the function returned 0 (error), terminate the FILES procedure with an error condition (false), and displaying a message.

***** Source Libraries *****

Source programs, data and other text may be in separate files or may be stored and maintained in source libraries. SOURCE_CODE_UTILITY (SCU) creates and maintains these libraries. Additional information may be found in CDC publication 60464313, "Source Code Management".

*** SCU Subcommands ***

The following are subcommands of the SOURCE_CODE_UTILITY used to create, maintain, and extract decks from a source library:

ADD_LIBRARY Add decks from source libraries to the working library.

CHANGE_DECK Change the contents of deck header fields.

CHANGE_DECK_NAMES
Rename decks.

CHANGE_DECK_REFERENCES
Change the deck names of COPY/COPYC directives in decks.

CHANGE_LIBRARY
Change the contents of working library header fields.

CHANGE_MODIFICATION
Change information in modification descriptions.

COMBINE_LIBRARY
Add/replace working library decks from source libraries.

CREATE_DECK Create decks.

CREATE_LIBRARY
Create an empty source library.

CREATE_MODIFICATION
Create modifications in the library modification list.

DELETE_DECK Delete decks from the working library.

DELETE_FEATURE
Delete features from a source library.

DELETE_GROUP Delete groups from a source library.

DELETE_MODIFICATION
Undo all changes introduced by specified modifications.

DISPLAY_DECK Display deck headers.

DISPLAY_DECK_LIST
List decks in working library alphabetically by deck name.

DISPLAY_DECK_REFERENCES

Display a cross-reference listing for decks (COPY/COPYC directives naming the decks).

DISPLAY_FEATURES

Display a feature's modifications.

DISPLAY_FEATURE_LIST

List the features in a source library.

DISPLAY_GROUP

List decks belonging to a group.

DISPLAY_GROUP_LIST

List the groups in a library.

DISPLAY_LIBRARY

Display the working library header.

DISPLAY_MODIFICATION

Display modification headers.

DISPLAY_MODIFICATION_LIST

List all working library modifications.

EDIT_DECK

Edit a deck.

END_LIBRARY

(Optionally) write the current working library to a file and end the current working library.

EXPAND_DECK

Expand decks.

EXPAND_FILE

Expand a text file.

EXTRACT_DECK

Extract decks from a source library.

EXTRACT_MODIFICATION

Generate a file of EDIT_FILE subcommands to make the modification changes.

QUIT

End SCU and optionally write the working library to a file.

REPLACE_LIBRARY

Replace working library decks with source library decks.

SEQUENCE_DECK

Sequence deck lines in released state.

SEQUENCE_MODIFICATION

Sequence modification lines.

SET_LIST_OPTIONS

Establish default LIST parameters for SCU subcommands.

USE_LIBRARY Start a new working library with the decks from a source library.

WRITE_LIBRARY Write the working library into a file.

*** NOS/VE Source Library Commands ***

The following NOS/VE commands are used with source libraries:

CONVERT_UPDATE_TO_SCU
Convert a source library from NOS UPDATE format to SCU format.

EXPAND_SOURCE_FILE
Expand a text file as though it were a deck in an SCU library.

EXTRACT_SOURCE_LIBRARY
Extract decks from a library for use as a separate library.

GENERATE_SCU_EDIT_COMMANDS
Compare a deck to a text file and produce a file of edit commands to convert the deck to the text file.

*** Embedded SCU Directives ***

The following SCU directives are embedded with decks:

*BLOCK
*BLOCKEND Delimit a block of text.

*COPY Copy a deck into the expanded text file.

*COPYC Conditionally copy a deck into the expanded text file.

*DECK Starts a new deck.

*IF
*ELSEIF
*ELSE
*IFEND Conditional expansion.

*TEXT
*TEXTEND Delimit a block of text whose embedded directives are not to be processed.

*WEOP Write end-of-partition on the expanded text file.

*WEOPC Conditionally write end-of-partition on the expanded text file.

*** Other Subcommands ***

In addition, there are many subcommands and functions for the selection criteria of an EXPAND_DECK, EXTRACT_DECK, and EXTRACT_SOURCE_LIBRARY; and many functions for obtaining information about the library.

***** Object Libraries *****

CREATE_OBJECT_LIBRARY (creol) is a utility for creating and maintaining object libraries of load modules, function modules, SCL procedure modules, and message modules. CREOL uses a module list while processing its subcommands. Your library is not modified until you generate a new library.

This section discusses object libraries. See section 9-4 for SCL procedure and functions libraries; and 9-3 for message/help libraries.

*** Subcommands ***

ADD MODULE

Add one or more modules to the module list.

```
Syntax:      ADD_MODULE      -or-
              ADD_MODULES    -or-
              ADDM
              L or LIBRARY or LIBRARIES=file
              -----
              M or MODULE or MODULES=list of program_name or
                                   list of range of program_name
              P or PLACEMENT=keyword
              D or DESTINATION=program_name
              STATUS=status variable
```

Parameters: L - object files, SCL procedure files, or object libraries containing the modules to be added

M - modules to be added
(default: all modules)

P - whether the modules are to be added before or after the DESTINATION module

ab	option	meaning
--	-----	-----
B	BEFORE	add modules before the destination module
A	AFTER	add modules after the destination module

(default: AFTER)

D - module before or after which the modules are to be added
(default: P=B: before the first module;
P=A: after the last module)

Remarks: If an SCL procedure has a header parameter of a non-standard type, the type definition must be created outside the procedure.

See also: COMBINE_MODULES; REPLACE_MODULES

Examples: COL/addm (bin1,bin2) P=before
 ^-- adds all modules in BIN1 and
 BIN2 at the start of the
 module list

CHANGE_COMMAND_DESCRIPTION

Change the command description of a command processor.

CHANGE_FUNCTION_DESCRIPTION

Change the command description parameters of a function processor.

CHANGE_MODULE_ATTRIBUTE

Change attributes of a module in the module list.

Syntax: CHANGE_MODULE_ATTRIBUTE -or-
 CHANGE_MODULE_ATTRIBUTES -or-
 CHAMA

M	or MODULE or MODULES=keyword or list of program_name or list of range of program_name

NN	or NEW_NAME=program_name
S	or SUBSTITUTE or SUBSTITUTES=list of record
O	or OMIT=list of program_name
G	or GATE=keyword or list of program_name
NG	or NOT_GATE=keyword or list of program_name
SP	or STARTING_PROCEDURE=program_name
OL	or OMIT_LIBRARY or OMIT_LIBRARIES= list of name
AL	or ADD_LIBRARY or ADD_LIBRARIES=list of name
R	or RETAIN=keyword or list of program_name
NR	or NOT_RETAIN=keyword or list of program_name
ONREP	or OMIT_NON_RETAINED_ENTRY_POINTS=boolean
ODT	or OMIT_DEBUG_TABLES or OMIT_DEBUG_ TABLES=keyword or list of keyword
C	or COMMENT=string
AI	or APPLICATION_IDENTIFIER=keyword or name
CPC	or CYBIL_PARAMETER_CHECKING=keyword STATUS=status variable

Parameters: M - the modules whose attributes are to be changed -or-
ALL - all modules

(defaults for rest of parameters: no change)

NN - new module name
(do not use if M=list of modules or
M=ALL)

(attribute for rest of parameters: BY_NAME)

S - a list of name substitutions -- each
record in the list specifies two names:
the existing name and the new name

O - list of names whose definitions are to be
removed from the module

G - list of entry points to which the gate
attribute is to be added -or-
ALL - all modules

NG - list of entry points to which the gate
attribute is to be removed -or-
ALL - all modules

SP - name of the entry point where execution
is to begin

OL - the local file names to be removed from
the object text (these are object
libraries to be added to the program
library list when the module is loaded)

AL - the local file names to be added to the
object text (these are object libraries
to be added to the program library list
when the module is loaded)

R - list of additional entry points to be
given the retain attribute, that is, are
to be kept in a new module created by
combining with other modules -or-
ALL - all modules

NR - list of additional entry points to lose
the retain attribute, that is, are not
to be kept in a new module created by
combining with other modules -or-
ALL - all modules

ONREP - all entry points are to be removed from
the module unless specifically retained

- ODT - the debug tables to be omitted when loading
- | ab | option | meaning |
|-----|--------------------------|--|
| LT | LINE_TABLE | omit line number table |
| ST | SYMBOL_TABLE | omit table of names and addresses of variables |
| SDT | SUPPLEMENTAL_DEBUG_TABLE | omit table of screen mode information |
| PC | PARAMETER_CHECKING | omit parameter checking records |
| | ALL | omit all debug tables |
- C - 1-40 characters to be stored in the module header
- AI - the name of the application associated with the module -or-
 \$UNSPECIFIED - remove the application identifier
 (need APPLICATION_ADMINISTRATION permission)
- CPC - only for CYBIL programs

Remarks: These changes change the attributes when a new library is generated.

Examples:

CHANGE_PROGRAM_DESCRIPTION

Change the components of a program description.

Syntax: CHANGE_PROGRAM_DESCRIPTION -or-
 CHAPD

N	or NAME or NAMES=list of name
F	or FILES or FILE=keyword or list of: file or string
L	or LIBRARIES or LIBRARY=keyword or list of: keyword or file or string
M	or MODULES or MODULE=keyword or list of program_name
SP	or STARTING_PROCEDURE=keyword or program_name
LM	or LOAD_MAP=keyword or file or string
LMO	or LOAD_MAP_OPTIONS or LOAD_MAP_OPTION=keyword or list of keyword
TEL	or TERMINATION_ERROR_LEVEL=keyword

PV or PRESET_VALUE=keyword
 SS or STACK_SIZE=keyword or integer
 AF or ABORT_FILE=keyword or file or string
 DI or DEBUG_INPUT=keyword or file or string
 DO or DEBUG_OUTPUT=keyword or file or string
 DM or DEBUG_MODE=keyword or boolean
 A or AVAILABILITY=keyword
 S or SCOPE=keyword
 LO or LOG_OPTION=keyword
 AI or APPLICATION_IDENTIFIER=keyword or
 name
 AO or ARITHMETIC_OVERFLOW=keyword or boolean
 ALOS or ARITHMETIC_LOSS_OF_SIGNIFICANCE=
 keyword or boolean
 DF or DIVIDE_FAULT=keyword or boolean
 EO or EXPONENT_OVERFLOW=keyword or boolean
 EU or EXPONENT_UNDERFLOW=keyword or boolean
 FI or FPI or FP_INDEFINITE=keyword or
 boolean
 FLOS or FP_LOSS_OF_SIGNIFICANCE=keyword or
 boolean
 IBD or IBDPD or INVALID_BDP_DATA=keyword or
 boolean
 STATUS=status variable

Parameters: N - the names of the program descriptions to
be changed

Remarks:

Examples:

COMBINE_MODULE

Add or replace modules in the module list.

Syntax: COMBINE_MODULE -or-
 COMBINE_MODULES -or-
 COMM
 L or LIBRARY or LIBRARIES=file

 M or MODULE or MODULES=list of program_name or
 list of range of program_name
 P or PLACEMENT=keyword
 D or DESTINATION=program_name
 STATUS=status variable

Parameters: L - object files, SCL procedure files, or object
libraries containing the modules to be added

M - modules to be added or replaced
 (default: all modules)

P - whether the added modules are to be placed before or after the DESTINATION module

ab	option	meaning
B	BEFORE	added modules go before the destination module
A	AFTER	added modules go after the destination module

(default: AFTER)

D - module before or after which the added modules are to be placed
(default: P=B: before the first module;
P=A: after the last module)

Remarks: If an SCL procedure has a header parameter of a non-standard type, the type definition must be created outside the procedure.

See also: ADD_MODULES; REPLACE_MODULES

Examples: COL/comm (bin1,bin2) P=before
 ^-- adds or replaces all modules in
 BIN1 and BIN2 at the start of
 the module list

CREATE_COMMAND_DESCRIPTION

For use with CYBIL programs.

CREATE_FORM_MODULE

Start the CREATE_FORM_MODULE utility to create a form.

Syntax: CREATE_FORM_MODULE -or-
 CREFM
 FN or FORM_NAME=name

 MO or MERGE_OPTION=keyword
 STATUS=status variable

Parameters: FN - the name of the form

MO - merge option

ab	option	meaning
A	ADD	add the module to the library
R	REPLACE	replace the module
C	COMBINE	add or replace the module

(default: COMBINE)

Subcommand prompt: CFM/

Examples:

CREATE_FUNCTION_DESCRIPTION

For use with CYBIL programs.

CREATE_LINKED_MODULE

Create a prelinked module from an existing module and add it to the module list.

Syntax: CREATE_LINKED_MODULE -or-
 CRELM
 N or NAME=program_name
 C or COMPONENT or COMPONENTS=list of record

 RB or RING_BRACKETS=record
 RCB or RETAIN_COMMON_BLOCK=keyword or
 list of program_name
 ISN or IGNORE_SECTION_NAMES=boolean
 SS or STARTING_SEGMENT=integer
 O or OUTPUT=file
 BT or DEBUG_TABLE=file
 NAS or NEXT_AVAILABLE_SEGMENT=integer variable
 AI or APPLICATION_IDENTIFIER=name
 DEP or DEFER_ENTRY_POINT=keyword or record or
 list of program_name
 DCB or DEFER_COMMON_BLOCKS=keyword or record or
 list of program_name
 STATUS=status variable

Parameters:

Remarks:

Examples:

CREATE_MODULE

Create a new load module from existing modules and add it to the module list.

Syntax: CREATE_MODULE -or-
 CREM
 N or NAME=program_name
 C or COMPONENT or COMPONENTS=list of record

 G or GATE or GATES=keyword or list of
 program_name
 R or RETAIN=keyword or list of program_name
 SP or STARTING_PROCEDURE=program_name
 PV or PRESET_VALUE=keyword

IBMS or INCLUDE_BINARY_SECTION_MAPS=boolean
 O or OUTPUT=file
 AI or APPLICATION_IDENTIFIER=name
 STATUS=status variable

Parameters:

Remarks:

Examples:

CREATE_PROGRAM_DESCRIPTION

Create a program description and add it to the module list.

Syntax: CREATE_PROGRAM_DESCRIPTION -or-
 CREPD

N	or NAME or NAMES=record

F	or FILES or FILE=list of: file or string
L	or LIBRARIES or LIBRARY=list of: keyword or file or string
M	or MODULES or MODULE=list of program_name
SP	or STARTING_PROCEDURE=program_name
LM	or LOAD_MAP=file or string
LMO	or LOAD_MAP_OPTIONS or LOAD_MAP_OPTION=keyword or list of keyword
TEL	or TERMINATION_ERROR_LEVEL=keyword
PV	or PRESET_VALUE=keyword
SS	or STACK_SIZE=integer
AF	or ABORT_FILE=file or string
DI	or DEBUG_INPUT=file or string
DO	or DEBUG_OUTPUT=file or string
DM	or DEBUG_MODE=boolean
A	or AVAILABILITY=keyword
S	or SCOPE=keyword
LO	or LOG_OPTION=keyword
MO	or MERGE_OPTION=keyword
AI	or APPLICATION_IDENTIFIER=name
AO	or ARITHMETIC_OVERFLOW=boolean
ALOS	or ARITHMETIC_LOSS_OF_SIGNIFICANCE= boolean
DF	or DIVIDE_FAULT=boolean
EO	or EXPONENT_OVERFLOW=boolean
EU	or EXPONENT_UNDERFLOW=boolean
FI	or FPI or FP_INDEFINITE=boolean
FLOS	or FP_LOSS_OF_SIGNIFICANCE=boolean
IBD	or IBDPD or INVALID_BDP_DATA=boolean STATUS=status variable

DELETE_MODULE

Delete one or more modules from a module list.

Syntax: DELETE_MODULE -or-
 DELM
 M or MODULE or MODULES=keyword or list of
 program_name or list of range of
 program_name

 STATUS=status variable

Parameters: M - the module or modules to be deleted -or-
 ALL - all modules

Examples: /create_object_library
 COL/addm old_library
 COL/delm (s4,m5)
 COL/genl new_library
 COL/quit
 /

DISPLAY_NEW_LIBRARY

Display the contents of the module list.

Syntax: DISPLAY_NEW_LIBRARY -or-
 DISNL

 M or MODULE or MODULES=list of program_name
 or list of range of program_name
 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=keyword or list of
 keyword
 O or OUTPUT=file
 AO or ALPHABETICAL_ORDER=boolean
 STATUS=status variable

Parameters: M - the modules whose information is to be
 displayed

DO - display options

ab	option	meaning
---	-----	-----
	NONE	type and name only
DT	DATE_TIME	creation date/time
EP	ENTRY_POINT	entry point names
H	HEADER	module header information
L	LIBRARY	local file names in the
	LIBRARIES	object text
R	REFERENCE	external references
C	COMPONENT	module headers of
		component modules of a
		bound module
	ALL	all options
	(default: DATE_TIME or the value set by	
	last SET_DISPLAY_OPTIONS)	

AO - TRUE - alphabetical listing
 FALSE - listing is in library or file order
 (default: FALSE)

Remarks: DISNL displays the modules as they could appear in a library created by GENERATE_LIBRARY.

Examples: /creol
 COL/addm mylib
 COL/disnl
 <list of contents>
 COL/more CREOL subcommands
 COL/genl mylib.\$next
 COL/quit
 /

GENERATE_LIBRARY

Generate a new object library containing the modules in the module list. It can also write an object file, SCL procedure text file, form source file, form variable file, or CREATE_MESSAGE_MODULE subcommands.

Syntax: GENERATE_LIBRARY -or-
 GENL
 L or LIBRARY=file

 F or FORMAT=keyword
 STATUS=status variable

Parameters: L - the file to hold the new library

F - the output format:

ab	option
-----	-----
L	LIBRARY
F	FILE
FS	FORM_SOURCE
FV	FORM_VARIABLE
SP, SP_PROC	SCL_PROCEDURE
MM	MESSAGE_MODULE
(default: LIBRARY)	

Remarks: Requires APPEND and SHORTEN access.

If the library is already in your (or someone else's) command list, it cannot be overwritten. You should create a new cycle. An attempt to overwrite such a file will cause a unique file to be created in \$LOCAL.

See also: CREATE_OBJECT_LIBRARY

QUIT End a CREATE_OBJECT_LIBRARY utility session.

Syntax: QUIT -or-
QUI

REORDER_MODULE

Reorder modules in the module list.

Syntax: REORDER_MODULE -or-
REORDER_MODULES -or-
REOM
M or MODULE or MODULES=list of program_name or
list of range of program_name

P or PLACEMENT=keyword
D or DESTINATION=program_name
STATUS=status variable

Parameters: M - list of modules in the order you want them

P - whether the modules are to be added before or
after the DESTINATION module

ab	option	meaning
--	-----	-----
B	BEFORE	add modules before the destination module
A	AFTER	add modules after the destination module

(default: AFTER)

D - module before or after which the modules are
to be added

(default: P=B: before the first module;
P=A: after the last module)

Remarks:

Examples: module list: A, B, C, D, E
REOM (c,b)
module list: A, D, E, C, B
REOM d B
module list: D, A, E, C, B
REOM e A c
module list: A, D, C, E, B

REPLACE_MODULE

Replace modules in the module list.

Syntax: REPLACE_MODULE -or-
 REPLACE_MODULES -or-
 REPM
 L or LIBRARY or LIBRARIES=file

 M or MODULE or MODULES=list of program_name or
 list of range of program_name
 STATUS=status variable

Parameters: L - object files, SCL procedure files, or object
 libraries containing the modules to be added

M - modules to be added
 (default: all modules)

Remarks: If an SCL procedure has a header parameter of a
 non-standard type, the type definition must be
 created outside the procedure.

See also: ADD_MODULES; COMBINE_MODULES

Examples: COL/repM (bin1,bin2)
 ^-- replaces all modules in BIN1 and
 BIN2

SATISFY_EXTERNAL_REFERENCE

Add modules to the module list which satisfy external
 references.

Syntax: SATISFY_EXTERNAL_REFERENCE -or-
 SATISFY_EXTERNAL_REFERENCES or
 SATER
 L or LIBRARY=list of file

 STATUS=status variable

Parameters: L - object library files in the order they are to
 be searched

Remarks:

Examples:

SET_DISPLAY_OPTION

Change/display default settings for subsequent DISPLAY_NEW_LIBRARY subcommand of CREOL.

Syntax: SET_DISPLAY_OPTION -or-
 SET_DISPLAY_OPTIONS or
 SETDO

DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=keyword or
 list of keyword
 STATUS=status variable

Parameters: DO - display options

ab	option	meaning
--	-----	-----
	NONE	type and name only
DT	DATE_TIME	creation date/time
EP	ENTRY_POINT	entry point names
H	HEADER	module header information
L	LIBRARIES	local file names in the object text
R	REFERENCE	external references
C	COMPONENT	module headers of component modules of a bound module
	ALL	all options

(initial default: DATE_TIME)

Remarks:

Examples:

*** DTRC Object Libraries ***

The following object libraries have been added to NOS/VE at DTRC:

.NSYS.DTLIB - Subprograms written or maintained by the
Computer Center

To use: /execute_task file=\$user.main_lgo ..
../library=.NSYS.DTLIB

.NSYS.UTILITY - Programs written or maintained by the
Computer Center

To use: .nsys.utility.<program>

*** Examples ***

- 1) Create a library of subprograms (all files are assumed to be in your working catalog).

```

/ftn mysubss          <-- compile subprograms
/create_object_library <-- invoke CREOL utility
COL/add_modules $local.lgo <-- addm
COL/display_new_library <-- disnl -- look at what new
                           library will contain
COL/generate_library mylib <-- genl -- create the new library
                           MYLIB
COL/quit              <-- leave CREOL
/
```

- 2) Add a subprogram to an existing library.

```

/fortran newsub
/creol
COL/addm mylib          <-- or "addm (mylib,$local.lgo)"
COL/addm $local.lgo
COL/disnl               <-- optional
COL/genl mylib          <-- replace old library
COL/quit
/
```

- 3) Replace a subprogram in an existing library of subprograms.

```

/for oldsub
/creol
COL/addm mylib
COL/repn $local.lgo
COL/genl mylib.$next    <-- make new cycle of library
COL/quit
/disol mylib            <-- display contents of mylib
/
```

- 4) Delete subprograms BADSUB1 and BADSUB2 from an existing library.

```
/creol  
COL/addm mylib  
COL/delete_modules (badsub1,badsub2)  
COL/genl mylib  
COL/quit  
/
```

***** Program/Command/Procedure Execution *****

Programs, commands and procedures are executed in one of the following ways:

- . name reference (if it is in your command list)
- . filename reference (if it is in a catalog in your command list)
- . full-path file reference (any file for which you have read or execute permission)
- . utility name and subcommand name (if you are in a utility with one or more active subutilities and a subcommand name is the same in the utility and/or its subutilities)
- . EXECUTE_COMMAND (to execute a single NOS/VE command as an asynchronous task)
- . EXECUTE_TASK (to execute a single program as either a synchronous or an asynchronous task -- use when multiple load modules, subprogram libraries, fault switches, memory preset, load maps, debugging mode, etc., are needed)

See Appendix H for descriptions of many of the commands mentioned in this section.

*** Command Lists ***

A command list is the set of commands and functions (section 8-4) available to you. Commands may reside in object libraries, in catalogs, as command entries in \$SYSTEM, or as utility subcommands. Each entry is called a command list entry. You can change your command list by adding, deleting, or replacing entries. You can also control the search order.

At login, your command list includes all NOS/VE system commands (in \$SYSTEM), the \$LOCAL catalog, and several additional entries with routines developed at DTRC.

\$SYSTEM commands may be executed specifically by:

\$SYSTEM.command_name

which causes only the \$SYSTEM command list entry to be searched.

When you start a utility, its command list entry (containing subcommands and functions) is added at the start of your command list. It is deleted when you exit from the utility.

Control statements are used to structure and control the job flow. The command list entry containing them is always searched first. It cannot be deleted or replaced in your command list. It also contains:

COLLECT_TEXT
JOB and JOBEND
TASK and TASKEND
UTILITY and UTILITYEND

which also are in \$SYSTEM.

The following commands, which are also in \$SYSTEM, are in a special command list entry which is always searched last and which cannot be deleted or replaced in your command list:

CHANGE_COMMAND_SEARCH_MODE
CREATE_COMMAND_LIST_ENTRY
DELETE_COMMAND_LIST_ENTRY
DELETE_VARIABLE
DISPLAY_COMMAND_INFORMATION
DISPLAY_COMMAND_LIST
DISPLAY_COMMAND_LIST_ENTRY
DISPLAY_VALUE
DISPLAY_VARIABLE_LIST
GET_LINE
INCLUDE_COMMAND
INCLUDE_FILE
INCLUDE_LINE
LOGIN
LOGOUT
PUT_LINE

*** Search Modes ***

The command list can be searched in one of the following ways:

GLOBAL	All entries in the command list can be searched; commands specified by path or command name may also be executed.
RESTRICTED	All entries in the command list are candidates to be searched (to search beyond the first entry, the command must be preceded by a slash (/); commands specified by path or command name may also be executed.
EXCLUSIVE	Only the first entry of the command list is searched; commands specified by path or command name may not be executed. Also, CHANGE_COMMAND_SEARCH_MODE, CREATE_COMMAND_LIST_ENTRY, and DELETE_COMMAND_LIST_ENTRY cannot be executed. SCL assignment and control statements may always be executed.

*** Examples ***

- 1) Execute program in file MYPROG in the current directory (.ABCD):

.abcd.myprog

- 2) Execute program MYPROG in library MYLIB in the current directory (.ABCD):

.abcd.mylib.myprog

- 3) Execute program MYPROG in library MYLIB, which is in my command list:

myprog

***** On-line Manuals *****

This section is reserved for a description of on-line manuals. It will be part of Revision 1.

***** Other Software *****

*** Accessing Other Software ***

Programs obtained from other vendors are normally execute-only. To execute them, you normally need only specify the filepath:

.un.file <-- where un is the username under which the
 program is stored (normally
 NSYS)
 file is the name of the program file

** .NSYS **

DTRC-written programs are in the NSYS catalog. As of the date of this page, .NSYS contains:

None

***** CDC CYBER 180/860 -- NOS *****

The Control Data Corporation (CDC) CYBER 180 model 860 when operating under NOS has a single central processing unit (CPU) and 16000000 octal (2,097,120) 60-bit words of memory, of which 400000 octal is addressable by each job.

The CPU has 24 registers for operating on information: 8 address (A), 8 operand (X) and 8 increment (B) registers. The CYBER 860 has a buffer of 12 central memory (CM) words of instructions, called an instruction stack, and a 2048-word, high-speed cache memory.

Peripheral processors (PPs) are small computers (4096 12-bit words of memory) which handle most input and output (I/O). There are 20 normal and 5 concurrent PPs on the CYBER 860.

There are 28 normal and 5 concurrent I/O channels. Most peripheral equipment interfaces with the central system through the PPs via the I/O channels. The printers and remote terminals interface with the system via CDCnet.

*** NOS Version 2.7.1 ***

One operating system for the CDC CYBER 860 at DTRC is called the Network Operating System, version 2.7.1 (NOS 2.7 - level 716) and differs only slightly from the standard NOS system. The interactive subsystem is called IAF (InterActive Facility); the subsystem for medium-speed remote batch terminals is called RBF (Remote Batch Facility).

Permanent files (user programs and data files retained for frequent use) reside on model 895 disk drives and the Mass Storage System. User files, if not specifically requested on a tape, will be assigned to available disk areas.

NOS operates in a "dual-state" with NOS/VE (see Chapter 8).

*** Accessing NOS on the CDC 860 ***

To access NOS on the CDC CYBER 860:

- . dial (301) 227-5200 <-- this will connect you with DTNET
 at 1200 baud (see page 1-1-4 for
 higher speeds)
- . press the RETURN key until it displays the DTNET> prompt
- . enter "CONNECT cdc860" (or "C cdc860") to connect to NOS
- . in response to the Family: prompt, enter either
 - . ,xxxx,pw,IAF (xxxx is your User Initials,
 pw is your login password
 IAF is the NOS InterActive Facility)
 - or-
 - . RETURN, then the rest of the information one item at a time
as prompted
- . in response to the CHARGE NUMBER (if requested): prompt,
enter your Job Order Number (see Appendix I: CHVAL,CN)
- . when you receive the "/" prompt, you are in IAF
 - . if you entered your Job Order Number incorrectly, you must
enter "CHARGE,number." at the "/" prompt until a valid
number is accepted

*** Terminal Keys ***

NOS supports screen formatting for most display terminals. Many commands use a full-screen mode when the SCREEN command is used. When these commands show function keys, they are shown as they appear on a CDC Viking 721 terminal. When using other terminals, different keys or sequence of keys may be needed for the desired function.

The following table shows the key(s) to be used for some terminals at DTRC. The DT100 keypad for use in FSE is on page 9-1-5.

CDC Viking 721	DEC VT100	Tektronix T4115	CDC Viking 721	DEC VT100	Tektronix T4115
F1	keypad 1 + RETURN	F1	shift F1	PF1 + RETURN	shift F1
F2	keypad 2 + RETURN	F2	shift F2	PF2 + RETURN	shift F2
F3	keypad 3 + RETURN	F3	shift F3	PF3 + RETURN	shift F3
F4	keypad 4 + RETURN	F4	shift F4	PF4 + RETURN	shift F4
F5	keypad 5 + RETURN	F5	shift F5	keypad - + RETURN	shift F5
F6	keypad 6 + RETURN	F6	shift F6	keypad , + RETURN	shift F6
F7	keypad 7 + RETURN	F7	shift F7	kp ENTER + RETURN	shift F7
F8	keypad 8 + RETURN	F8	shift F8	keypad . + RETURN	shift F8
F9	keypad 9 + RETURN	ctrl A	shift F9		ctrl Q
F10		ctrl S	shift F10		ctrl W
F11		ctrl D	shift F11		ctrl E
F12		ctrl F	shift F12		ctrl R

CDC Viking 721	DEC VT100	Tektronix T4115	CDC Viking 721	DEC VT100	Tektronix T4115
F13			shift F13		
F14			shift F14		
F15			shift F15		
F16			shift F16		
NEXT	RETURN	RETURN			
HELP			shift HELP		
BACK			shift BACK		
STOP ctrl T+NEXT	ctrl T + RETURN	ctrl T + RETURN	shift STOP	ctrl T + RETURN	ctrl T + RETURN
FWD			shift FWD		
BKW			shift BKW		
UP			shift UP		
DOWN			shift DOWN		
			shift CLEAR		

SCREEN,DT100 before entering FSE puts you in full-screen mode with the following definition of the keypad.

PF1	PF2	PF3	PF4
del b	join	del c	del w
7	8	9	-
ins b	split	ins c	ins w
4	5	6	,
mark c	move	del l	pos
1	2	3	ENTER
mark l	copy	ins l	
0		.	
forward		back	home

After pressing one or more of the keypad keys, the RETURN key must be used to perform the requested functions.

The arrow keys may be used to position the cursor.

*** Direct versus Indirect Files ***

Unlike most other operating systems, NOS supports two distinct types of permanent file: direct and indirect.

Disk space is allocated by PRU (physical record unit) with one PRU holding 64 words (640 6-bit or 320 8/12-bit characters).

A direct file (not to be confused with a "direct access" or random file) occupies one or more blocks of 704 PRUs and is charged by the number of blocks needed to hold the file. (A 705-PRU file occupies 2 blocks.) When you ATTACH a direct file, you are working with the actual file. Changes made by a program immediately change the actual file and cannot be "undone". Changes made while editing alter the file with you QUIT FSE. To "undo" the changes before QUITting, enter "SET FILE dummy" (SF dummy) and the changes will be made to local file "dummy".

An indirect file occupies up to 696 PRUs and is charged by the number of PRUs needed to hold the file. (A 1-PRU indirect file occupies 1 PRU, while a 1-PRU direct file occupies 1 block, or 704 PRUs.) When you GET an indirect file, you are working with a copy of the file. Any changes made by editing or by a program affect only the copy and may be "undone" any time prior to REPLACE-ing the file. An indirect file is actually a portion of a larger "file" containing other indirect files. You cannot work with the actual file because a change could lengthen it, thus destroying the file which physically follows it. When you REPLACE an indirect file, the new file is put wherever there is room in the larger "file". Notice that the largest indirect file fits within one block.

Direct files are required for files larger than 696 PRUs and for files which require that changes be made in real time, perhaps for other users of the file.

Indirect files are recommended for short files. They are especially useful for source programs and data files which are under development, where you might want to try some changes but not make them permanent until you decide. The largest indirect file actually holds a lot of information (445,440 6-bit, 222,720 8/12-bit characters).

*** Batch Jobs ***

Two forms of batch job may be SUBMITTED to the Input Queue. A job may be in an indirect or direct file, since it will be copied to the queue.

1) Traditional form (complete JCL):

```
jobname,....  
USER,username,password.  
CHARGE,account.  
ATTACH,MYSOR.  
FTN5,I=MYSOR.  
LGO.  
(EOR)  
  <data, if needed>
```

2) Reformatting directive form (assumes batch password and account number are the same as the interactive):

```
/JOB  
jobname,....  
/USER  
/CHARGE  
ATTACH,MYSOR.  
FTN5,I=MYSOR.  
LGO.  
/EOR  
  <data, if needed>
```

Either form may be sent to the input queue by

```
SUBMIT,jclfile,TO.
```

to have the output in the Wait Queue.

If the traditional form is in an indirect local file, it may alternatively be sent to the input queue by

```
ROUTE,jclfile,DC=TO.
```

The reformatting-directives form is preferred since such JCL files do not need modification each time the user password is changed. More importantly, it keeps your account number and password away from prying eyes. Note that end-of-record is not in the same form for these 'SUBMIT' files.

**** Killing Batch Jobs ****

CDC NOS jobs are identified by their Job Sequence Numbers (jsn).
To find the jsn, use

ENQUIRE, JSN.

To kill a batch job, use

```
$ DROP,jsn.      <-- drops job <jsn> from all queues
```

**** Batch Job Classes ****

Batch jobs in NOS fall into three service classes or priorities: P2 (normal), P3 (I/O), and P4 (I/I). Charges for P3 are higher than for P2, and charges for P4 are higher than for P3. To specify a job class, use P2, P3, or P4 on the JOB statement. P2 is the default. SECURE processing is not available on the CYBER 860.

The following chart shows, for each job class, the maximum number of such jobs to be allowed to execute at the same time. They are listed in order of relative priority, lowest priority first.

Priority Level	Service Class	Priority	Number of Jobs
P2 Jobs	Normal Batch (Default)	21	3
P3 Jobs	I0 Service Class	24	7
P4 Jobs	I1 Service Class	27	20
Interactive		30	100

For the current rates:

On NOS: BEGIN,RATES.

On VMS: HELP RATES

*** Accessing Other Networks ***

CDCnet supports Telnet and FTP using TCP/IP. Note that the Host Tables cannot be updated automatically, therefore, if the desired host is not available, call User Services, (301) 227-1907.

** Connecting to Another System **

To connect to another system, the CDCnet command "CREC" must be used. If at the CDCnet level, the command is "CREC <hostname>". If logged in to CYBER 860, the command is "%CREC <hostname>" and a "\$B" session will be started. The original "\$A" (NOS) session will be logged out automatically if you do not reconnect to it within the default time out period (20 minutes). After logout on the remote system, to return to the \$A session, type "CHAWC \$A".

If you connect via DTNET, you must use DTNET to initiate another session.

** Transferring Files **

FTP may be used to transfer files to or from the CDC CYBER 860 and VAXcluster, OASYS VAXes, Code 60 VAX, or other systems defined in the CYBER TCP/IP Host Tables. All commands have a unique CDCnet format, but most have as an alias the familiar FTP commands.

When logged in to NOS:

- . Files to be sent must be local
- . Files received will be local and you must DEFINE or SAVE them
- . Files received will be ASCII (6/12)
- . Files to be sent assume ASCII - if Display Code files contain carets (^), use 'name,CS=D' to enforce a Display Code translation (the apostrophes are required)

Use %1 to cancel an FTP command. Using %2 to stop a printout will exit FTP.

For OASYS VAX, the username and password must be entered lower case.

An ASCII (6/12) indirect prolog file called FTPPRLG may be used to include commands needed for each FTP execution.

When logged in to other system (see also page 4-1-5):

- . Files send to NOS will be direct unless 'name/IA' is used
- . Permissions may be set on NOS file by 'name/CT=PU'
- . Files being received at other host may be indirect or direct
- . Files being received that are only on MSS cartridge will give STAGE INITIATED message without transfer; repeat the GET in a few seconds

Limited help is available using "HELP" or "DISCI command".

** Some FTP Commands **

The following are some of the commands available in FTP. They are illustrated using our VAXcluster.

ftp	<-- File Transfer Protocol
verbose, on	<-- useful to see information on your FTP session - may be turned on and off at any time
open dtvms3	<-- Enter DT3
User?	<-- Enter your VAX user name
Password?	<-- Enter your current VAX password; if rejected, type USER <user> and you will be prompted again for your password
get <VMS filename> <NOS local file>	<-- must be on one line; both file names are required
put <NOS local file> <VMS filename>	<-- if <VMS filename> is omitted, <NOS local file> is used without a filetype; embedded EORs are removed; only the first file is sent; the sent file will have WORLD:READ permission
put '<NOS local file>,CS=D' <VMS filename>	<-- same, except if <VMS filename> is omitted, the generated <VMS filename> is "<NOS local file>\$5LCSS7DD."
ls	<-- list names of remote files one per line
dir -or- list	<-- list file names with date, permission, size (in bytes) of all or directory
list <name>, <outfile>	
pwd	<-- display remote working directory
cd <name>	<-- change remote working directory
cdup	<-- back up one level directory
quit -or- bye	<-- exit from FTP

***** NOS CCL Commands *****

The NOS CYBER Control Language (CCL) statements are grouped by function in this section. See Appendix I for a description of the syntax for each command. (DTRC) indicates a command or program added at DTRC.

*** Flow Control ***

BEGIN Transfer control to a procedure.

DISPLAY Evaluate an expression and put the result into the job's dayfile in octal and decimal.

ELSE Terminate skipping (false IF command with same label), or initiate skipping (true IF command with same label) to ENDIF with same label.

ENDIF Terminate skipping by a SKIP, IF, or ELSE command with a matching label.

ENDW The end of a WHILE loop.

EXIT Resume processing commands after a previous error.

IF Conditionally skip one or more commands.

name Transfer control to a procedure.

NOEXIT Continue processing with the next command even if an error has occurred (suppress EXIT processing).

ONEXIT Reverse the effect of NOEXIT.

REVERT Return from a procedure.

SET Assign a value to a control register, an error flag, or the enter-skipped-commands-in-the-dayfile flag.

SKIP Unconditionally skip succeeding commands, ending with an ENDIF with a matching label.

WHILE Start of a command loop.

*** Job Control ***

* Entire line is a comment.

BLOCK Add one or more lines of 10x10 block letters to a file.

CHARGE Validate charging information for the job.

CHVAL,CN Make your current account number your login default.

COMMENT Place a comment in the system dayfile and the dayfile for any of your jobs.

CSUBMIT Submit a job to a Cray mainframe.

CTIME Put the accumulated CPU time (in seconds) into the job's dayfile.

DAYFILE Write the job's dayfile to a file.

DROP Drop any of your executing or queued files (except the job issuing the DROP command).

ENQUIRE Get information about your jobs.

ENTER Enter a series of commands on one line.

ERRMSG Control the display of error messages in a procedure.

GO Clear the pause bit of one of your jobs.

job Identifies requirements for a batch job.

JOB COST (DTRC) Display the job cost, SRUs used, and CPU time in the dayfile and file OUTPUT.

LENGTH Gives the current status of one of your local files.

LIMITS List your validation limits.

LISTLID List network configuration and host availability information.

MFL Reset maximum field length for subsequent job steps.

NEWCHRG (DTRC) Set the current charge number of some or all files.

NORERUN Clear the job rerun status.

NOTE Create a file with the command line containing the lines for the new file.

OFFSW Clear sense switches.

ONSW Set sense switches.

PASSWOR Change your batch and interactive passwords.

PAUSE Set the pause bit of one of your executing jobs.

QGET Assign a queued file to your job.

RERUN Allow a job to be rerun if necessary.

RESOURC Specify that more than one tape drive is required.

RFL Set running field length.

RTIME Put the real-time clock time into the dayfile.

SETASL Set the SRU limit for an accounting block.

SETCORE Preset each word of the field length except for RA+2.

SETJOB Change some of the current job's attributes.

SETJSL Set the SRU limit for each subsequent job step.

SETPR Decrease the CPU priority of a job.

SETTL Set the CPU time limit for each subsequent job step.

STIME Put the accumulated SRU value for the job into the dayfile.

SUBMIT Put a job into the input queue.

SWITCH Set sense switches.

UPROC Specify a user prologue to be executed each time you start a job.

USER Identify you and provide validation information for each batch job.

*** Interactive ***

** Terminal Control **

ASCII Set terminal to ASCII.

CSET Change the terminal's character set mode.

LINE Set your terminal for line (or scrolling) mode for FSE and HELPME.

NORMAL Reverse the effect of ASCII, AUTO, BRIEF, and CSET,ASCII commands.

SCREEN Set your terminal for screen mode.

TDU Compile a terminal definition file and store it in a user library which can later be accessed by a SCREEN or LINE command.

TRMDEF Change terminal characteristics.

%1 Interrupt current job step.

%2 Terminate current job step.

%HELP Display the CDCnet command list.

.

** Subsystem Selection **

ACCESS Select the ACCESS subsystem.

BASIC Select the BASIC subsystem.

BATCH Select the BATCH subsystem.

EXECUTE Select the EXECUTE subsystem.

FORTRAN Select the FORTRAN subsystem.

NULL Select the NULL subsystem.

** Interactive Status **

%D Immediately detach a terminal job from the terminal.

%E Immediate detailed job status.

%S Immediate abbreviated job status.

** Job Processing **

APPSW Switch temporarily to an alternate NAM application program.

BYE Terminate an application.

DIAL Send a one-line message to another user.

EXPLAIN Retrieve an on-line version of a CDC manual.

GOODBYE Terminate an application.

HELLO Logs you out of IAF and switches you to another application, or starts another login.

HELP Ask for help.

HELPBE On-line help for the NOS-equivalent of NOS/BE commands.

HELPME Display a brief description of a command, prompt for parameters, execute the command.

LIST List lines of a local file.

LOGIN Terminate your current application and start another.

LOGOUT Terminate an application.

RECOVER Recover a detached job or interrupted terminal session.

REDO Modify and re-execute a previously entered command without having to retype the entire command.

SHOW Display a screen formatting panel for testing purposes.

WHATJSN Get the job sequence number for the specified user name.

X Execute a batch command.

XMODEM Transfer a file between NOS and a PC using the Christensen protocol.

*** File Management ***

ASSIGN Assign a file to a device.

AUX (DTRC) Turn an auxiliary printer on or off, send control codes to control character size and to eject a page.

AUXPRNT (DTRC) Print a file on an auxiliary printer.

BKSP Backspace a file (by logical records).

CLEAR Release all (or all but one or more specified) auto-drop files assigned to the job.

COPY Copy data from one file to another.

COPYBF Copy a multi-file file.

COPYBFR (DTRC) Restore the "randomness" of a random file which was copied using a sequential copy such as COPYBF.

COPYBR Copy a records from one file to another.

COPYCF Copy a coded multi-file file.

COPYCR Copy a records from one coded file to another.

COPYEI Copy a file through end-of-information.

COPYSBF Copy a file, shifting the lines one character to the right for printing on a printer.

COPYSF8 (DTRC) Copy an ASCII8 file, shifting the lines one character to the right for printing on a printer.

COPYX Copy a file until a user-specified condition is met.

FCOPY Convert a file from one character set to another.

FILE (CRM) Describe the attributes of a file.

LO72 Reformat files.

LOCK Prevent writing on a local file.

OUT Send deferred output files to the print or punch queue immediately.

OVWRITE Overwrite files to destroy their contents.

PACK Remove all EORs and EOFs from a file.

RENAME Change the name of a local file.

REQUEST Assign a file to receive checkpoint dumps, or send a message to the operator to assign to the described device.

RETURN Release files (and file space depending on file type) assigned to a job.

REWIND Position file: at beginning-of-information (BOI).

ROUTE Direct the disposition of an indirect file and define its characteristics.

SCOPY Copy coded file(s) displaying EORs and EOFs in the receiving file.

SETFS Set the auto-drop/no-auto-drop status of files assigned to your job.

SKIPEI Position a file at end-of-information.

SKIPF Skip forward a specified number of files.

SKIPFB Skip backward a specified number of files.

SKIPR Skip forward a specified number of record or file marks.

TCOPY Copy X (binary), E, B, or SI files to disk, I, or SI (binary) tape.

TDUMP Octal or alphanumeric dump of all or part of a file.

UNIROUT (DTRC) Shift a UNICOS ASCII8 file and print it.

UNLOAD Release files assigned to your job and perhaps their file space.

UNLOCK Rescind the LOCK command and clear the write interlock for specified local disk files.

VERIFY Binary file comparison.

WRITEF Write a specified number of file marks on a file.

WRITER Write a specified number of empty records on a file.

*** Permanent File ***

APPEND Append information to the end of an indirect access file without retrieving the file.

ASSIGN Assign a file to a device.

ATTACH Assign a direct access permanent file to a job.

CATLIST List permanent file information.

CHANGE Change some characteristics of a permanent file.

DEFINE Create an empty direct access permanent file.

GET Get copies of indirect access permanent files as local files.

GETASC (DTRC) Get an indirect or direct 8/12 ASCII file and convert to 6/12 Display Code.

GETATT (DTRC) Get an indirect file or attach a direct file.

PERMIT Explicitly permit another user to access one of your private files.

PURGALL Purge all your files which match the parameters.

PURGE Purge one or more direct or indirect permanent files.

PUTASC (DTRC) Convert a 6/12-bit Display Code file to 8/12-bit ASCII and make it a permanent indirect or direct file.

RECLAIM Selectively backup and reload local and permanent files.

REPLACE Purge an indirect access file and replace it with a copy of a local file; save a copy of a local file as a new indirect access file.

SAVE Put a copy of a local file on disk as an indirect access file.

*** Dump Memory ***

DMB Binary dump of exchange package and central memory.

DMD Dump the exchange package or central memory in both octal and display code.

DMP Dump the exchange package or central memory in octal.

*** Tape Management ***

ASSIGN Assign a file to a device.

BLANK Blank label a magnetic tape.

LABEL Mount a magnetic tape and, if labelled, check the label.

LISTLB List labels of an ANSI-labelled tape.

REQUEST Assign a file to receive checkpoint dumps, or send a message to the operator to assign to the described device.

REQUEST Request a tape be mounted (LABEL is preferred).

TAPDMP (DTRC) Driver to execute TAPEDMP program.

TAPEDMP (DTRC) Analyze a magnetic tape.

TPAUDIT (DTRC) Audit the magnetic tapes assigned to you.

TPGET (DTRC) Get up to 3 magnetic tapes for use on the CDC CYBER 860 or DEC VAXcluster.

TPRLS (DTRC) Release magnetic tapes assigned to you by TPGET.

VSN Associate a local file name with one or more volume serial numbers.

*** Checkpoint/Restart ***

CKP Take a checkpoint dump.

RESTART Restart a checkpointed job.

*** Procedures ***

BEGIN Transfer control to a procedure.

REVERT Return from a procedure.

*** System Utilities ***

FSE Invoke the full screen editor.

UPDATE Create, edit or copy an Update-formatted program library.

*** Library Maintenance ***

CATALOG List information about each record in a file.

COPYL Selective single replacement of object modules.

COPYLM Selective multiple replacement of object modules.

GTR Selective extraction of records from a file.

ITEMIZE List information about each record of a binary file.

LIBEDIT Create and maintain a library of programs, subprograms, procedures, or text.

LIBGEN Create a new user library of routines for use by the loader.

LIBRARY (Loader) Specify a set of global libraries to be searched for externals and programs and the order in which they are to be considered.

ULIB Create a user library; add, delete or replace a record.

VFYLIB List differences in name, type, length, and checksum for the records of two files.

*** Programming Languages ***

COBOL5 Compile COBOL 74 program.

FTN5 Compile Fortran 77 program.

X,BASIC Compile a BASIC program without changing to the BASIC subsystem.

*** Loader and Loader-related Control Statements ***

EXECUTE Complete loading, fill unsatisfied references by system (and user) library search, generate load map and execute the program.

LDSET Set any of several loader options for the current load only.

LGO Load and execute the default compiler binary output file.

LIBLOAD Load modules from specified library which contains the specified entry points.

LIBRARY Specify a set of global libraries to be searched for externals and programs and the order in which the libraries are to be searched.

LOAD A list of files whose contents are to be loaded.

MAP Specify the global default option for load maps.

name Load and execute binary program or procedure in local file <name>.

NOGO Complete the loading of a program, including generating load map, but do not execute.

REDUCE Turn the reduce flag on or off.

RFL Set running field length.

SATISFY Satisfy unsatisfied externals prior to normal satisfaction at load completion.

SLOAD Selectively load modules from local file <lfn>.

*** Combination ***

FLR (DTRC) Compile FTN5, load with up to 2 libraries, and run.

*** Miscellaneous ***

DATE (DTRC) Put the date and time into the dayfile.

FICHE (DTRC) Route a local file to microfiche.

GRIPE (DTRC) On-line gripe/suggestion facility.

NEWRTNS (DTRC) Display a brief description of new routines added at DTRC.

NEWS (DTRC) Display the current news items.

OFLREQ (DTRC) Generate an Off-Line Request to process tapes for the Calcomp, Xerox or Microfiche.

OLDNEWS (DTRC) Display the old news items.

RATES (DTRC) Display the current computer rates.

WHOAMI (DTRC) Display your user ID and account number.

XEROX (DTRC) Send a file to the Xerox 8700.

***** Procedures *****

A procedure is a group of control statements separate from the job control statement file. Calling a procedure provides a simplified way to process that group of control statements. A procedure may be called by a job repeatedly, by another procedure, or by itself.

In general, the "CCL CYBER Control Language Reference Guide" for NOS/BE can be used for NOS. It is available from User Services. See also "NOS 2 Reference Set Volume 3: System" for additional features.

*** Procedure Directives ***

Procedure directives allow you to control procedure processing options. The procedure "title", the help text, and all "text" and "message"s may be in 6/12-bit upper and lower case.

.CC(n) Specify the concatenation character for a procedure.

.CORRECT,text.

.CORRECT=text.

Specify the prompt to follow an incorrect procedure parameter entry for an interactive procedure.

.DATA,lfn.

Create a local file from a procedure.

.DATA is terminated by another .DATA, an end-of-record (not .EOR), an end-of-file (not .EOF), or end-of-information.

.IF, .ELSE, .ENDIF can be used within the data lines for conditional inclusion.

.ELSE,label.

End skipping by a matching .IF or start skipping to a matching .ENDIF in a procedure.

.ENDHELP.

Mark the end of the help section of an interactive procedure.

.ENDIF,label.

End skipping from a matching .IF or .ELSE in a procedure.

.ENTER,text.

.ENTER=text.

Specify the prompt for before an interactive procedure parameter entry.

.EOF. Put an end-of-file into a file created by .DATA in a procedure or in the procedure command record.

.EOR. Put an end-of-record into a file created by .DATA in a procedure or in the procedure command record.

.EX.command.

Submit a command to the system for immediate execution.

.EXPAND,option

End or resume procedure expansion.

.Fn,text.

.Fn=text.

Specify a label for one of the six programmable function keys for use with screen mode parameter displays in an interactive or menu procedure.

On a VT-100, these correspond to keypad keys 1-6.

.HELP.

.HELP,NOLIST.

.HELP,param.

.HELP,param,NOLIST.

Specify the help (upper and lower case) text for a procedure or parameter.

.IC(n)

Specify the inhibit character for a procedure.

.IF,expression.command. <-- note 2 terminators

.IF,expression,label. <-- only 1 terminator

Conditional expansion of a procedure.

.NOCLR,message.

.NOCLR=message.

Inhibit automatic screen clearing during a procedure.

.NOTE,message.

.NOTE=message.

Specify a message to be displayed on the screen and in your dayfile at the end of a procedure call (when all required parameters are supplied).

Use the NOTE command to display comments during the execution of a procedure.

.PAGE,text.

.PAGE=text.

Specify the string to precede the page number on the screen.

```
.PROC,pname*I"title",p1,p2,...,pn.ck.      <-- interactive
.PROC,pname*M"title",keyword=(selections).ck.  <-- menu
.PROC,pname,p1,p2,...,pn.                  <-- passive
```

The procedure header specifying the procedure name and parameters, and enabling parameter prompting.

```
Parameters:  pname      - the name of the procedure
                        (1-7 alphanumeric, first should be
                        alphabetic; append *I for
                        interactive, *M for menu-driven;
                        nothing for passive)

                        title      - the procedure title
                                    (default: pname)

                        pi         - up to 50 parameters, each of the
                                    form:
                                        Interactive:
                                            keyword"description"=(checklist)
                                            keyword'description'=(checklist)
                                        Passive:
                                            keyword
                                            keyword=
                                            keyword=default1
                                            keyword=default1/default2
                                            keyword=/default2
                                            keyword=#DATA
                                            keyword=#FILE
                                        keyword      - 1-10 alphanumeric
                                                        characters
                                        description - parameter prompt
                                                        (see title above)
                                        checklist   - a list of acceptable
                                                        values and the
                                                        parameter syntax
                                        default1    - 1-40 chars if pi
                                                        is omitted
                                        default2    - 1-40 chars if pi is
                                                        specified without
                                                        value or with the
                                                        value pi
                                        #DATA       - the name of an
                                                        unnamed .DATA file
                                        #FILE       - the file containing
                                                        pname

                        selections - the menu selections in the form:
                                    n1"desc1",n2"desc2",...,nn"descn"
                                    where ni      - integer (1-10 digits)
                                                        identifying the menu
                                                        selection
                                                desc1 - the menu item descrip-
                                                        tion (see title above)

                        ck         - comment keyword (1-10 characters)
```

.PROMPT,text.

.PROMPT=text.

Specify the text for the general request for input in a procedure.

.SET,keywd_1=strex_1,...,keywd_n=strex_n.

Build new procedure parameters.

.*comment

A comment in a procedure.

Example: .PROC,myproc,....

```
...
REVERT...myproc
.*
.* created 88/04/12
.* last modified 88/05/20 (add "PW" parameter)
.*
.* End of myproc
```

*** DTRC Procedure Library ***

Public-access procedure library PROCFIL has been added to NOS at DTRC and will be searched if you do not have a local or permanent file named PROCFIL.

The following are the public procedures available at DTRC as of the date of this page. For the current list, type BEGIN,CONTENT. See Appendix I for further information on individual procedures.

AUX	(interactive) Turn the auxiliary printer on or off.
AUXPRNT	(interactive) Print a local file on the auxiliary printer.
CONTENT	List the contents of the public procedure file.
COPYBFR	Restore "randomness" to a random file which was copied using a sequential copy such as COPYBF.
FICHE	Route a copy of a local file to the microfiche (COM).
FLR	Compile FTN5, load with up to 2 libraries, and run.
GETASC	Convert an 8/12 ASCII indirect or direct permanent file to 6/12 Display Code.
GRIPE	Submit a gripe or suggestion.
HELP	Obtain help on a command or topic (very limited).
JOB COST	Display the job cost, number of SRUs and CPU time in the dayfile and file OUTPUT.
MSAUDIT	Sorted CATLIST.
NEWCHRG	Set the current charge number on all/some files.
NEWRTNS	Announcement of new routines.
NEWS	Display the current news.
OFLREQ	Off-line Request Generator.
OLDNEWS	Display the old news.
PUTASC	Convert a 6/12 Display Code file to 8/12 ASCII and store as an indirect or direct file.
RATES	Display the current computer rates.
TAPDMP	Driver to execute TAPEDMP to dump/analyze a magnetic tape. or

TPAUDIT Obtain a list of the tapes assigned to you.

TPGET Get up to 3 tapes to use on the CDC CYBER 860 or DEC
VAXcluster.

TPRLS Release up to 5 tapes.

WHOAMI Display your User Initials and account number.

XEROX Route a copy of a local file to the Xerox 8700.

*** Sample Procedure ***

The following illustrates a simple interactive procedure to compile a Fortran program and, optionally execute the program.

```
.PROC,F5*I"Compile and execute Fortran 5",  
    I 'Input'      = ( *F, *N=INPUT ),  
    B 'Binaries'   = ( *F, *N=LGO ),  
    L 'Output'     = ( *F, *N=OUTPUT ),  
    LO 'List options' = ( *N=0,0,0,R,A,M,S ),  
    GO 'Execute'    = ( *N= ).  
FTN5,#I=I,#B=B,#L=L,#LO=LO.  
IF,$GOS.NE.$$LGO.  
REVERT,NOLIST...F5
```

Invoking this procedure with

f5,?

causes the following dialog:

```
PARAMETERS FOR F5 ARE  I, B, L, LO, GO  
Input? test  
Binaries? bfile  
Output? <press the RETURN key for the default>  
List options? s  
Execute? y
```

The generated FTN5 statement will be:

```
FTN5,I=TEST,B=BFILE,L=OUTPUT,LO=S.
```

Since the value of GO (Y) is non-null, "LGO." will be executed.

***** Program Libraries *****

Source programs and data may be in separate datasets or may be stored and maintained in program libraries. UPDATE creates and maintains these libraries, which may be display code or ASCII (8/12).

*** UPDATE ***

UPDATE is a program for creating and modifying a program library (PL). In addition, UPDATE will extract individual modules for input to a compiler or other program.

By default, 72 columns of information are retained. Fifteen additional characters are retained for each line: an 9-character identifier, a 6-digit sequence number, i.e., id_seq, and is often referenced as id.seq.

UPDATE supports two kinds of text modules or decks:

- a regular deck (beginning with a DECK directive)
- a common deck (beginning with a COMDECK directive) which may be included in decks with a CALL directive

Each type includes all lines following the deck directive until the next deck or modification directive.

History information is retained allowing the deletion, modification, or restoration of previous modifications.

See Appendix I for a description of the UPDATE control statement parameters.

*** UPDATE Directives ***

An UPDATE directive has the following format:

m directive_name [parameters]

where m is the master character (default: asterisk (*)). There are five categories of directives.

** DECK and COMDECK **

*DECK deck (*DK)

First line of a new deck. <deck> is up to 9 characters except comma, period, blank, colon, equals.

*COMDECK cmdk (*CDK)

First line of a new common deck.

** Compile File **

*CALL cmdk (*CA)
Include the contents of a common deck.

*COMPILE p1,p2,...,pj.pk,...,pn (*C)
Write one or more decks, including a range (pj.pk), to the compile and/or source datasets. Use UPDATE,K to force the output order.

*CWEOF (*CW)
Write an EOF on the compile dataset if anything was written since the last EOF.

*WEOF (*W)
Write an EOF on the compile dataset.

*WIDTH linelen,idlen (*WI)
Change the data and id length (default: 72,4).

*DO, *DONT, *IF, and *ENDIF are also available.

** Modification **

*ADDFILE lfn,name (*AF)
Read creation directives and text from file lfn and insert after the specified deck or line.

*BEFORE id.seq (*B)
Insert before a line.

*CHANGE oldid,newid,...,oldid,newid
Change correction set identifier.

*COPY dk,id1.seq1,id2.seq2 (*CY)
Copy a range of lines from deck or comdeck <dk>.

*DELETE id1.seq1 (*D) <-- one line
*DELETE id1.seq1,id2.seq2 <-- a range of lines
*DELETE id1.seq1,.seq2 <-- same (short form)
Delete a line or a range of lines.

*IDENT ident (*ID)
*IDENT ident,B=num,K=id,U=id
Identify a set of modifications. You can specify a sequence number bias, and require that other modification sets be known (K=) or unknown (U=).

*INSERT id.seq (*I)
Insert after a line.

*MOVE dk1,dk2 (*M)
Move deck <dk1> to follow deck <dk2>.

*PURDECK dk,dk2,...,dkj.dkk,...,dkn (*PD)
Permanently remove decks.

*PURGE id1,id2,...,idj.idk,...,idn (*P)
Remove the effect of a modification set (idi), a range of
datasets (idj.idk), or a set and all following (idn=*).

*RESTORE id1.seq1 (*R) <-- one line
*RESTORE id1.seq1,id2.seq2 <-- a range of lines
Restore a line or a range of lines.

*SEQUENCE id1,id2,...,idj.idk,...,idn.. (*S)
Resequence active lines and purge inactive lines in the
specified decks.

*YANK id1,id2,...,idj.idk,...,idn
Temporarily delete a deck, comdeck, or modification set
previously yanked.

*YANKDECK dk1,dk2,...,dkj.dkk,...,dkn
Temporarily deactivate decks.

*SELPURGE, and *SELYANK are also available.

** File Manipulation **

*COPY name,id1.seq1,id2.seq2,1fn (*CY)
Copy a range of lines from deck or comdeck <name> to file
<1fn>.

*READ 1fn (*RD)
Read input from another file.

*REWIND 1fn
Rewind a file.

*SKIPF 1fn,n
Skip record(s) in a local file.

** Input Stream Directives **

*ABBREV
Resume recognition of abbreviations.

*ENDTEXT (*ET)
End a *TEXT section.

*LIST (*L)
Resume listing input lines. UPDATE,L=0 overrides *LIST.

*NOABBREV (*NA)
Do not check for abbreviation.

*NOLIST (*NL)
Stop listing input lines.

*TEXT (*T)
Treat all statements between *TEXT and *ENDTEXT as text.

*SKIP and *ENDSKIP are also available.

** Special **

*LIMIT n (*LT)
Limit the output listing to n lines.

*/comment
A comment line.

*DECLARE, *DEFINE, and *PULLMOD are also available.

*** Examples ***

1) Create a PL:

```

jobnam1,....
USER,user,pw.
CHARGE,....
UPDATE,P=0,C=0.          <-- no OLDPL or COMPILE
SAVE,NEWPL=mypl.        <-- create indirect file
<eor>
*DECK DECK1
  lines for deck DECK1
*DK DECK2
  lines for deck DECK2
*DK DECK3
  lines for deck DECK3

```

2) Interactively extract, compile and execute deck DECK2 from PL MYPL:

```

GET,OLDPL=mypl.          <-- get indirect file
NOTE,uin./*COMPILE deck2
UPDATE,I=uin.
FTN5,I.
LGO.

```

3) Create a PL using a common deck, compile and execute:

```

jobnam3,....
USER,user,pw.
CHARGE,....
PURGE,mypl/NA.
DEFINE,NEWPL=mypl.      <-- direct file
UPDATE,P=0.             <-- no OLDPL
FTN5,I.
LGO.
<eor>
*COMDECK COM3
  common / mycom / a, b
  real a, b
*DK PROG3
  program prog3
*CALL COM3
  call sub
  print *, 'a,b=', a, b
  end
*DECK SUB
  subroutine sub
*CA COM3
  a = 1.
  b = 2.
  return
  end
<eoi>

```

- 4) Update old source library to new, compile all decks and execute:

```
jobnam4,....  
USER,user,pw.  
CHARGE,....  
GET,OLDPL=mypl.          <-- get indirect old library  
UPDATE,F,N.  
FTN5,I.  
LGO.  
REPLACE,NEWPL=mypl.      <-- replace indirect old library  
<eor>  
*IDENT DS0620            <-- correction must be unique (initials,date)  
*INSERT ALONE.57         <-- correct deck ALONE by insert after line 57  
    (Fortran statements)  
*DELETE FOUR.12,13      <-- correct deck FOUR replacing lines 12-13  
    (new lines to replace deletions - optional)  
<eor>  
    (data lines, if any)  
<eoi>
```

- 5) Select routines from source subroutine library and compile with your own program:

```
jobnam5,....  
USER,user,pw.  
CHARGE,....  
FTN5.                    <-- compile your own programs  
ATTACH,thatpl/UN=NSYS.  
UPDATE,P=thatpl,Q,L=0.  
FTN5,I.  
LGO.  
<eor>  
    (own Fortran decks)  
<eor>  
*C rtn1,rtn6.rtn8        <-- select decks RTN1, 6, 7, 8 from library  
<eor>  
    (data records, if any)  
<eoi>
```

***** Object Libraries *****

LIBEDIT and LIBGEN are utilities for creating and maintaining libraries of absolute and relocatable object modules. These libraries can then be used by the loader to locate the program to execute or the subprograms to be loaded with your program.

See Appendix I for the LIBEDIT and LIBGEN control statements.

*** LIBEDIT Directives ***

The following are used in the descriptions of the LIBEDIT directives:

rid - record identifier

format	meaning
-----	-----
type/name	the record has this type and name
name	the record has this name and the default type
*	end-of-file (*BEFORE only)

gid - group identifier

format	meaning
-----	-----
type/name	the record with this type and name
name	the record with this name and the default type
type1/name1-type2/name2	a group of records
type1/name1-name2	a group of records of type1
name1-name2	a group of records with the default type
type/name-*	all records of this type beginning with <name>
name-*	all records of the default type beginning with <name>
type/*	all records of this type
*	all records
0	insert a zero-length record

The following are some of the LIBEDIT directives. Directives start with an asterisk in column 1, followed by the directive name (or abbreviation). Directives can be continued (gid entries cannot be split). For example:

```
*BEFORE,ov1/p1,ov1/p2
ov1/p3
```

*ADD LIBn,gid1,gid2,...

Append records to a record group.

Parameters: LIBn - a record group (from a CATALOG listing)
(1 <= n <= 63)

gidi - records from the current replacement file
to be appended

*BEFORE rid,gid1,gid2,... (*B)

Insert records before a specified record.

*BUILD dname

Build a directory at the end of the new file.

Parameters: dname - the name for the directory record
(1-7 alphanumeric)

*COMMENT rid comment

Add a comment to the prefix table.

Parameters: comment - up to 70 characters with excess
truncated

*COPY Copy the new file to the old file after editing.

Remarks: *COPY is the same as LIBEDIT,...,C.

*DATE rid comment

Add the date and a comment to the prefix table.

Parameters: comment - up to 70 characters with the excess
truncated

*DELETE gid1,gid2,... (*D)

Do not copy the specified records to the new file.

*FILE lfn The name of the file containing the replacement records.

Parameters: lfn - use * for the replacement file from the
LIBEDIT command (B=)
(default: LGO)

*IGNORE gid1,gid2,...

Ignore specified records in the replacement file.

Examples: *FILE myrecs
IGNORE D-

^-- ignore all records from D to the
end-of-file

*INSERT rid,gid1,gid2,... (*I, *AFTER, *A)

Place the replacement records after the specified groups in
the new file.

*LIBGEN record_name

Generate a user library (using LIBGEN) after processing.

Parameters: record_name - the name of the new user library
directory record

Remarks: *LIBGEN overrides *VERIFY.

*LIST list_file,list_opt

Specify the list file and the list options.

Parameters: list_file - same as LIBEDIT,L=
list_opt - same as LIBEDIT,LO=

*NEW newfile

Specify the name of the new file.

Parameters: newfile - same as LIBEDIT,N=

*NOINS Prevent the insertion of unreplaceable records.

Remarks: same as LIBEDIT,NI

*NOREP lfn1,lfn2,...

Do not automatically replace records from the specified files.

Remarks: Records from these files can be copied to the
new file only by using *AFTER, *BEFORE, *INSERT,
or *REPLACE.

*NOREW Do not rewind the old or new files.

Remarks: Same as LIBEDIT,NR.

*OLD oldfile

Specify the name of the old file.

Remarks: Same as LIBEDIT,P=.

*RENAME rid,name

Rename a record.

Parameters: rid - the name of the replacement or old file
record to be renamed

name - the new name

*REPLACE gid1,gid2,...

Replace old file records with records from the replacement
file.

Examples: The old and replacement files each contain
records A, B, C, D. To replace only C and D, use
either of the following:

*FILE replfyl
*NOREP replfyl
*REPLACE C-D

*FILE replfyl
*IGNORE A-B

*REWIND lfn

Rewind a file before and after editing.

*TYPE type (*NAME)

Set the default record type.

Parameters: type - the record type (ABS, OPL, OVL, PROC, REL,
TEXT, ULIB)

*VFYLIB Verify the new file against the old file using VFYLIB.

Remarks: Overridden by *LIBGEN.

*** DTRC Object Libraries ***

The following object libraries have been added to NOS at DTRC:

DTLIB/UN=NSYS - Subprograms written or maintained by the
Computer Center

To use: ATTACH,DTLIB/UN=NSYS.
LDSET,LIB=DTLIB. -or- LIBRARY,DTLIB.
LGO.

UTILITY/UN=NSYS - Programs written or maintained by the
Computer Center

To use: ATTACH,UTILITY/UN=NSYS.
LIBRARY,UTILITY.
prognam.

*** Examples ***

- 1) Create a library of subprograms.

ATTACH,mysubss.
FTN5,I=mysubss,OPT=2,L=0.
PURGE,mysubs/NA.
DEFINE,mysubs/CT=PU.
LIBGEN,P=mysubs.

- 2) Create a library of all subprograms from an UPDATE library.

ATTACH,OLDPL=mypl.
UPDATE,F.
FTN5,I,L=out2,OPT=2.
PURGE,mysubs/NA.
DEFINE,mysubs/CT=PU.
LIBGEN,P=mysubs.
ROUTE,out2,DC=PR.

- 3) Add a subprogram to an existing library and have the output list in alphabetical order.

Direct files

jobnam3.
USER,user,pw.
CHARGE,....
FTN5,OPT=2.
ATTACH,subs.
PURGE,NEW/NA.
DEFINE,NEW.
LIBEDIT,P=subs,U,I=0.
PURGE,subs.
CHANGE,subs=NEW/CT=PII.
<eor>
<FTN5 subprograms source>

Indirect files

jobnam3.
USER,user,pw.
CHARGE,....
FTN5,OPT=2.
GET,subs.

LIBEDIT,P=subs,U,I=0.
REPLACE,NEW=subs.

<eor>
<FTN5 subprograms source>

- 4) Delete subprogram BADSUB from an existing library.

GET,OLD=subs.
LIBEDIT,B=0,Z.*DELETE REL/badsub
REPLACE,NEW=subs.

***** Loader *****

The loader is responsible for loading all programs, resolving any external references, and optionally initiating execution.

Once loading of a program is started, no other control statements may interrupt the load sequence. For instance, a 'LOAD,lfn.' statement may only be followed by another 'LOAD,lfnl.' or one of the loader control statements or MAP, REDUCE or others listed in the Loader Reference Manual.

*** Types of Loading ***

Loading differs according to whether the input is one or more object modules or a single memory image module. Loading of object modules can involve overlay or segment generation and can result in one or more memory image modules. A basic load results in one memory image (absolute) module.

Object module loading	One or more object modules are loaded, libraries are searched for the external references, addresses are adjusted, and a memory image module may be produced.
Memory image loading	This is a special case because no external linkage or address adjustment is required.
Basic loading	All object code is loaded at the same time, resulting in a single memory image module.

Segmentation

For large programs, segmentation should be used to divide the program into several memory image modules, called segments.

With segmentation, only those portions of the program needed at a given moment are in memory. Different memory image modules reside in the same area of memory at different times. Depending on execution requirements, different memory image modules are loaded dynamically.

Features:

- Segmentation allows any number of levels, limited only to a total of 4093 segments.
- After segments have been generated, their loading is automatic.
- References between segments may be upward or downward.
- At execution time, a resident program is loaded which loads the root segment. Thereafter, it loads the other segments as required.

Overlay generation

An overlay is a collection of executable programs which are called into memory at execution time, according to an overlay structure which is defined in the source code.

Overlay capsule generation

For large programs, overlay capsules may be used to divide the program into an absolute main program and one or more capsules which are loaded and unloaded by the user.

Because overlays and overlay capsules require statements in your program to cause the overlaying to take place, they are not recommended. Instead, use segmentation, which is controlled by directives external to your program.

*** Loader Control Statements ***

See Appendix I for the syntax of the following control statements used to load a program.

EXECUTE Complete loading, fill unsatisfied references by system (and user) library search, generate load map and execute the program.

LDSET Set any of several loader options for the current load only.

LGO Load and execute the default compiler binary output file.

LIBLOAD Load modules from specified library which contains the specified entry points.

LOAD A list of files whose contents are to be loaded.

name Load and execute binary program or procedure in local file <name>.

NOGO Complete the loading of a program, including generating load map, but do not execute.

SATISFY Satisfy unsatisfied externals prior to normal satisfaction at load completion.

SLOAD Selectively load modules from local file <lfm>.

In addition, the following Loader-related control statements are also available:

LIBRARY Specify a set of global libraries to be searched for externals and programs and the order in which the libraries are to be searched.

MAP Specify the global default option for load maps.

REDUCE Turn the reduce flag on or off.

RFL Set field length for the next program execution.

*** Segmentation ***

To implement segmentation, a separate directive record is prepared to describe the tree structure. The modules will be loaded automatically as needed. Job field length is adjusted dynamically if the program has no blank common, has no level statements and is not in RFL mode.

All necessary Record Manager routines must be in the root segment. Other LOAD and LDSET statements follow the SEGLOAD statement:

SEGLOAD,I=1fndir,B=1fnabs,LO=1fnout.

** SEGLOAD Directives **

x TREE y

To define a tree structure.
<y> may be comma-separated list of other trees (pre-defined), segments or names of individual subprograms to be assigned a common starting address.

x TREE f-(c,d)

To indicate branching of the tree use -, then all following items are enclosed in parentheses.

c INCLUDE a,b

To assign programs <a> and to segment <c>.
Copies of a routine may be in different segments.

c GLOBAL com1,com2

To establish named commons at desired segment.
Reference name to left of directive must be defined by a previous directive.

c GLOBAL com1,com2-save

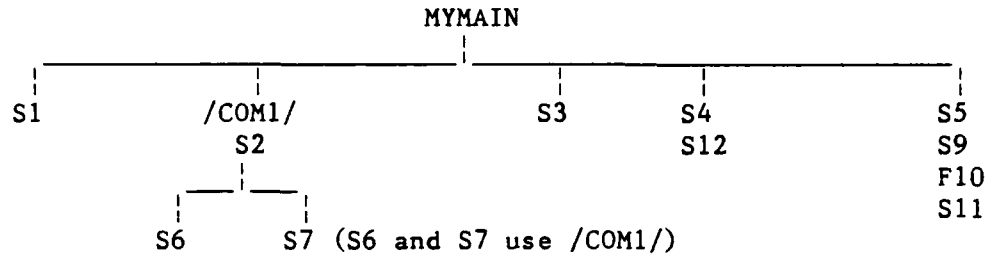
To save global block on disk for later calls to the segment which contains it.

END ept

Should be the last directive in the record, where <ept> is the entry of the main program in the root segment.
Non-fatal error if omitted.

** Sample Tree Diagram **

A block data subprogram defines common /COM1/ which is to be loaded with program S2. /COM1/ is also referred to by S6 and S7.



```

SEG  INCLUDE  S2,BLKDAT.
SEG  GLOBAL   COM1
PLUM TREE     SEG-(S6,S7)
PEAR TREE     MYMAIN-(S1,PLUM,S3,S4,S5)
S5    INCLUDE  S9,F10,S11
S4    INCLUDE  S12
      END
  
```

By using nested parentheses one TREE directive may be eliminated.

*** Segmentation Cautions ***

1. To develop a segmented job, several runs may be required, so relocatable object code should be cataloged. Common blocks and Record Manager routines may need to be INCLUDED in lower segments to operate properly.

2. The load map must be checked carefully for any duplicate common block entries. Each common block which is referenced in more than one segment must be put into a global at the nearest-to-the-root segment. If any common block appears more than once without "safe", a global is required to eliminate duplicate storage areas. If input/output is performed in several segments, some Record Manager common blocks may be multiply defined (e.g., AOB.RM or Q8.IO.).

Subfields in SEGLOAD directives which contain any of the special characters , - () or which start with a \$ must be defined as literals (i.e., delimited by \$...\$). Embedded \$ is represented by \$\$, thus, \$TAN must be specified as \$\$\$TAN\$.

When Record Manager common is global to a root segment, the loader may detect errors in initializing. If so, an INCLUDE directive will be required to move the RM routines to that segment (e.g., 'MYMAIN INCLUDE INCOM=').

3. Directives must not go beyond column 72 of a line. They may be broken almost anywhere and continued on the next one or more cards/lines. The continuations have a comma (,) in column 1 as the continuation signal, then the directive is continued starting in column 2.

Continued directives should be avoided, if possible, to improve readability.

4. FTN5 users should avoid passing external references as subprogram arguments. When the external is not in the root segment or the same segment as the call, execution will generate the fatal error message 'NON-EXECUTABLE WORD LOADING A SEGMENT'.

*** Compile, Load and Catalog Absolute Program ***

** Simple Load **

```

jobname,....
USER,user,pw.
CHARGE,....
FTN5,OPT=1.
LOAD,LGO.
NOGO,myprog.          <-- absolute module into MYPROG
REPLACE,myprog.
<eor>
    program myprog
    ...
<eoi>

```

** SEGLOAD **

```

jobname,....          name/code
USER,user,pw.
CHARGE,....
FTN5,OPT=1.
REPLACE,LGO=mysglgo.  <-- save relocatable modules for
                        possible re-segmentation
PURGE,mysegnu/NA.     <-- MYSEGNU for absolute segments
DEFINE,mysegnu.
SEGLOAD,B=mysegnu.
LOAD,LGO.
NOGO.                 <-- absolute segments onto MYSEGNU
PURGE,mysegl/NA.      <-- replace old
CHANGE,mysegl=mysegnu. <-- copy of MYSEGL
<eor>
    < FTN5 source program >
<eor>
    < SEGLOAD directives >
<eoi>

```

** Interactive Simple Execution **

```

GET,mysor.
FTN5,I=mysor,L=myout,OPT=1.
LDSET,MAP=S.          <-- to see any missing routines
LGO.
...
ROUTE,myout,DC=PR.    <-- print the compilation listing

```

*** Object Program Execution ***

** COBOL5 ***

The following system parameters may be added to the execution statement of a COBOL 68 program to display in the dayfile certain information about the execution:

- *CORE Displays the maximum amount (in octal) of memory used.
- *MSGs Displays the Sort/Merge summary messages.
- *TIME Displays the execution CPU time.

For example, LGO,*TIME,*CORE,*MSGs.

** Fortran **

The following system parameters may be added to the execution statement of a Fortran 77 program to control the execution:

*DA=i+j+k+l+m+n+o

Specify the subscript limits for Post Mortem Dump. For example, LGO,*DA=4+2.
dumps 1- and 2-dimensional arrays with the maximum first dimension of 4 and second of 2.
(default: *DA=20+2+1+1+1+1+1)

*OP=op

Specifies the format and destination of the Post Mortem Dump.

op	meaning
---	-----

A	all active routines are included in the dump
F	output to file PMDUMP if OUTPUT is connected (interactive only)
T	a condensed form of the output is displayed at the terminal if OUTPUT is connected (interactive only)

For example, LGO,*OP=AF.

(defaults: interactive: *OP=F; batch: *OP=)

*PL=pl

The maximum number of lines written to file OUTPUT. This overrides the value specified or implied (5000) at compilation time by the PL parameter of the FTN5 statement. This is placed after any file replacements in the execute statement
For example, LGO,*PL=10000.

LGO(,MYFILE1,MYFILE2,*PL=600)

User parameters follow any system parameters.

***** Other Software *****

*** Accessing Other Software ***

Programs obtained from other vendors are normally execute-only. To access them, you normally need

ATTACH,program/UN=un,M=E. <-- A below

-or-

GET,program/UN=un. <-- G below

where un currently is APPLLIB, LIBRARY or NSYS.

*** UN=APPLLIB ***

As of the date of this page, UN=APPLLIB contains:

GPSS A General Purpose Simulation System

PERT78 A Pert/Time

*** UN=LIBRARY ***

As of the date of this page, UN=LIBRARY contains:

BETONOS A NOS/BE to NOS command help (HELPBE) and NOS/BE file loader
(BELOAD)

DTLIB A DTRC subprogram library

HOTSPOT G Analyze a program for inefficient code

PROCFIL G DTRC procedure library

SIMII5 A Simscript II.5

*** UN=NSYS ***

As of the date of this page, UN=NSYS contains:

CALCFN G Calcomp Functional Package

CALC936 G Calcomp 936 Subroutine Package (7-track tapes only)

DISSPLA G DISSPLA Graphic Subroutine Package

IMSLM A IMSL 10 Mathematical Subroutine Library

IMSLSS A IMSL 10 Special Function and Statistical Subroutine Library

LINPACK G Simultaneous Linear Algebraic Equation Solver Package

UTILITY G DTRC program library

***** Appendix H *****

*** CDC NOS/VE SCL Commands ***

CDC NOS/VE SCL commands have the following general syntax:

```
statement name -or-  
plural statement name -or-  
statement name abbreviation  
    parameter name=value list  
    parameter name=value list  
    ...  
    parameter name=value list
```

```
function name  
    (parameter name:value list,  
    ...  
    parameter name:value list)
```

statement name -or- plural statement name
is the name of the utility to be performed. It has the form
aaaaa_bbb_ccc_ddd_eee_...

statement name abbreviation
is the short form of the statement name and is the first three
characters followed by the first character after each "_",
e.g.,
aaabcde...

parameter name=value list
(parameter name:value list, ... parameter name:value list)
is the list of parameters for the utility or function.

function name
is the name of the function to be performed. E.g.,
\$STRING
 (VALUE: any,
 FORMAT:keyword,
 MAX_STRING:integer)

*** Value Descriptors ***

The following value descriptors are used in this appendix:

boolean One of:
TRUE (synonyms: ON, YES)
FALSE (synonyms: OFF, NO)

date_ISO The date in ISO format:
yy-mm-dd

date_time The date (and time) in one of the following formats:
YYYY-MM-DD.hh.mm.ss
YYYY-MM-DD <-- assumes .00.00.00 (midnight)

file A filename.

keyword A specific word or phrase.

list of ...
One or more value. If more than one, they are separated by
a comma or space.

name An SCL or user name.

string A string of characters.
aa...a - 1 or more alphabetic characters
axx...x - 1 or more alphanumeric characters, the first
alphabetic
xxx...x - 1 or more alphanumeric characters
nn...n - 1 or more decimal (unless otherwise stated) digits
nn...nB - 1 or more octal digits
nn...nD - 1 or more decimal digits

*** Condition Handling ***

You may establish handlers for the following conditions (severity error level ERROR or greater):

PROGRAM_FAULT	Program terminated in error
LIMIT_FAULT	Job resource limit reached
COMMAND_FAULT	Command syntax error
ANY_FAULT	Any of the above conditions occurred
EXIT	A procedure is terminating
PAUSE(INTERRUPT)	Pause break (terminal interrupt) was entered

Without a condition handler, a batch job will terminate and an interactive job will prompt you for another command.

Within the condition handler, you may include

CONTINUE	to continue the job with the command following the one that caused the error
CONTINUE RETRY	to retry the command that caused the error

When an error occurs, the current block is searched for the most recent WHEN block for that condition. For example,

```
WHEN PROGRAM_FAULT DO
  PUT_LINE ' Program terminated abnormally.'
WHENEND
```

The CANCEL command is used to cancel one or more condition handlers.

*** Some Common Parameters ***

The following parameters are used in many SCL commands. If they have a different meaning or a special condition, it will be mentioned in the individual description.

BINARY_OBJECT=file (BO, BINARY, B)

The file to receive the binary object code produced by a compiler. To omit generation of the binary, use B=\$NULL.

(default: \$LOCAL.LGO)

DEBUG_MODE=boolean (DM)

Specify if debug mode is to be used.

(default: OFF)

ERROR=file (E)

The file to which error messages are written.

(default: \$ERRORS)

ERROR_LEVEL=keyword (EL)

The minimum error severity level to be reported.

ab option

-- -----

I INFORMATIONAL

T TRIVIAL

W WARNING

F FATAL

C CATASTROPHIC

NONE

Each command supporting this will list the options available and the default.

FAMILY=name or string (F)

The disk family where the file is stored. At DTRC, there is only one family (NVE), so this parameter can be omitted.

FILE=file (F)

The file to be processed.

FILE=list of file (F)

One or more files to be processed.

INPUT=file (I)
The input file.

(defaults: compilers: \$INPUT
others: \$COMMAND)

JOB_NAME=name (JN)
The name of the batch job.

LIST=file (L), Nam
The listing file. To omit generation of this file, use
L=\$NULL.

(default: \$LIST)

LOCAL_FILE_NAME=name (LFN)
The local file name used by subsequent commands to refer
to an attached file.

OUTPUT=file (O)
The output file. To omit generation of this file, use
O=\$NULL.

(default: \$OUTPUT)

PASSWORD=keyword or name or string (PW) (attribute: SECURE)
The password for access to a file.

PROLOG=file (P)
Some commands allow you to execute subcommands each time
you execute the command. Normally, these subcommands are
in this file.

(default: \$NULL)

RUNTIME_CHECKS=list of keyword (RC)
Used by compilers to control execution-time checking.

rc	meaning
R	runtime range checking for character substring expressions
S	runtime range checking for substring expressions
NONE	no options
ALL	both R and S

(default: NONE)

SCOPE=keyword (S) (attribute: BY_NAME)
The scope of the variable.

keyword	meaning
LOCAL	local to the current block
XDCL	externally declared
XREF	externally referenced (cannot have an initial value)
ENVIRONMENT	local to the block in which it is defined
JOB	global to the job block
TASK	local to the current block (if a task block) -or- in the task block that most directly called the current block -or- in the job block
UTILITY	local to the current block (if a utility block) -or- in the utility block that most directly called the current block -or- in the job block
PUSH	if defined, it is suspended (pushed) if not defined, is it created with scope of ENVIRONMENT

Each command supporting this will list the options
available and the default.

STATUS=status variable

Causes continues execution when an abnormal condition
occurs (except command syntax error). Without STATUS,
the succeeding commands in the block are skipped.

Before using the STATUS parameter, you must define a
variable of type status. For example,

status=my_status

Variable MY_STATUS may then be interrogated and the flow
of commands altered as desired.

The status variable is a record and contains:

NORMAL - TRUE => the command completed without an
error

FALSE => either the command failed, or it
completed successfully but there
is more information about the
completion

CONDITION - the condition code of the diagnostic
message -- undefined if the NORMAL field is
TRUE.

TEXT - a string (up to 256 characters) with the diagnostic message -- undefined if the NORMAL field is TRUE.

STATUS has attributes: BY_NAME, VAR.

SUBSTITUTION_MARK=keyword or string (SM)

A single character used within a statement to delimit the text to be substituted. It cannot be any character that is a valid SCL name. Corresponding pairs must appear on the same line, if not, an end-of-line is treated as the substitution mark. Two consecutive marks appear as a single mark in the substituted string.

NONE - no substitution
(default: NONE)

SYSTEM_JOB_NAME=name (SJN)

The system-supplied name of a batch job.

TERMINATION_ERROR_LEVEL=keyword (TEL)

Minimum severity level for returning abnormal status

ab option

--
I INFORMATIONAL
W WARNING
F FATAL
C CATASTROPHIC

(normal default: FATAL)

TITLE=string (T) (windowing)

The title of the window.

Each command supporting this will list the default.

TITLE has attribute: BY_NAME.

USER_JOB_NAME=name (UJN, JOB_NAME, JN)

The user-supplied name of a batch job.

WIDTH=integer (W) (windowing)

The width of the window (in characters).

Each command supporting this will list the range and default.

WIDTH has attribute: BY_NAME.

X=integer (windowing)
The X-coordinate of the upper left-hand corner of the window.
(default: 5; range: 2-80)

X has attribute: BY_NAME.

Y=integer (windowing)
The Y-coordinate of the upper left-hand corner of the window.
(default: 5; range: 2-30)

Y has attribute: BY_NAME.

*** Summary of CDC NOS/VE SCL Commands ***

The following are NOS SCL statements, except as indicated by:

(DTRC) A command, procedure or program added at DTRC

(UTILITY subcommand)

A subcommand used in the definition of a utility

(<utility> subcommand)

A subcommand of the utility <utility>

(VEIAF) NOS/VE InterActive Facility

SCL statements for certain NOS/VE features are indicated by:

(CRM) Cyber Record Manager

(Loader) Loader control statements

In the syntax descriptions, parameters preceding "-----" are required; those following are optional. If a command description has several "-----" lines, the first means the parameters following it are optional; the second means those following are required; the third that those following are optional; etc.

Because parameters may be specified either positionally or by keyword, all parameters are listed in the Syntax section, but are not necessarily described in the Parameters section. Some common parameters are described in the preceding section of this Appendix.

ANALYZE_OBJECT_LIBRARY

Begin an ANALYZE_OBJECT_LIBRARY utility session.

Syntax: ANALYZE_OBJECT_LIBRARY -or-
ANAOL

L or LIBRARY=file
STATUS=status variable

Parameters: L - object library or object file to be analyzed
(default: you must use USE_LIBRARY subcommand
to specify it)

Subcommand prompt: AOL/

Similar commands: NOS: ITEMIZE

ANALYZE_PROGRAM_DYNAMICS

Measure program execution characteristics and restructure program as a single load module.

Syntax: ANALYZE_PROGRAM_DYNAMICS -or-
 ANAPD
 TT or TARGET_TEXT=file
 RM or RESTRUCTURED_MODULE=file

 F or FILES or
 FILE=list of file
 P or PARAMETER=string
 L or LIBRARIES or
 LIBRARY=list of file
 M or MODULES or
 MODULE=list of program-name
 SP or STARTING_PROCEDURE=program_name
 RMN or RESTRUCTURED_MODULE_NAME=program_name
 RC or RESTRUCTURING_COMMANDS=file
 STATUS=status variable

ANY_MAIL (DTRC) Check for mail.

Syntax: ANY_MAIL -or-
 ANYM

 DO or DISPLAY_OPTION=keyword
 STATUS=status variable

Parameters: DO - display options

ab	option	meaning
-----	-----	-----
B	BRIEF	only a message
F	FULL	list of unread letters

(default: B)

See also: CHECK_FOR_MAIL

Similar commands: VMS: MAIL

Examples: /any_mail
 You have new mail
 =====

 /anym
 You have no unread mail
 =====

/anym f

1. date time sender subject size(bytes) codes

2. date time sender subject size(bytes) codes

...

^-- codes: C (confirm delivery)

R (reply requested)

ATTACH_FILE

Attach a file to a job.

Syntax:

ATTACH_FILE -or-

ATTF

F or FILE=file

LFN or LOCAL_FILE_NAME=name

AM or ACCESS_MODE or

ACCESS_MODES=list of keyword

EEPN or ERROR_EXIT_PROCEDURE_NAME or

EEN or ERROR_EXIT_NAME=name

EL or ERROR_LIMIT=integer

FB or FREE_BEHIND=boolean

JWC or JOB_WRITE_CONCURRENCY=boolean

MC or MESSAGE_CONTROL=list of keyword

OP or OPEN_POSITION=keyword

PW or PASSWORD=keyword or name

PR or PRIVATE_READ=boolean

RV or RECORDED_VSN=list of: string or name

SA or SEQUENTIAL_ACCESS=boolean

SM or SHARE_MODE or

SHARE_MODES=list of keyword

TS or TRANSFER_SIZE=integer

VOA or VOLUME_OVERFLOW_ALLOWED=boolean

W or WAIT=boolean

STATUS=status variable

See also:

CREATE_FILE; DETACH_FILE

Similar commands:

COS: ACCESS; ACQUIRE

NOS: ATTACH; FILE; GET

VMS: no local file concept

Examples:

/attf,in,prog_data

ATTACH_JOB

Reconnect your terminal to a previously disconnected job.

Syntax:

ATTACH_JOB -or-

ATTJ

JN or JOB_NAME=name

STATUS=status variable

Remarks: The jobnames of any disconnected jobs are listed at login time.

Examples: /attj \$0860_0488_aap_8075

AUXPRINT (DTRC) Print a file on your auxiliary printer (the one attached to your terminal.

Syntax: AUXPRINT -or-
AUXP
F or FILE=file

STATUS=status variable

Parameters: F - the file to be printed

See also: PRINT_FILE

Similar commands: NOS: BEGIN,AUXPRNT
VMS: AUXPRINT

Examples: /AUXPRINT my_file

BACKUP_PERMANENT_FILES

Begin a utility session to back up permanent files and catalogs.

Syntax: BACKUP_PERMANENT_FILES -or-
BACKUP_PERMANENT_FILE -or-
BACPF
BF or BACKUP_FILE=file

L or LIST=file
STATUS=status variable

Subcommand prompt: PUB/

Similar commands: NOS: RECLAIM; CATLIST
VMS: BACKUP

BASIC Compile a BASIC source program.

Syntax: BASIC

I or INPUT=file
B or BINARY=file
L or LIST=file
LO or LIST_OPTIONS=list of keyword
AD or ARRAY_DIMENSIONS=keyword
ISM or INPUT_SOURCE_MAP=file
STATUS=status variable

Parameters: LO - list options
 lo meaning
 - -----
 S source list
 O object code list
 R cross-reference list
 N none
 (default: S)

AD - specify dimensioning for arrays
 ab ad meaning
 -- -----
 S STATIC fixed when compiled
 D DYNAMIC dimensions can be changed
 (default: STATIC)

Similar commands: NOS, VMS: BASIC

Examples: /basic
 =====
 /basic ad=d lo=sr l=\$local.b_lis
 ^-- list source and cross-reference
 in file \$LOCAL.B_LIS; use
 dynamic storage allocation

BLOCK Group a sequence of statements into a block.

Syntax: label: BLOCK
 statement list
 BLOCKEND label

Parameters: label - the name of the block
 (can be used in EXIT statements within
 the block)

Similar commands: VMS: DECK...EOD

Examples: blk01: BLOCK
 ...
 EXIT blk01 WHEN NOT status.normal
 ...
 BLOCKEND blk01

CANCEL Cancel the most recently selected conditions.

Syntax: CANCEL condition names

Parameters: condition names - the conditions to be cancelled
(separate multiple condition names with comma or space)
ANY_FAULT
COMMAND_EXIT
EXIT
LIMIT_FAULT
PROGRAM_FAULT
PAUSE(INTERRUPT)

Remarks: Non-selected conditions are ignored.

Examples: /cancel ce pf

CAUSE Cause a condition to occur.

Syntax: CAUSE status expression

Parameters: status expression - the condition to be caused --
must be of type STATUS
(this defines OSV\$STATUS)

Remarks: If there is a condition handler for this condition, it is activated.

CHANGE_170_REQUEST

Change the 170 (NOS) tape file description in a temporary NOS/VE file formed in a preceding CREATE_170_REQUEST command.

Syntax: CHANGE_170_REQUEST
CHAlR

F or FILE=file

FSP or FILE_SET_POSITION=keyword

FI or FILE_IDENTIFIER=string

FSN or FILE_SEQUENCE_NUMBER=integer

GN or GENERATION_NUMBER=integer

IC or INTERNAL_CODE=keyword

CC or CHARACTER_CONVERSION=boolean

BT or BLOCK_TYPE=keyword

RT or RECORD_TYPE=keyword

MBL or MAXIMUM_BLOCK_LENGTH=integer

MRL or MAXIMUM_RECORD_LENGTH=integer

TF or TAPE_FORMAT=keyword

STATUS=status variable

CHANGE_BACKUP_LABEL_TYPE

Change the job default label type for backup to tape.

Syntax: CHANGE_BACKUP_LABEL_TYPE -or-
 CHABLT
 FLT or FILE_LABEL_TYPE=keyword

 STATUS=status variable

Parameters: FLT - U or UNLABELLED
 L or LABELLED

CHANGE_CATALOG_CONTENTS

Delete damage condition notices from files in the specified catalog and all its subcatalogs, and delete catalog entries for files with no cycle data.

Syntax: CHANGE_CATALOG_CONTENTS -or-
 CHANGE_CATALOG_CONTENT or
 CHACC
 C or CATALOG=keyword or file

 DDC or DELETE_DAMAGE_CONDITION or
 DELETE_DAMAGE_CONDITIONS=list of
 keyword
 DUR or DELETE_UNRECONONCILED_FILES or
 DLC or DELETE_LOST_CYCLES=boolean
 STATUS=status variable

Parameters: C - the catalog to be changed
 ALL - your master catalog

DDC - the damage condition to be deleted
 ab option

MII MEDIA_IMAGE_INCONSISTENT
PCR PARENT_CATALOG_RESTORED
RMM RESPF_MODIFICATION_MISMATCH

DUF - delete catalog entries for all files
 residing on a missing or unavailable
 volume
 (default: FALSE)

Remarks: DDC or DUF is required.

Examples: /chacc \$user.cat1

CHANGE_CATALOG_ENTRY

Change a file's catalog entry item(s).

Syntax: CHANGE_CATALOG_ENTRY -or-
 CHACE
 F or FILE=file

 PW or PASSWORD=keyword or name
 NFN or NEW_FILE_NAME=name
 NC or NEW_CYCLE=integer
 NPW or NEW_PASSWORD=keyword or name
 NL or NEW_LOG=boolean
 NR or NEW_RETENTION=integer
 NAP or NEW_ACCOUNT_PROJECT=boolean
 DDC or DELETE_DAMAGE_CONDITION=list of
 keyword
 STATUS=status variable

Similar commands: COS: ALTACN; MODIFY
 NOS: CHANGE
 VMS: RENAME; SET PROTECTION

Examples: /chace old_name,nfn=new_name
 ^-- change name of file from
 OLD_NAME to NEW_NAME

CHANGE_COMMAND_SEARCH_MODE

Change the command list search mode.

Syntax: CHANGE_COMMAND_SEARCH_MODE -or-
 CHACSM
 SM or SEARCH_MODE=keyword

 STATUS=status variable

Parameters: SM - G or GLOBAL
 R or RESTRICTED
 E or EXCLUSIVE

Similar commands: NOS: LIBRARY

Examples: /chasm,g

CHANGE_CONNECTION_ATTRIBUTES

Change the terminal file's connection attributes.

Syntax: CHANGE_CONNECTION_ATTRIBUTES -or-
 CHANGE_CONNECTION_ATTRIBUTE -or-
 CHACA -or-
 CHANGE_TERM_CONN_ATTRIBUTE -or-
 CHANGE_TERM_CONN_ATTRIBUTES -or-
 CHATCA
 TFN or TERMINAL_FILE_NAME=file

 AC or ATTENTION_CHARACTER=string
 BKA or BREAK_KEY_ACTION=integer
 EOI or END_OF_INFORMATION=string
 IBS or INPUT_BLOCK_SIZE=integer
 IEM or INPUT_EDITING_MODE=keyword
 IOM or INPUT_OUTPUT_MODE=keyword
 IT or INPUT_TIMEOUT=boolean
 ITL or INPUT_TIMEOUT_LENGTH=integer
 ITP or INPUT_TIMEOUT_PURGE=boolean
 PCF or PARTIAL_CHARACTER_FORWARDING=boolean
 PF or PROMPT_FILE=file
 PS or PROMPT_STRING=string
 SBC or STORE_BACKSPACE_CHARACTER=boolean
 SND or STORE_NULS_DELS=boolean
 TCM or TRANSPARENT_CHARACTER_MODE=keyword
 TFC or TRANSPARENT_FORWARD_CHARACTER=
 list of string
 TLM or TRANSPARENT_LENGTH_MODE=keyword
 TML or TRANSPARENT_MESSAGE_LENGTH=integer
 TPM or TRANSPARENT_PROTOCOL_MORE=keyword
 TTC or TRANSPARENT_TERMINATE_CHARACTER=
 list of string
 TTM or TRANSPARENT_TIMEOUT_MODE=keyword
 STATUS=status variable

Parameters: TFN - the terminal file name

Similar commands: NOS: TRMDEF
 VMS: SET TERMINAL

CHANGE_FILE_ATTRIBUTES

Change an existing file's attributes.

Syntax: CHANGE_FILE_ATTRIBUTES -or-
 CHANGE_FILE_ATTRIBUTE -or-
 CHAFA
 F or FILE=file

 FAPN or FILE_ACCESS_PROCEDURE_NAME=
 FAP or FILE_ACCESS_PROCEDURE=
 keyword or name

FC or FILE_CONTENTS=
 FILE_CONTENT=keyword
 FL or FILE_LIMIT=integer
 FP or FILE_PROCESSOR=keyword or name
 FS or FILE_STRUCTURE=keyword or name
 FW or FORCED_WRITE=keyword or boolean
 LN or LINE_NUMBER=record
 LF or LOADING_FACTOR=integer
 LET or LOCK_EXPIRATION_TIME=integer
 LO or LOGGING_OPTIONS or
 LOGGING_OPTION=list of keyword
 LR or LOG_RESIDENCE=keyword or file
 RL or RECORD_LIMIT=integer
 RA or RING_ATTRIBUTES=list of integer
 SI or STATEMENT_IDENTIFIER=record
 UI or USER_INFORMATION=string
 STATUS=status variable

Similar commands: NOS: FILE

CHANGE_INPUT_ATTRIBUTE

Change an input job's attributes -- resubmit a job.

Syntax:

CHANGE_INPUT_ATTRIBUTE -or-
 CHANGE_INPUT_ATTRIBUTE -or-
 CHAIA
 N or NAME or NAMES=list of name

 CB or COMMENT_BANNER=string
 C or COPIES=integer
 CTL or CPU_TIME_LIMIT=keyword or integer
 D or DEVICE=keyword or name
 EPT or EARLIEST_PRINT_TIME=keyword or
 date_time
 ERT or EARLIEST_RUN_TIME=keyword or
 date_time
 EC or EXTERNAL_CHARACTERISTICS=keyword or
 name
 FC or FORMS_CODE=keyword or string
 JAD or JOB_ABORT_DISPOSITION=keyword
 JC or JOB_CLASS=name
 JDBO or JOB_DEFERRED_BY_OPERATOR=boolean
 JDBU or JOB_DEFERRED_BY_USER=boolean
 JQ or JOB_QUALIFIER=keyword or list of name
 JRD or JOB_RECOVERY_DISPOSITION=keyword
 LPT or LATEST_PRINT_TIME=keyword or
 date_time
 LRT or LATEST_RUN_TIME=keyword or date_time
 LA or LOGIN_ACCOUNT=keyword or name
 LP or LOGIN_PROJECT=keyword or name
 MTL or MAGNETIC_TAPE_LIMIT=keyword or integer
 MWS or MAXIMUM_WORKING_SET=keyword or integer

OF or OPERATOR_FAMILY=name
 OU or OPERATOR_USER=name
 OC or OUTPUT_CLASS=keyword
 ODBU or OUTPUT_DEFERRED_BY_USER=boolean
 ODE or OUTPUT_DESTINATION=name or string
 ODU or OUTPUT_DESTINATION_USAGE=keyword or
 name
 ODI or OUTPUT_DISPOSITION=keyword or file
 OP or OUTPUT_PRIORITY=keyword
 PD or PURGE_DELAY=keyword or time_increment
 RHD or REMOTE_HOST_DIRECTIVE=string
 RB or ROUTING_BANNER=string
 SI or SITE_INFORMATION=string
 SL or SRU_LIMIT=keyword or integer
 S or STATION=keyword or name
 UI or USER_INFORMATION=string
 UJN or USER_JOB_NAME=name
 VPD or VERTICAL_PRINT_DENSITY=keyword
 VLP or VFU_LOAD_PROCEDURE=keyword or name
 STATUS=status variable

Examples: /chaia myjob c=2 sl=200
 ^-- change copies and SRU limit and
 resubmit

CHANGE_INTERACTION_STYLE

Change from line mode to screen mode or vice versa.

Syntax: CHANGE_INTERACTION_STYLE -or-
 CHAI
 S or STYLE=keyword
 MR or MENU_ROW or MENU_ROWS=integer
 EUI or ENTEND_UTILITY_INTERACTION=boolean

 STATUS=status variable

Parameters: S - the desired mode
 LINE
 SCREEN

MR - (attribute: BY_NAME)
 the number of rows of function keys that
 full-screen applications should display
 mr meaning
 -- -----
 0 do not display function keys
 1 display one row of function keys
 2 display two rows
 (not all full-screen applications use this)
 (default: 1)

EUI - (attribute: BY_NAME)
 control whether a default screen interface
 screen is to be used with utilities which
 do not have one
 (default: FALSE)

Remarks: When you log in, interactive prompting is
 disabled. CHAIS enables it.

SCL function: \$INTERACTION_STYLE - obtain current setting

Similar commands: NOS: LINE; SCREEN

Examples: /chais mr=0

CHANGE_JOB_ATTRIBUTE

Change the current job's attributes.

Syntax: CHANGE_JOB_ATTRIBUTE -or-
 CHANGE_JOB_ATTRIBUTES -or-
 CHAJA

 CB or COMMENT_BANNER=string
 C or COPIES=integer
 CAI or CYCLE_AGING_INTERVAL=integer
 DJWT or DETACHED_JOB_WAIT_TIME=keyword or
 integer
 D or DEVICE=keyword or name
 DP or DISPATCHING_PRIORITY=keyword or
 integer
 EPT or EARLIEST_PRINT_TIME=keyword or
 date_time
 EC or EXTERNAL_CHARACTERISTICS=keyword or
 name
 FC or FORMS_CODE=keyword or string
 JAD or JOB_ABORT_DISPOSITION=keyword
 JRD or JOB_RECOVERY_DISPOSITION=keyword
 LPT or LATEST_PRINT_TIME=keyword or
 date_time
 MAXWS or MAXIMUM_WORKING_SET=keyword or
 integer
 MINWS or MINIMUM_WORKING_SET=keyword or
 integer
 OF or OPERATOR_FAMILY=name
 OU or OPERATOR_USER=name
 OC or OUTPUT_CLASS=keyword
 ODBU or OUTPUT_DEFERRED_BY_USER=boolean
 ODE or OUTPUT_DESTINATION=name or string
 ODU or OUTPUT_DESTINATION_USAGE=keyword or
 name
 ODI or OUTPUT_DISPOSITION=keyword or file
 OP or OUTPUT_PRIORITY=keyword
 PAI or PAGE_AGING_INTERVAL=integer

PD or PURGE_DELAY=keyword or time_increment
 RHD or REMOTE_HOST_DIRECTIVE=string
 RB or ROUTING_BANNER=string
 SI or SITE_INFORMATION=string
 S or STATION=keyword or name
 UI or USER_INFORMATION=string
 UJN or USER_JOB_NAME=name
 VPD or VERTICAL_PRINT_DENSITY=keyword
 VLP or VFU_LOAD_PROCEDURE=keyword or name
 STATUS=status variable

Similar commands: NOS: SETJOB

amples: /chaja "new banner" 10
 ^-- new banner and 10 copies (the
 maximum)

CHANGE_JOB_LIMIT

Change a job's resource limits.

Syntax: CHANGE_JOB_LIMIT -or-
 CHAJL
 N or NAME=keyword or name
 WL or WARNING_LIMIT or RESOURCE_LIMIT or
 RL=keyword or integer

 STATUS=status variable

Parameters: N - the resource to be changed

name	meaning
CPU_TIME	cpu time limit
SRU	system resource unit limit
TASK	system task resource limit

WL - a new value for the limit (an integer or
 UMLIMITED)

SCL function: \$JOB_LIMIT - obtain the job's resource limits

See also: DISPLAY_JOB_LIMIT

Similar commands: NOS: SETASL; SETJSL; SETTTL

Examples: /chaja cpu_time 60

CHANGE_LINK_ATTRIBUTES

Change individual link attributes for communication between dual-state partners (NOS/VE and NOS).

Syntax: CHANGE_LINK_ATTRIBUTES -or-
 CHANGE_LINK_ATTRIBUTE -or-
 CHALA
 F or FAMILY=name or string
 U or USER=name or string
 PW or PASSWORD=name or string
 C or CHARGE or ACCOUNT or A=name or string
 P or PROJECT=name or string

 STATUS=status variable

Examples: /chala pw=mymssp
 ^-- you may wish to put this into
 your PROLOG file

CHANGE_LOGIN_PASSWORD

Set or change your login/batch password.

Syntax: CHANGE_LOGIN_PASSWORD -or-
 CHALP -or-
 SET_PASSWORD -or-
 SETPW
 OPW or OLD_PASSWORD or FROM or F=name
 NPW or NEW_PASSWORD or TO or T=name

 ED or EXPIRATION_DATE=keyword or date_time
 EI or EXPIRATION_INTERVAL=keyword or integer
 STATUS=status variable

Parameters: ED ~ the date and time your password will expire

EI ~ the number of days until your password expires

Remarks: If ED and EI are omitted, your expiration interval is used.

If ED and EI are specified, ED is used, but your expiration interval is changed.

Similar commands: COS: ACCOUNT,NUPW=; CSUBMIT/NUPW (on VAXcluster)
 NOS: PASSWOR
 VMS: SET PASSWORD

Examples: /setpw old new

CHANGE_MESSAGE_LEVEL

Select the NOS/VE message display format.

Syntax: CHANGE_MESSAGE_LEVEL -or-
 CHAML -or-
 SET_MESSAGE_LEVEL -or-
 SETML
 L or LEVEL or INFORMATION_LEVEL or
 IL=keyword

 STATUS=status variable

Parameters: L - the desired format
 BRIEF - severity level and message text
 (file references are in current
 working catalog)
 FULL - BRIEF plus product identifier,
 condition code
 (file references include full path)

Similar commands: VMS: SET MESSAGE

Examples: /chaml brief

CHANGE_OUTPUT_ATTRIBUTE

Change an output file's attributes.

Syntax: CHANGE_OUTPUT_ATTRIBUTE -or-
 CHANGE_OUTPUT_ATTRIBUTES -or-
 CHAOA
 N or NAME=list of name

 CB or COMMENT_BANNER=string
 C or COPIES=integer
 D or DEVICE=keyword or name
 EPT or EARLIEST_PRINT_TIME=keyword or
 date_time
 EC or EXTERNAL_CHARACTERISTICS=keyword or
 name
 FC or FORMS_CODE=keyword or string
 LPT or LATEST_PRINT_TIME=keyword or
 date_time
 OF or OPERATOR_FAMILY=name
 OU or OPERATOR_USER=name
 OC or OUTPUT_CLASS=keyword
 ODBO or OUTPUT_DEFERRED_BY_OPERATOR=boolean
 ODBU or OUTPUT_DEFERRED_BY_USER=boolean
 ODE or OUTPUT_DESTINATION=name or string
 ODU or OUTPUT_DESTINATION_USAGE=keyword or
 name
 OP or OUTPUT_PRIORITY=keyword
 PD or PURGE_DELAY=keyword or time_increment
 RHD or REMOTE_HOST_DIRECTIVE=string

```

RD    or REPRINT_DISPOSITION=keyword
RB    or ROUTING_BANNER=string
SI    or SITE_INFORMATION=string
S     or STATION=keyword or name
UI    or USER_INFORMATION=string
VPD   or VERTICAL_PRINT_DENSITY=keyword
VLP   or VFU_LOAD_PROCEDURE=keyword or name
      STATUS=status variable

```

Examples: /chaoa myjob "new banner"

CHANGE_SCL_OPTION

Change the options provided by the SCL interpreter.

```

Syntax:  CHANGE_SCL_OPTION  -or-
         CHANGE_SCL_OPTIONS -or-
         CHASCLO            -or-
         CHASO
         -----
         LSCP or LINE_STYLE_CORRECTION_PROMPTS=keyword
         SSCP or SCREEN_STYLE_CORRECTION_PROMPTS=keyword
         NFL  or NAME_FOLDING_LEVEL=keyword
         WCPT or WILD_CARD_PATTERN_TYPE=keyword
         STATUS=status variable

```

Parameters: (attribute (all parameters): BY_NAME)

```

LSCP - controls prompting for corrections to
      parameters entered in line mode
      lscp          meaning
      -----
      LINE          prompted in line mode
      SCREEN        prompted in screen mode
      NONE          no prompting -- the command must
                   be reentered
      (initial value: LINE)

SSCP - controls prompting for corrections to
      parameters entered in screen mode
      sscp          meaning
      -----
      SCREEN        prompted in screen mode
      NONE          no prompting -- the command must
                   be reentered
      (initial value: SCREEN)

```

NFL - controls the interpretation of SCL names
(file, variable and parameter names, and
names used within the Source Code and
Object Code Utilities)

STANDARD_FOLDING - upper and lower case
letters are equivalent

FULL_FOLDING - STANDARD_FOLDING plus
lower upper

```

@      c
[      {
\      |
]      }
^      ~

```

Initial value: STANDARD_FOLDING

WCPT - the wild card pattern that can be used to
match catalog or file paths; lists of
names, program names, or strings

ab	option	meaning
B	BASIC	*, ?, and ' plus \$ALL
E	EXTENDED	B plus wild card classes (requires NFL=STANDARD_FOLDING; if NFL=FULL_FOLDING, [] around character classes are treated as {})

(initial value: EXTENDED)

SCL function: \$WILD_CAF FILES
\$SELECT_WILD_CARD_NAMES
\$SELECT_WILD_CARD_PROGRAM_NAMES
\$SELECT_WILD_CARD_STRINGS

Examples: /chaso wcpt=b
 ^-- accept only basic wildcards

CHANGE_TAPE_LABEL_ATTRIBUTES

Change the current magnetic tape label attributes.

Syntax: CHANGE_TAPE_LABEL_ATTRIBUTES -or-
CHANGE_TAPE_LABEL_ATTRIBUTE -or-
CHATLA

F or FILE=file

```

BT      or BLOCK_TYPE=keyword
CC      or CHARACTER_CONVERSION=boolean
CS      or CHARACTER_SET=keyword
CD      or CREATION_DATE=date
ED      or EXPIRATION_DATE=date
FAC     or FILE_ACCESSIBILITY_CODE=string
FI      or FILE_IDENTIFIER=string
FSN     or FILE_SEQUENCE_NUMBER=integer

```

```

FSI    or FILE_SET_IDENTIFIER=string
FSP    or FILE_SET_POSITION=keyword
GN     or GENERATION_NUMBER=integer
GVN    or GENERATION_VERSION_NUMBER=integer
MAXBL  or MAXIMUM_BLOCK_LENGTH=integer
MAXRL  or MAXIMUM_RECORD_LENGTH=integer
PC     or PADDING_CHARACTER=string
RT     or RECORD_TYPE=keyword
RL     or REWRITE_LABELS=boolean
        STATUS=status variable

```

See also: DISPLAY_TAPE_LABEL_ATTRIBUTES

Similar commands: NOS: LABEL

```

Examples: /reqmt file=$local.tapel type=mt9$6250 ..
          ../extv='tape01' recv='tape01' ring=true
          /chatla file=$local.tapel rewrite_labels=true
          /copy_file file1 $local.tapel
          /detach_file $local.tapel

```

CHANGE_TERMINAL_ATTRIBUTES

Define and change an interactive terminal's attributes.

```

Syntax:  CHANGE_TERMINAL_ATTRIBUTES  -or-
         CHANGE_TERMINAL_ATTRIBUTE   -or-
         CHATA                       -or-
         SET_TERMINAL_ATTRIBUTES     -or-
         SET_TERMINAL_ATTRIBUTE      -or-
         SETTA
         -----
         AC or ATTENTION_CHARACTER=string
         BC or BACKSPACE_CHARACTER=string
         BLC or BEGIN_LINE_CHARACTER=string
         CLC or CANCEL_LINE_CHARACTER=string
         CRD or CARRIAGE_RETURN_DELAY=integer
         CRS or CARRIAGE_RETURN_SEQUENCE=string
         CFC or CHARACTER_FLOW_CONTROL=boolean
         CS or CODE_SET=keyword or name
         CCR or CONTROL_CODE_REPLACEMENT=keyword or
                                list of record
         E or ECHOPLEX=boolean
         ELC or END_LINE_CHARACTER=string
         ELP or END_LINE_POSITIONING=keyword
         EOS or END_OUTPUT_SEQUENCE=string
         EPA or END_PAGE_ACTION=keyword
         EPC or END_PAGE_CHARACTER=string
         EPP or END_PARTIAL_POSITIONING=keyword
         FL or FOLD_LINE=boolean
         FFD or FORM_FEED_DELAY=integer
         FFS or FORM_FEED_SEQUENCE=string

```


FKC or FUNCTION_KEY_CLASS=keyword or name
 HP or HOLD_PAGE=boolean
 HPO or HOLD_PAGE_OVER=boolean
 LFD or LINE_FEED_DELAY=integer
 LFS or LINE_FEED_SEQUENCE=string
 NCC or NETWORK_COMAND_CHARACTER=string
 PL or PAGE_LENGTH=integer
 PW or PAGE_WIDTH=integer
 P or PARITY=keyword
 PSC or PAUSE_BREAK_CHARACTER=string
 SA or STATUS_ACTION=keyword
 TC or TERMINAL_CLASS=name
 TM or TERMINAL_MODEL=keyword or name
 TBC or TERMINATE_BREAK_CHARACTER=string
 STATUS=status variable

Parameters: E - controls whether input characters are echoed back to the terminal

PL - the number of lines displayed an output page
0 - continuous

PW - the number of characters displayed in a line
0 - infinite (the network does no line folding)

P - parity
EVEN, ODD, MARK, NONE, ZERO

Similar commands: NOS: TRMDEF
VMS: SET TERMINAL

Examples: /chata pl=23

CHANGE_TERM_CONN_DEFAULTS

Change a terminal's default connection attributes.

CHANGE_UNSEEN_MAIL_ACTION

Specify whether you are to receive notification of new mail while your job is running.

Syntax: CHANGE_UNSEEN_MAIL_ACTION -or-
CHAUMA

A or ACTION=keyword
STATUS=status variable

Parameters: A - how to respond to new mail

ab	option	meaning
D	DISPLAY	immediate notification
P	POST	post the message until you change to DISPLAY

(default at DTRC: DISPLAY)

Examples: /chauma p ^-- hold new mail messages while
 running a program so any such
 message won't interrupt the
 program
 /exet myprog
 /chauma d ^-- restore display mode

CHANGE_UTILITY_ATTRIBUTES

Change the attributes of a utility command (UTILITY/
UTILITYEND).

Syntax: CHANGE_UTILITY_ATTRIBUTES -or-
 CHANGE_UTILITY_ATTRIBUTE -or-
 CHUAU
 U or UTILITY=name

 ESL or ENABLE_SUBCOMMAND_LOGGING=boolean
 IIP or INTERACTIVE_INCLUDE_PROCESSOR=keyword
 or name
 LP or LINE_PROPROCESSOR=keyword or name
 OM or ONLINE_MANUAL=keyword or name
 P or PROMPT=string
 T or TABLES=file
 STATUS=status variable

Parameters: (attribute: BY_NAME (all except U))

SCL function: \$UTILITY

CHANGE_WORKING_CATALOG

Establish the default working catalog.

Syntax: CHANGE_WORKING_CATALOG -or-
 CHAWC -or-
 SET_WORKING_CATALOG -or-
 SETWC
 C or CATALOG=file

 STATUS=status variable

Parameters: C - the catalog to be the working catalog --
 one of:
 . a specific catalog
 . \$LOCAL
 . \$USER (the master catalog)

SCL function: \$WORKING_CATALOG

See also: DISPLAY_WORKING_CATALOG

Similar commands: VMS: SET DEFAULT

Examples: /chawc \$user
 /chawc \$user.mycat_1
 /disv \$working_catalog -or- disv \$wc
 :NVE.AMDS
 /

CHECK_FOR_MAIL

Check for unread mail.

Syntax: CHECK_FOR_MAIL -or-
 CHEFM

 STATUS=status variable

Remarks: CHEFM executes outside of Mail/VE.

MVV\$HAVE_MAIL is set to TRUE if you have mail.

See also: ANY_MAIL (DTRC)

Similar commands: VMS: MAIL

Examples: /CHEFM
 /if mvv\$have_mail then
 if/email
 if/ifend

COBOL Compile a COBOL source program.

Syntax: COBOL
 I or INPUT=file

 B or BINARY=file
 L or LIST=file
 LO or LIST_OPTIONS=list of keyword
 AUD or A or AUDIT=boolean
 BL or BASE_LANGUAGE=keyword
 DA or DEBUG_AIDS=list of keyword
 DD or DUMP_DATA=boolean
 E or ERROR=file
 EL or ERROR_LEVEL=keyword
 EI or EXTERNAL_INPUT or EX_INPUT=file
 FIPS or FED_INFO_PROCESSING_STANDARD=list of
 keyword
 ISM or INPUT_SOURCE_MAP=file
 LBZ or LEADING_BLANK_ZERO=boolean
 LC or LITERAL_CHARACTER=string
 OL or OPTIMIZATION_LEVEL or OPTIMIZATION or
 OPT=keyword
 RC or RUNTIME_CHECKS=list of keyword
 SD or STANDARDS_DIAGNOSTICS=list of keyword
 SP or SUBPROGRAM=boolean
 STATUS=status variable

Parameters: LO - one or more of the following:

lo	meaning
NONE	none of the options
M	data- and procedure-name map
O	object code listing (do not use unless requested by User Services)
R	cross-reference list of all data- and procedure-names referenced in program
RA	cross-reference list of all data- and procedure-names
S	source program list
SA	like S, but include lines turned off by NOLIST

(default: S)

BL - specify the variety of COBOL being used

bl	meaning
ANS74	1974 ANSI COBOL Standard
ANS85	1985 ANSI COBOL Standard
COBOL5	written for compilation by COBOL5

(default: ANS85)

DA - one or more of the following:

da	meaning
NONE	none of the options
ALL	all except SY
DS	compile source debugging lines (D in column 7)
DT	put line number, symbol, and source map loader tables into object code
OC	produce object code even if source errors
SY	syntax check - no object code
TR	flow trace

(default: NONE)

EL - T and I are the same

EI - SCU library file for COPY statements
\$NULL - do not use an SCU library
(default: do not use an SCU library)

ISM - the file with the source map generated by
 SCU EXPAND_DECK OUTPUT_SOURCE_MAP
 (default: the compiler constructs the file)

LBZ - TRUE - treat leading blanks in numeric
 fields as zeros in arithmetic and
 comparisons
 (default: blanks in numeric fields are
 an error)

LC - change the non-numeric literal delimiter
 character

lc	meaning
'''	apostrophe
'''	quotation mark

(default: ''')

OL - controls optimization

option	meaning
LOW	production quality code
DEBUG	stylized for debugging

(default: LOW)

SD - diagnose non-ANSI code (ignored if BL <>
 ANS85)

sd	meaning
NONE	do not select any option
(severity,ANSI)	assign a severity to non-ANSI errors severity is I, W, or F

(default: NONE)

SP - specify whether the source is a main or
 subprogram

sp	meaning
TRUE	subprogram
FALSE	main program

(default: FALSE)

Similar commands: NOS: COBOL5
 VMS: COBOL

Examples: /cobol i=my_cob_source b=my_cob_lgo ..
 ../lo=(s,r,m) l=my_cob_list

COLLECT_TEXT

Copy lines of text from the command list to a specified file.

Syntax: COLLECT_TEXT -or-
COLT
O or OUTPUT=file

U or UNTIL=string literal
P or PROMPT=string
SM or SUBSTITUTION_MARK=keyword or string
I or INPUT=file
STATUS=status variable

Parameters: U - the string terminating the text -- is not
copied
(default: '**')

P - the prompt if input is from the terminal
(first character is format effector
(carriage control) - use space to put
prompt on a new line)
' ' - (null string) no prompt
(default: ' ct?')

Subcommand prompt: ct?

Similar commands: NOS: COPY; .DATA; NOTE
VMS: CREATE

Examples: /colt,data
ct?Run 1 on 90/06/29
ct? 1 4 7
ct?26.3 17.9e4 23
ct?**
/

COMMAND Declare an entry in utility command table.

Syntax: COMMAND
N or NAME or NAMES=list of name

P or PROCESSOR=name
A or AVAILABILITY=keyword
AL or AUTOMATICALLY_LOG=boolean
STATUS=status variable

COMPARE_FILE

A binary comparison of two files.

Syntax: COMPARE_FILE -or-
 COMPARE_FILES -or-
 COMF
 F or FILE=file
 W or WITH=file

 EL or ERROR_LIMIT=integer
 O or OUTPUT=file
 STATUS=status variable

Similar commands: COS: COMPARE
 NOS: VERIFY
 VMS: DIFFERENCES

Examples: /comf myprog.3 myprog.2
 ^-- compare two versions of a file

COMPARE_OBJECT_LIBRARY

Compare two object libraries or two object files.

Syntax: COMPARE_OBJECT_LIBRARY -or-
 COMOL
 F or FILE=file
 W or WITH=file

 O or OUTPUT=file
 STATUS=status variable

See also: CREATE_OBJECT_LIBRARY; DISPLAY_OBJECT_LIBRARY;
 DISPLAY_OBJECT_TEXT

Similar commands: NOS: VFYLIB

Examples: /comol lib1 lib2

COMPRESS_MAIL_DATABASE

Create a new copy of your mailbox file (MVFS\$ELECTRONIC_MAIL_DATABASE) in your master catalog, with unused space removed and any data faults corrected.

Syntax: COMPRESS_MAIL_DATABASE -or-
 COMMD

 STATUS=status variable

Remarks: This command reclaims deleted space.

It also allows you access to Mail/VE after any system interrupt occurring while mail is sent or received by repairing any damage done by the interrupt.

See also: EMAIL

Similar commands: VMS: MAIL> COMPRESS

Examples: /commd

CONTINUE Exit the current WHEN statement.

Syntax: CONTINUE
 option
 WHEN boolean expression

Parameters: option - one of the following:
 NEXT
 continue with statement following the
 one that activated the condition
 handler
 RETRY
 retry with statement that activated
 the condition handler
 (default: NEXT)

when - if the boolean expression is:
 TRUE - the WHEN block is exited
 FALSE - the WHEN block is not exited -
 processing continues with the
 next statement
 (default: the block is exited)

CONVERT_UPDATE_TO_SCU

Convert an UPDATE source library to SCU format.

Syntax: CONVERT_UPDATE_TO_SCU -or-
CONUTS

OPL or OLD_PROGRAM_LIBRARY=file
R or RESULT=file
L or LIST=file
NL or NAME_LIST=file
DO or DISPLAY_OPTIONS=keyword
CS or CODE_SET=keyword
SC or SELECTION_CRITERIA=file
STATUS=status variable

Parameters: OPL - UPDATE library file
 (default: OLDPL)

- R - SCU library file
(default: SOURCE_LIBRARY in your working catalog)
- NL - the substitution file
(default: no names are replaced)
- DO - controls L output - one of:
B or BRIEF - brief listing
F or FULL - full listing including the changed lines
(default: BRIEF)
- CS - the UPDATE library file code set - one of:
ASCII64 - 64-character set
(6-bit display code)
ASCII128 - 128-character set
(8/12 ASCII code)
(default: ASCII128)
- SC - the criteria fileconvert DEFINE directives from the YANK\$\$\$ deck to selection criteria commands and write to a file
(default: no selection criteria commands are written)

Remarks: The UPDATE library must be sequential.

Examples: /conuts nl=new_names cs=ascii64 ..
../sc=oldpl_criterion

COPY_FILE

Copy a file.

Syntax: COPY_FILE -or-
COPF

I or INPUT=file
O or OUTPUT=file
STATUS=status variable

Remarks: Data is copied from the open position until end-of-information (EOI) (disk files) or a tape mark (magnetic tape).

Error conditions:

I empty - abnormal status
I at EOI - FSE\$INPUT_FILE_AT_EOI
I at double tape mark - AME\$INPUT_AFTER_EOI

For multi-file, unlabeled tape, use one COPY_FILE for each file.

Single tape marks are not copied.

The output file inherits the input file's cycle attributes (except the ring attributes) unless SET_FILE_ATTRIBUTES has been issued for the output file.

The output file's structure may differ from the input file's.

Similar commands: COS: COPYD
 NOS: COPY; COPYEI; REPLACE; SAVE
 VMS: COPY

Examples: /copy_file input=\$user.myfile output=\$local.mf
 /copf \$user.myfile
 ^-- copy file to \$OUTPUT
 /copf \$user.myfile1.\$boi \$user.myfile2.\$boi
 ^-- append MYFILE1 to MYFILE2

CREATE_170_REQUEST

Create a NOS/VE temporary file to be associated with a 170 (NOS) tape file.

Syntax: CREATE_170_REQUEST -or-
 CRE1R
 F or FILE=file

 EV or EXTERNAL_VSN=list of string
 RV or RECORDED_VSN=list of string
 FSP or FILE_SET_POSITION=keyword
 FI or FILE_IDENTIFIER=string
 FSN or FILE_SEQUENCE_NUMBER=integer
 GN or GENERATION_NUMBER=integer
 IC or INTERNAL_CODE=keyword
 CC or CHARACTER_CONVERSION=boolean
 BT or BLOCK_TYPE=keyword
 RT or RECORD_TYPE=keyword
 MBL or MAXBL or MAXIMUM_BLOCK_LENGTH=integer
 MRL or MAXIMUM_RECORD_LENGTH=integer
 LT or LABEL_TYPE=keyword
 TF or TAPE_FORMAT=keyword
 STATUS=status variable

Parameters: F - the name of the NOS/VE temporary file associated with a VAX tape file by a previous CREATE_VAX_REQUEST.

EV - external identification/sticker
 (each is 1-6 characters)
 (default: EV=RV)

RV - internal VSN (each is 1-6 characters)
(ignored for unlabeled tapes unless EV is omitted - in which case, EV=RV)

FSP - the position of the VAX tape file to be read

ab	fsp	file to be read
BOS	BEGINNING_OF_SET	first tape file
CF	CURRENT_FILE	the current file
FIP	FILE_IDENTIFIER_POSITION	file identified by the FILE_IDENTIFIER and GENERATION_NUMBER parameters
FSP	FILE_SEQUENCE_POSITION	file identified by the FILE_SEQUENCE_NUMBER parameter
NF	NEXT_FILE	the next file

(default: NF)

FI - the 1- to 17-character file label
(ignored if FSP <> FIP)

FSN - file number on a multi-file tape
(range: 1-9999)
(ignored if FSP <> FSN)

GN - the specific revision number of the FI file
(range: 1-9999)
(ignored if FSP <> FIP)
(default: 1)

IC - the character set of the data on the tape

ic	meaning
AS6	6/12-bit ASCII
AS8	8/12-bit ASCII
D63	63-character display code
D64	64-character display code

(default: D64)

CC - controls character conversion

cc	meaning
TRUE	convert from <internal code>
FALSE	do not convert

(default: FALSE)

BT - the block type of the tape file

ab	bt	blocking
I	INTERNAL	internal
CC	CHARACTER_COUNT	character count

(default: INTERNAL)

RT - the record type of the tape file

abbrev	rt	meaning
W, CW	CONTROL_WORD	control word
F, FL	FIXED_LENGTH	fixed length
S, SR	SYSTEM_RECORD	system record
Z, ZB	ZERO_BYTE	zero byte

(default: CONTROL_WORD)

MBL - maximum length in 6-bit bytes of a block on the 170 tape file

(maximum: 2,147,483,615)
(default: 5120)

MRL - maximum length in 6-bit bytes of a record on the 170 tape file

(maximum: 4,398,046,511,103)
(default: 5120)

LT - specifies whether the tape is labeled

ab	lt	meaning
L	LABELLED	same as STANDARD
S	STANDARD	tape has standard labels
U	UNLABELLED	tape is unlabelled

(default: STANDARD)

TF - the format the tape

abbrev	tf	meaning
I, NI	NOS_INTERNAL	INTERNAL, NOS default
NBI, SI	NOS_BE_INTERNAL	SCOPE internal; NOS/BE default
S	STRANGER	stranger
L, LS	LONG_STRANGER	long stranger

(default: NOS_INTERNAL)

See also: CHANGE_170_REQUEST

Similar commands: NOS: LABEL

Examples: Associate temporary file FILE_FTN with a typical Fortran formatted tape file (BT=CC, RT=ZB):

```
/create_170_request file=file_ftn ..
../external_vsn='NA9876' ..
../file_set_position=file_identifier_position ..
../file_identifier='RUN 123' ..
../internal_code=as6 ..
../character_conversion=true ..
../block_type=character_count ..
```

```

../record_type=zero_byte ..
../mbl=4000 ..
../mrl=200 ..
../label_type=labelled ..
../tape_format=nos_internal
                        ^-- standard labeled tape NA9876
                           with ASCII 6/12-bit data, file
                           "RUN 123", blocked 200 by 20

```

CREATE_CATALOG

Create a new catalog.

Syntax: CREATE_CATALOG -or-
 CREC
 C or CATALOG=file

 STATUS=status variable

Parameters: C - the new catalog to be created

Remarks: An empty catalog is created.

Only the master catalog owner can create a new catalog.

Similar commands: VMS: CREATE/DIRECTORY

Examples: /crec rm

CREATE_CATALOG_PERMIT

Grant or deny permission to use a catalog.

Syntax: CREATE_CATALOG_PERMIT -or-
 CRECP
 C or CATALOG=file

 G or GROUP=keyword
 FM or FAMILY_NAME=name
 U or USER=name
 A or ACCOUNT=name
 P or PROJECT=name
 AM or ACCESS_MODE or ACCESS_MODES=list of
 keyword
 SM or SHARE_MODES=list of keyword
 AI or APPLICATION_INFORMATION=string
 STATUS=status variable

Parameters: C - the new catalog to be created

G - controls whether the permit entry is for one user or a group of users
 PUBLIC - all users
 FAMILY - all users in family
 ACCOUNT - all users in family and account
 PROJECT - all users in family, account and project
 USER - the user in family
 USER_ACCOUNT - the user in family and account
 MEMBER - the user in family, account and project
 (default: USER)

FM - the family name
 (default: the requesting job's family)

U - the specific user
 (default: the requesting job's user name)

A - the account
 (default: the requesting job's account)

P - the project
 (default: the requesting job's project)

AM - how the user(s) may access the file

mode	meaning
READ	read access is granted
APPEND	data may be added at the end of the file
MODIFY	data may be altered
EXECUTE	object code or SCL procedures may be executed
SHORTEN	data may be deleted from the end of the file
WRITE	SHORTEN, MODIFY and APPEND
ALL	READ, APPEND, MODIFY, EXECUTE, SHORTEN, and WRITE
CONTROL	delete files and change file identifications and/or attributes
CYCLE	add new cycles and create initial cycles
NONE	access is prohibited

(default: READ and EXECUTE)

SM - how all files in the catalog must be shared
 (same as AM, except CONTROL, CYCLE, NONE)
 (default: same as AM, except APPEND, SHORTEN, MODIFY, or WRITE forces NONE;
 otherwise: READ and EXECUTE)

AI - up to 31 characters for use by application programs for additional access controls (default: a string of spaces)

Remarks: If an access control entry already exists for the user or group, then AM, SM, and AI are replaced.

See also: DISPLAY_CATALOG DO=

Examples: /crecp rm g=public am=read
 ^-- allow everyone to read catalog RM

CREATE_COMMAND_LIST_ENTRY

Add entries to the beginning of end of the command list.

Syntax: CREATE_COMMAND_LIST_ENTRY -or-
 CREATE_COMMAND_LIST_ENTRIES -or-
 CRECLE
 E or ENTRIES or ENTRY=list of: keyword or
 file

 P or PLACEMENT=keyword
 STATUS=status variable

Parameters: E - the entries to be added to the command list
 \$\$SYSTEM - the \$\$SYSTEM command library is
 added
 P - controls where the entries are added
 A or AFTER - added after the current entries
 B or BEFORE - added before the current
 entries
 (default: BEFORE)

Remarks: Cannot be used when the command list search mode is EXCLUSIVE.

If the search mode is RESTRICTED, only P=A is allowed.

Similar commands: COS, NOS: LIBRARY

Examples: /crecle my_procs
 ^-- add object library MY_PROCS
 containing my procedures

CREATE_DEFAULT_VARIABLE

Create an environment variable to be used as a default.

Syntax: CREATE_DEFAULT_VARIABLE -or-
 CREDV
 N or NAME=data_name
 D or DEFAULT=string

 S or SCOPE=keyword
 STATUS=status variable

Parameters: N - the name of the default variable

D - the value
 (when the variable name is used as a default,
 this value becomes the default value)

S - ENVIRONMENT, JOB, TASK, UTILITY
 (default: JOB)

Examples: /credv osc\$disci_display_option ..
 ../'parameter_description'
 /

CREATE_FILE

Create a new file (or cycle of a file) and attach it to the job.

Syntax: CREATE_FILE -or-
 CREF
 F or FILE=file

 LFN or LOCAL_FILE_NAME=name
 PW or PASSWORD=keyword or name
 R or RETENTION=integer
 L or LOG=boolean
 STATUS=status variable

Parameters: F - (default cycle: \$NEXT)

R - the number of days the file is to be retained
 (range: 1-999; default: 999)

L - controls whether a log of file activity is to
 be kept
 (default: FALSE)

Remarks: This command is not normally used to create a
 file. It is needed if
 . the file needs password protection, attachment
 logging, or an expiration date

- . subsequent commands require a local file name for reference
- . a local file name is needed to ensure that the file cycle is consistently referenced

See also: ATTACH FILE; DETACH FILE

Similar commands: NOS: DEFINE; SAVE
VMS: CREATE (no local file concept)

Examples: /cref \$user.data 1 pw=mypw i=true

CREATE FILE CONNECTION

Create a connection between two files so that any data access request against the subject file is passed to the target file.

```
Syntax:      CREATE_FILE_CONNECTION  -or-
              CREFC
              SF or STANDARD_FILE=file
              F  or FILE=file
              -----
              STATUS=status variable
```

Parameters: SF - one of the following:
\$ECHO, \$ERRORS, \$INPUT, \$LIST, \$OUTPUT,
\$RESPONSE, any other local file

F - the target file to be connected to the subject file.

Remarks: A subject file may be connected to more than one target file.

Input: access requests are passed to the most recently connected target file.

Output: access requests are passed to all connected files.

See also: DELETE FILE CONNECTION; DISPLAY FILE CONNECTIONS

```
Similar commands:  COS: ASSIGN
                   NOS: ASSIGN; FILE
                   VMS: ASSIGN: DEFINE
```

```
Examples:  /crefc $echo    echo_file
              ^-- echo commands to file ECHO_FILE
= = = = =
/crefc sf=$output f=my_out_file
              ^-- write batch output into file
              MY_OUT_FILE
= = = = =
```

```
crefc $local.tape5 $local.input
      ^-- program reading from Fortran unit
          5 will read from keyboard -- you
            must be in your $LOCAL catalog
```

CREATE_FILE_PERMIT

Establish or modify an access control entry for a file.

Syntax: CREATE_FILE_PERMIT -or-
 CREFP
 F or FILE=file

 G or GROUP=keyword
 FM or FAMILY_NAME=name
 U or USER=name
 A or ACCOUNT=name
 P or PROJECT=name
 AM or ACCESS_MODE or ACCESS_MODES=list of
 keyword
 SM or SHARE_MODES=list of keyword
 AI or APPLICATION_INFORMATION=string
 STATUS=status variable

Parameters: same as CREATE_CATALOG_PERMIT, except they
 apply to the specified file

Similar commands: NOS: CHANGE; PERMIT
 VMS: EDIT/ACL; SET FILE

Examples: /crefp my_file g=public am=execute
 ^-- allow anyone to execute file
 MY_FILE

CREATE_FILE_VARIABLE

Create an environment variable of type FILE.

Syntax: CREATE_FILE_VARIABLE -or-
 CREFV
 N or NAME=date_name
 F or FILE=file

 S or SCOPE=keyword
 STATUS=status variable

Parameters: N - the name of the file variable

F - the value of the variable
 (must be of type FILE)

S - ENVIRONMENT, JOB, TASK, UTILITY
 (default: JOB)

Remarks: A file variable is useful when referring to files
 or catalogs that are not in your working catalog.

Similar commands: VMS: ASSIGN; DEFINE

Examples: /create_file_variable n=john_doe ..
 ../f=.xxxx.reports
 /display_catalog john_doe
 FILE: DAILY_REPORTS
 FILE: WEEKLY_REPORTS
 FILE: MONTHLY_REPORTS

CREATE_INTERSTATE_CONNECTION

Establish a NOS batch control point on a dual-state system.

Syntax: CREATE_INTERSTATE_CONNECTION -or-
 CREIC

 PJC or PARTNER_JOB_CARD=string
 STATUS=status variable

Parameters: PJC - the job statement parameters for the NOS
 job

Remarks: After CREIC, you are prompted (FA/) until you
 enter QUIT or DELETE_INTERSTATE_CONNECTION.

 While the connection is open, you may enter
 . any NOS/VE command (except this command)
 . any NOS command (executed using EXECUTE_
 INTERSTATE_COMMAND)

Subcommand prompt: FA/

See also: DELETE_INTERSTATE_CONNECTION; EXECUTE_INTERSTATE_
 COMMAND

Examples: /creic pjc='myjob,,64.'
 FA/exeic c='attach,oldfile.'
 FA/exeic c='define,newfile.'
 FA/exeic c='copy,oldfile,newfile.'
 FA/delic

CREATE_MANUAL

Read a source file of directives and text to create an on-line manual for use by EXPLAIN or HELP.

Syntax: CREATE_MANUAL -or-
 CREM
 I or INPUT=file
 O or OUTPUT=file

 E or ERROR=file
 L or LIST=file
 LO or LIST_OPTIONS=keyword
 STATUS=status variable

Parameters: O - output bound file with the finished manual
 (default: \$LOCAL.MANUAL)

E - error message file
 (default: \$LOCAL.ERRORS (the terminal))

L - if LO=R, cross-reference listing of screen
 names and indexed subjects
 (default: \$LIST)

LO - controls the contents of the LIST file

lo	meaning
----	-----
R	produce the listing
NONE	no listing
	(default: NONE)

Examples: /crm i=\$user.ccrm_in ..
 ../o=\$user.ccrm ..
 ../e=\$user.ccrm_errors ..
 ../l=\$user.ccrm_cross_reference ..
 ../lo=r

CREATE_OBJECT_LIBRARY

Begin a CREATE_OBJECT_LIBRARY utility session.

Syntax: CREATE_OBJECT_LIBRARY -or-
 CREOL

 STATUS=status variable

See also: COMPARE_OBJECT_LIBRARY; DISPLAY_OBJECT_LIBRARY;
 DISPLAY_OBJECT_TEXT

Subcommand prompt: COL/

See also: Sections 8-3, 8-4, 8-6

Similar commands: NOS: LIBEDIT; LIBGEN
VMS: LIBRARY

Examples: See sections 8-3, 8-4, 8-6.

CREATE_PROGRAM_PROFILE

Generate a program profile.

Syntax: CREATE_PROGRAM_PROFILE -or-
CREPP
TT or TARGET_TEXT=file

F or FILES or FILE=list of file
P or PARAMETER=string
L or LIBRARIES or LIBRARY=list of file
M or MODULES or MODULE=list of program_name
SP or STARTING_PROCEDURE=program_name
PO or PROFILE_ORDER=keyword
PUC or PROGRAM_UNIT_CLASS=keyword
N or NUMBER=keyword or integer
O or OUTPUT=file
STATUS=status variable

Parameters: TT -

F - object files or libraries to be included
(must include TT file) -- all modules are
included
(default: TT file)

P - parameter list passed to the program

L - object libraries to be added to the program
library list -- only those modules needed
are included -- libraries are searched in
the order specified

M - modules to be loaded from the program
library list libraries

SP - starting entry point name
(default: last transfer point encountered
during loading)

PO - order for display of program profile output

abb	po	meaning
T	TIME	time % from greatest to least
PU	PROGRAM_UNIT	alphabetically by program name
MPU	MODULE_PROGRAM_UNIT	alphabetically by module name

(default: TIME)

PUC - controls the class of program whose statistics are displayed

puc	meaning
ALL	local and remote programs units
LOCAL	program units in TT
REMOTE	program units called by TT, but not in TT
(default: ALL)	

N - number of program unit statistics to be displayed -- the statistics are sorted according to PO and displayed until N statistics have been displayed
(default: the entire program profile)

Examples: /crepp lgo o=profile_lgo
 ^-- puts the profile for the program
 in LGO into file PROFILE_LGO

CREATE_TOPICS_FILE

Create a Topics manual from a text file using binding.

Syntax: CREATE_TOPICS_FILE -or-
 CRETf
 I or INPUT=file
 O or OUTPUT=file

 E or ERROR=file
 STATUS=status variable

Parameters: F - file with the text of the manual including directives defining the manual title, headlines, and index entries (optional)

See also: DISPLAY_TOPICS_FILE; Section 8-8

Examples: See section 8-8.

CREATE_VAX_REQUEST

Create a NOS/VE temporary file to be associated with a VAX tape file used for future references to the VAX tape file.

Syntax: CREATE_VAX_REQUEST -or-
 CREVR
 F or FILE=file

 EVSN or VSN or EXTERNAL_VSN=list of string
 RVSN or RECORDED_VSN=list of string
 FSP or FILE_SET_POSITION=keyword
 FI or FILE_IDENTIFIER=string
 FSN or FILE_SEQUENCE_NUMBER=integer
 GN or GENERATION_NUMBER=integer
 STATUS=status variable

Parameters: F - the name of the NOS/VE temporary file

VSN - the external sticker number (1-6 characters) -- if more than one, they are requested in the order specified (default: RECORDED_VSN is used)

RVSN - the VSN written in the ANSI VOL1 label (1-6 characters) -- if more than one, they are located and verified in the order specified (default: EXTERNAL_VSN is used)

(note: either VSN or RVSN must be specified)

FSP - the position of the file in a set of files on a set of tape volumes
BOS or BEGINNING_OF_SET
read the first file in the set
CF or CURRENT_FILE
the last file accessed is rewritten
FIP or FILE_IDENTIFIER_POSITION
read the tape file identified by the FI parameters or and GN
FSP or FILE_SEQUENCE_POSITION
read the tape file identified by the FSN parameter
NF or NEXT_FILE
read or write the file after the last file accessed
(default: NEXT_FILE)

FI - file label (1-17 characters)

FSN - the numeric position of the file on a multifile tape -- required if FILE_SET_POSITION is specified, ignored otherwise (default: last file accessed + 1)

GN - a specific version of the tape file named in the FI parameter (range: 1-9999)

Similar commands: NOS: LABEL

CYCLE Execute the next iteration, if any, of a repetitive command.

Syntax: CYCLE
label
WHEN boolean expression

Parameters: label - a label associated with the enclosing repetitive statement to be cycled
(default: the innermode repetitive statement is cycled)

when - if the boolean expression is:
TRUE - cycling takes place
FALSE - no cycling
(default: no cycling)

DECRYPT_FILE

Decrypt a file encrypted using ENCRYPT_FILE.

Syntax: DECRYPT_FILE -or-
DECF
I or INPUT=file
O or OUTPUT=file
DK or DES_KEY=integer or string or name

DM or DES_MODE=keyword
DIV or DES_INITIAL_VECTOR=integer or string
or name
RA or RESTORE_ATTRIBUTES=boolean
STATUS=status variable

Parameters: DK - (attribute: SECURE)
the cryptographic key
type requirement

integer each byte of the binary
representation of the integer
must have odd parity
string exactly 8 ASCII-128 characters
name a valid NOS/VE name with exactly
8 characters

DM - (attribute: SECURE)
the decryption mode to be use
ab option and meaning

ECB ELECTRONIC_CODEBOOK
use the basic Data Encryption
Standard algorithm
CBC CIPHER_BLOCK_CHAINING
use the cipher block chaining mode
of operation
CFB CIPHER_FEEDBACK
use the cipher feedback mode of
operation (feedback unit data length
is 64 bits)

OFB OUTPUT_FEEDBACK
use the output feedback mode of
operation (feedback unit data length
is 64 bits)
(default: ELECTRONIC_CODEBOOK)

DIV - (attribute: SECURE)
define a seed value for CBC, CFB, OFB
(must be same as DIV of ENCRYPT_FILE;
ignored for ECB)

RA - TRUE => NOS/VE file attributes of the
input file are restored to the
decrypted file (they must have
been preserved on encryption)
FALSE => the file attributes were not
preserved or are to be ignored
(the length of the input file
must be a multiple of 8 bytes)

See also: ENCRYPT_FILE

Similar commands: VMS: ENCRYPT (DTRC)

Examples: /decf crypt_file plain_file ABCD1234

DEFINE_PRIMARY_TASK

Designate the requesting task as the primary task for the job.

Syntax: DEFINE_PRIMARY_TASK -or-
DEFPT

STATUS=status variable

DEFINE_TERMINAL

Compile your terminal definition file.

Syntax: DEFINE_TERMINAL -or-
DEFT
I or INPUT=file

B or BINARY=file
L or LIST=file
STATUS=status variable

Parameters: B - (default: TERMINAL_DEFINITIONS)

DELETE_ALL_CYCLES

(DTRC) Delete all cycles of one or more files.

Syntax: DELETE_ALL_CYCLES -or-
 DELAC
 F or FILE=list of file

 STATUS=status variable

Parameters: F - the file or list of files to be deleted

Similar commands: COS: DELETE
 NOS: PURGE
 VMS: DELETE; PURGE

Examples: /DELETE_ALL_CYCLES F=my_file
 /DELAC (my_file_1,my_file_2)

DELETE_CATALOG

Delete a catalog and, optionally, its contents.

Syntax: DELETE_CATALOG -or-
 DELC
 C or CATALOG=file

 DO or DELETE_OPTION=keyword
 STATUS=status variable

Parameters: DO - the parts of the catalog to be deleted
 CAC or CATALOG_AND_CONTENTS
 CO or CONTENTS_ONLY
 OIE or ONLY_IF_EMPTY

Remarks: You cannot delete your master catalog.

Similar commands: VMS: DELTREE (DTRC)

Examples: /delc cat1 do=co
 ^-- delete contents of catalog CAT1

DELETE_CATALOG_PERMIT

Delete an access control entry for a catalog.

Syntax: DELETE_CATALOG_PERMIT -or-
 DELC
 C or CATALOG=file

 G or GROUP=keyword
 FM or FAMILY_NAME=name
 U or USER=name
 A or ACCOUNT=name
 P or PROJECT=name
 STATUS=status variable

Similar commands: VMS: SET PROTECTION

Examples: /delcp cat1

DELETE_COMMAND_LIST_ENTRY

Delete entries from the command list.

Syntax: DELETE_COMMAND_LIST_ENTRY -or-
DELETE_COMMAND_LIST_ENTRIES -or-
DELCLE
ENTRY=list of: keyword or file

STATUS=status variable

Parameters: E - the entries to be deleted from the command
list
\$SYSTEM - the \$SYSTEM command library is
deleted from the list

Remarks: Cannot be used when the command list search mode
is EXCLUSIVE.

If the search mode is RESTRICTED, the entry at
the beginning of the command list cannot be
deleted.

Similar commands: COS, NOS: LIBRARY

Examples: /delcle my_procs

DELETE_FILE

Delete a file or cycle of a file.

Syntax: DELETE_FILE -or-
DELELE_FILES -or-
DELF
F or FILE=list of file

PW or PASSWORD=keyword or name
STATUS=status variable

Parameters: F - (default cycle: SLOW)

PW - (attribute: SECURE)

Remarks: CONTROL permission is required.

Similar commands: COS, VMS: DELETE
NOS: PURGE

Examples: /delf (a, bc, d_e)

DELETE_FILE_CONNECTION

Delete the connection between files.

Syntax: DELETE_FILE_CONNECTION -or-
 DELFC
 SF or STANDARD_FILE=file
 F or FILE=file

 STATUS=status variable

Parameters: SF - one of the following:
 \$ECHO, \$ERRORS, \$INPUT, \$LIST, \$OUTPUT,
 \$RESPONSE, any other file

F - the target file to be disconnected from the
 subject file.

Remarks: The original connection for \$RESPONSE cannot be
 deleted.

DELFC lists each subject file and its target
files.

See also: CREATE_FILE_CONNECTION; DISPLAY_FILE_CONNECTIONS

Similar commands: NOS: FILE
 VMS: DEASSIGN

Examples: /delfc \$echo echo_file

DELETE_FILE_PERMIT

Delete an access control entry.

Syntax: DELETE_FILE_PERMIT -or-
 DELFP
 F or FILE=file

 G or GROUP=keyword
 FM or FAMILY_NAME=name
 U or USER=name
 A or ACCOUNT=name
 P or PROJECT=name
 STATUS=status variable

Similar commands: NOS: CHANGE; PERMIT
 VMS: EDIT/ACL; SET FILE

Examples: /delfp my_file public

DELETE INTERSTATE CONNECTION

Terminate the interstate connection.

```
Syntax:      DELETE_INTERSTATE_CONNECTION  -or-
              DELIC
              QUIT
              QUI
              -----
              STATUS=status variable
```

See also: CREATE_INTERSTATE_CONNECTION; EXECUTE_INTERSTATE_COMMAND

```
Examples:    /creic pjc='myjob,,64.'
             FA/exeic c='NOS command'
             ...
             FA/delic
```

DELETE VARIABLE

Delete variables accessible from the current block.

```
Syntax:      DELETE_VARIABLE    -or-
             DELETE_VARIABLES   -or-
             DELV
             N or NAME or NAMES=data_name or
             list of data_name
             -----
             STATUS=status variable
```

Parameters: N - the variable(s) to be deleted

Similar commands: VMS: DEASSIGN; DELETE/SYMBOL

```

Examples:      /delete_variables (abc,def)
               ^-- delete variables ABC and DEF
               = = = = =
PROCEDURE sample
  VAR
    global_abc: (IOB) integer = 0
    errors: boolean = FALSE
  VAREND
  ...
  IF errors THEN
    delv global_abc
    delv global_abc
  IFEND
PROCEND

               ^-- GLOBAL_ABC is deleted twice --
               if a block creates a variable
               with a scope wider than the block
               itself, an additional access is
               established in that block -- the
               first DELV deletes the proce-
               dure's access, the second deletes
               the variable from the job block

```

DESIGN_SCREEN

Enter the Screen Design Facility (SDF).

Syntax: DESIGN_SCREEN -or-
 DESIGN_SCREENs -or-
 DESS

 L or LIBRARY=file
 M or MODE=keyword
 SM or SCREEN_MODE=name
 S or SOURCE=file
 UID=application
 UPW=application
 DL or DM_LIBRARY=file
 DS or DM_SOURCE=file
 STATUS=status variable

Similar commands: NOS: screen management
 VMS: FMS

DETACH_FILE

Detach one or more files from a job.

Syntax: DETACH_FILE -or-
 DETACH_FILES -or-
 DETF
 F or FILE or FILES=list of file

 STATUS=status variable

Remarks: Files to be detached must be closed.

Standard files (e.g., \$LIST) in \$LOCAL cannot be detached.

An error causes any remaining files in the list to remain unprocessed.

See also: ATTACH_FILE; CREATE_FILE

Similar commands: COS: RELEASE
 NOS: RETURN
 VMS: no local file concept

Examples: /detach_file file=myfile
 /detf files=(myfile1,myfile2)
 =====
 /reqmt file=\$local.tapel type=mt9\$6250 ..
 ../extv='tape01' recv='tape01' ring=true
 /chatla file=\$local.tapel rewrite_labels=true
 /copy_file file1 \$local.tapel
 /detach_file \$local.tapel

DETACH_JOB

Explicitly disconnect your terminal from the current job.

Syntax: DETACH_JOB -or-
DETJ

STATUS=status variable

Remarks: (ncc)D will do this at any time ((ncc) is your network command character).

Examples: /detj -or- %D

DISPLAY_ACTIVE_TASKS

Display tasks executing within the current job.

Syntax: DISPLAY_ACTIVE_TASKS -or-
DISAT

0 or OUTPUT=file
STATUS=status variable

Remarks: (ncc)A will do this at any time ((ncc) is your network command character).

Similar commands: VMS: SHOW SYSTEM; SHOW USERS /FULL

Examples: /disat out
 ^-- put display into file OUT

DISPLAY_BACKUP_LABEL_TAPE

Display the current job default label type for a permanent file backup file on tape.

Syntax: DISPLAY_BACKUP_LABEL_TAPE -or-
DISBLT

0 or OUTPUT=file
STATUS=status variable

DISPLAY_CATALOG

Audit files and catalogs in a specific catalog or about the access control list at the catalog level.

Syntax: DISPLAY_CATALOG -or-
DISC

C or CATALOG=file
DO or DISPLAY_OPTIONS=keyword
0 or OUTPUT=file
D or DEPTH=keyword or integer
IEC or INCLUDE_EXCEPTION_CONDITIONS=keyword
STATUS=status variable

Parameters: C - (default: current working catalog)

DO - type of display:

ab	option	meaning
ID	IDENTIFIER	display file or catalog name and type (file or catalog)
I		
F	FILE	display a summary description of the files
P	PERMIT	display access control entries
	PERMITS	
C	CONTENT	display:
	CONTENTS	. disk space occupied (bytes)
		. file size (bytes)
		. damage condition (if IEC specified)

(default: IDENTIFIER)

D - amount of information to be displayed

d	meaning
1	1-line summary of the catalog
2	files and subcatalogs
3	2 + files in each subcatalog
ALL	all subcatalogs and files

(default: 2)

IEC - (attribute: BY_NAME)

display conditions which preclude attaching the cycle

iec	meaning
ALL	display conditions (requires: DO=C and (D=3 or D=ALL))
NONE	do not display conditions

(default: NONE)

Similar commands: COS: AUDIT; DS
NOS: CATLIST; ENQUIRE,F
VMS: DIRECTORY

Examples: /display_catalog \$user
 ^-- display files in master catalog
 CATALOG: mycat_1
 FILE: EPILOG
 FILE: myfile_1
 FILE: PROLOG
/disc \$local
 ^-- display local files
/disc
 ^-- display files in current working catalog


```

/disc $user do=permits
      ^-- display access control list for
          master catalog

```

DISPLAY_CATALOG_ENTRY

Audit a file, its usage, or its access control list.

Syntax: DISPLAY_CATALOG_ENTRY -or-
DISCE
 F or FILE=file

 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=keyword
 O or OUTPUT=file
 D or DEPTH=keyword or integer
 IEC or INCLUDE_EXCEPTION_CONDITIONS=keyword
 STATUS=status variable

Parameters: DO - type of display:

ab	option	meaning
L	LOG	display file usage information
P	PERMIT PERMITS	display access control entries
D	DESCRIPTOR	display descriptive information about the file and file cycles
C	CYCLES	display: . disk space occupied (bytes) . damage condition (if IEC specified)

(default: DESCRIPTOR)

D - amount of information to be displayed

d	meaning
1	1-line summary of the file
2	each cycle of the file
ALL	all cycles of the file

(default: 2)

IEC - (attribute: BY_NAME)

display conditions which preclude attaching the cycle

iec	meaning
ALL	display conditions (requires DO=C)
NONE	do not display conditions

(default: NONE)

Similar commands: COS: AUDIT
 NOS: CATLIST
 VMS: DIRECTORY

Examples: /display_catalog_entry \$user.myfile do=log
 ^-- displays date and time, family,
 user, access count, last cycle
 /disce \$user.myfile do=descriptor
 ^-- displays number of cycles,
 account, project, password,
 log selection, cycle number,
 access count, creation date/time,
 last access date/time, mast
 modification date/time,
 expiration date
 /disce \$user.myfile do=cycles
 ^-- displays size of each cycle and
 damage condition, if any, e.g.,
 myfile 0 bytes in cycle 88
 -- cycle 88 -- respf modification mismatch

DISPLAY_COMMAND_INFORMATION

Display information about a command.

Syntax: DISPLAY_COMMAND_INFORMATION -or-
 DISCI -or-
 DISPLAY_COMMAND_PARAMETER -or-
 DISPLAY_COMMAND_PARAMETERS -or-
 DISCP
 C or COMMAND=command_reference

 O or OUTPUT=file
 DO or DISPLAY_OPTIONS=list of keyword
 STATUS=status variable

Parameters: C - the name of the command

DO - the type of information to display

ab	option	meaning
AN	ALL_NAMES	all of the command's names
	ALL_NAME	including aliases
S	SOURCE	the command list entry that contains the command
BH	BRIEF_HELP	brief help message
FH	FULL_HELP	full help message (if no full, brief is displayed)
CPD	COMPACT_PARAMETER_DESCRIPTION	
	COMPACT_PARAMETER_DESCRIPTIONS	information from the parameter description table (PDT) in compact format

PD PARAMETER_DESCRIPTION
 PARAMETER_DESCRIPTIONS
 information from the
 parameter description table
 (PDT) in expanded, easy-to-
 read format

PH PARAMETER_HELP
 parameter help messages

AU ADVANCED_USAGE
 information about
 parameters with ADVANCED
 or ADVANCED_KEY attribute

HMS HELP_MODULE_SEED
 the seed name for the help
 module associated with the
 command

 ALL
 all available information
 about the command

(default: AN,BH,CPD)

Remarks: SCL system variable OSD\$DISCI_DISPLAY_OPTIONS
 has the default display options.

Similar commands: COS: CRAY> HELP; Cint> HELP (both on
 VAXcluster)
 NOS: HELPM; HELP /LIBR=VSYN:NOS (on
 VAXcluster)
 VMS: HELP

Examples: /disci delac
 Names: -----
 description
 Parameters:
 ...

DISPLAY_COMMAND_LIST

Display information about the command list.

Syntax: DISPLAY_COMMAND_LIST -or-
 DISCL

 O or OUTPUT=file
 DO or DISPLAY_OPTIONS=list of keyword
 STATUS=status variable

Parameters: DO - display option

ab	option	meaning
E	ENTRY	all entry names
SM	SEARCH_MODE	the command list search mode
	ALL	E + SM

(default: ENTRY)

```

Examples:  /discl
           :nve.culwe.ds.command_library.5
           :nve.nsys.helps_proclib.10
           :nve.nsys.proclib.60
           :nve.$system_files.,dtrc_user_library.1
           :local
           $system
           :$system.$system.application.app$command_library.2
           /

```

DISPLAY_COMMAND_LIST_ENTRY

Display information about one or more entries in a command list.

Syntax: DISPLAY_COMMAND_LIST_ENTRY -or- DISCLE

```

-----
E  or ENTRIES  or
    ENTRY=list of name or file
DO or DISPLAY_OPTIONS=list of keyword
O  or OUTPUT=file
    STATUS=status variable

```

Parameters: E - command list entry or entries
 CS or CONTROL_STATEMENT - display SCL control statement and commands
 F or FIRST - first entry in command list, allowing a display of the subcommands of an active utility
 ALL - the entire sommand list including control statements
 (default: FIRST)

DO - the contents of the output

ab	option	meaning
N	NAME	names of commands, functions, or control statements (depending on other DOs)
AN	ALL_NAMES	all names including abbreviations and aliases
C	COMMAND	individual commands
F	FUNCTION	individual functions
AU	ADVANCED_USAGE	commands and functions with ADVANCED attribute
	ALL	all other options

(default: COMMAND,NAME;
 if COMMAND and FUNCTION omitted: CCOMMAND)

```

Examples:  /discle .nsys.proclib
           <displays 2-column list of routines in PROCLIB>

```

DISPLAY_CONNECTION_ATTRIBUTES

Display the terminal file's connection attributes.

Syntax: DISPLAY_CONNECTION_ATTRIBUTES -or-
 DISPLAY_CONNECTION_ATTRIBUTE -or-
 DISCA -or-
 DISPLAY_TERM_CONN_ATTRIBUTE -or-
 DISPLAY_TERM_CONN_ATTRIBUTES -or-
 DISTA
 TFN or TERMINAL_FILE_NAME=file

 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=list of name
 O or OUTPUT=file
 STATUS=status variable

Parameters: TFN - the terminal file name

DO - see CHANGE_CONNECTION_ATTRIBUTES
 (default: ALL)

Similar commands: VMS: SHOW TERMINAL

DISPLAY_FILE

Display a file in hexadecimal and/or ASCII.

Syntax: DISPLAY_FILE -or-
 DISF
 I or INPUT=file

 O or OUTPUT=file
 F or FORMAT or FORMATS=list of keyword
 BA or BYTE_ADDRESSE or
 BYTE_ADDRESSES=list of range of integer
 STATUS=status variable

Parameters: F - the display format

f	meaning
ASCII	display in ASCII
HEX	display in hexadecimal
(default: ASCII,HEX)	

BA - address ranges to be displayed
 (default: the complete from from the open
 position)

Similar commands: NOS: TAPEDMP (DTRC); TDUMP
 VMS: DUMP

Examples: /display_file myfile f=ascii
 ^-- the file in ASCII
 /disf myfile f=(ascii,hex)
 ^-- the file in ASCII and hexadecimal
 /disf myfile ba=(3..8,16..24,56..88)
 ^-- bytes 3 thru 8, 16 thru 24, and
 56 thru 88 in ASCII and
 hexadecimal

DISPLAY_FILE_ATTRIBUTES

Display selected attributes of one or more files.

Syntax: DISPLAY_FILE_ATTRIBUTES -or-
 DISPLAY_FILE_ATTRIBUTE -or-
 DISFA
 F or FILES or FILE=list of file

 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=list of keyword
 O or OUTPUT=file
 STATUS=status variable

Parameters: DO - the attributes to display

ab	option
-----	-----
AM	ACCESS_MODE
ARL	AVERAGE_RECORD_LENGTH
BT	BLOCK_TYPE
CC	CHARACTER_CONVERSION
CTN	COLLATE_TABLE_NAME
CPN	COMPRESSION_PROCEDURE_NAME
DP	DATA_PADDING
EK	EMBEDDED_KEY
EEN	ERROR_EXIT_NAME or
EEPN	ERROR_EXIT_PROCEDURE_NAME
EL	ERROR_LIMIT
ERC	ESTIMATED_RECORD_COUNT
FAP	FILE_ACCESS_PROCEDURE or
FAPN	FILE_ACCESS_PROCEDURE_NAME
FC	FILE_CONTENTS
FLT	FILE_LABEL_TYPE
FL	FILE_LIMIT
FO	FILE_ORGANIZATION
FP	FILE_PROCESSOR
FS	FILE_STRUCTURE
FW	FORCED_WRITE
HPN	HASHING_PROCEDURE_NAME
IL	INDEX_LEVEL
IP	INDEX_PADDING
IHBC	INITIAL_HOME_BLOCK_COUNT
IC	INTERNAL_CODE
KL	KEY_LENGTH
KP	KEY_POSITION
KT	KEY_TYPE

LN	LINE_NUMBER
LET	LOCK_EXPIRATION_TIME
LO	LOGGING_OPTIONS or LOGGING_OPTION
LR	LOG_RESIDENCE
MAXBL	MAXIMUM_BLOCK_LENGTH
MAXRL	MAXIMUM_RECORD_LENGTH
MC	MESSAGE_CONTROL
MINBL	MINIMUM_BLOCK_LENGTH
MINRL	MINIMUM_RECORD_LENGTH
OP	OPEN_POSITION
PC	PADDING_CHARACTER
PF	PAGE_FORMAT
PL	PAGE_LENGTH
PW	PAGE_WIDTH
PV	PRESET_VALUE
PL	RECORD_LIMIT
RT	RECORD_TYPE
RPB	RECORDS_PER_BLOCK
SI	STATEMENT_IDENTIFIER
UI	USER_INFORMATION
AI	APPLICATION_INFORMATION
S	SIZE (file length in bytes)
GAM	GLOBAL_ACCESS_MODE
GFA	GLOBAL_FILE_ADDRESS (returns READ, WRITE, APPEND, MODIFY, SHORTEN)
GFN	GLOBAL_FILE_NAME
CFP	GLOBAL_FILE_POSITION (returns BOI, BOP, MID_RECORD, EOR, EOP, EOI)
GSM	GLOBAL_SHARE_MODE (returns NONE, READ, WRITE, APPEND, MODIFY, SHORTEN, EXECUTE)
P	PERMANENT (true or false)
RA	RING_ATTRIBUTES

(default: FC, FP, FS, GAM, P, S)

See also: SET_FILE_ATTRIBUTES

Similar commands: COS: AUDIT
NOS: CATLIST
VMS: DIRECTORY/FULL

Examples: /disfa my_file do=(am,s,p)

DISPLAY_FILE_CONNECTIONS

Display the names of the files currently connected to the subject files.

Syntax: DISPLAY_FILE_CONNECTIONS -or-
DISPLAY_FILE_CONNECTION -or-
DISFC
--
SF or STANDARD_FILES=keyword or list of file
O or OUTPUT=file
STATUS=status variable

See also: CREATE_FILE_CONNECTION; DELETE_FILE_CONNECTION

Examples: /disfc

DISPLAY_FUNCTION_INFORMATION

Display information about a function.

Syntax: DISPLAY_FUNCTION_INFORMATION -or-
DISFI
F or FUNCTION=data_name

O or OUTPUT=file
DO or DISPLAY_OPTION or
DISPLAY_OPTIONS=list of keyword
STATUS=status variable

Parameters: DO - type of information to display
(same as DISPLAY_COMMAND_INFORMATION)

Remarks: System variable OSD\$DISFI_DISPLAY_OPTIONS has the
a fault display options.

Examples: /disfi my_func

DISPLAY_INPUT_ATTRIBUTE

Display the input attributes of a queued or processing job.

Syntax: DISPLAY_INPUT_ATTRIBUTE -or-
DISPLAY_INPUT_ATTRIBUTES -or-
DISIA
N or NAME or
NAMES=list of name

DO or DISPLAY_OPTION or
DISPLAY_OPTIONS=list of keyword
O or OUTPUT=file
STATUS=status variable

Parameters: N - the names of the jobs whose attributes are
to be displayed

DO - the information to display

ab option

-----	-----
	ALL
CB	COMMENT_BANNER
CF	CONTROL_FAMILY
CU	CONTROL_USER
C	COPIES
CTL	CPU_TIME_LIMIT
DM	DATA_MODE
D	DEVICE
EPT	EARLIEST_PRINT_TIME
ERT	EARLIEST_RUN_TIME

EC	EXTERNAL_CHARACTERISTICS
FC	FORMS_CODE
JAD	JOB_ABORT_DISPOSITION
JC	JOB_CLASS
JDBO	JOB_DEFERRED_BY_OPERATOR
JDBU	JOB_DEFERRED_BY_USER
JD	JOB_DESTINATION
JDU	JOB_DESTINATION_USAGE
JER	JOB_EXECUTION_RING
JM	JOB_MODE
JQ	JOB_QUALIFIER or JOB_QUALIFIERS
JRD	JOB_RECOVERY_DISPOSITION
JS	JOB_SIZE
JST	JOB_SUBMISSION_TIME or QT or QUEUED_TIME
LPT	LATEST_PRINT_TIME
LRT	LATEST_RUN_TIME
LA	LOGIN_ACCOUNT
LF	LOGIN_FAMILY
LP	LOGIN_PROJECT
LU	LOGIN_USER
MTL	MAGNETIC_TAPE_LIMIT
MAXWS	MAXIMUM_WORKING_SET
OF	OPERATOR_FAMILY or DF or DESTINATION_FAMILY
OU	OPERATOR_USER or SO or STATION_OPERATOR
OAN	ORIGINATION_APPLICATION_NAME
OC	OUTPUT_CLASS
ODBU	OUTPUT_DEFERRED_BY_USER
ODE	OUTPUT_DESTINATION
ODU	OUTPUT_DESTINATION_USAGE or DU
ODI	OUTPUT_DISPOSITION
OP	OUTPUT_PRIORITY
PD	PURGE_DELAY
RHD	REMOTE_HOST_DIRECTIVE
RB	ROUTING_BANNER
SI	SITE_INFORMATION
SL	SRU_LIMIT
S	STATION
SJN	SYSTEM_JOB_NAME
UI	USER_INFORMATION
UJN	USER_JOB_NAME
VPD	VERTICAL_PRINT_DENSITY
VLP	VFU_LOAD_PROCEDURE

(default: ALL)

SCL function: \$JOB_INPUT - determine the input attributes of
a job

Examples: /disia my_job (js,sjn)

DISPLAY_JOB_ATTRIBUTE

Display the attributes of your current job.

Syntax: DISPLAY_JOB_ATTRIBUTE -or-
 DISPLAY_JOB_ATTRIBUTES -or-
 DISJA

 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=list of keyword
 O or OUTPUT=file
 STATUS=status variable

Parameters: DO - the information to display

ab	option
-----	-----
	ALL
CB	COMMENT_BANNER
CF	CONTROL_FAMILY
CU	CONTROL_USER
C	COPIES
CAI	CYCLIC_AGING_INTERVAL
DJWT	DETACHED_JOB_WAIT_TIME
D	DEVICE
DP	DISPATCHING_PRIORITY
EPT	EARLIEST_PRINT_TIME
ERT	EARLIEST_RUN_TIME
EC	EXTERNAL_CHARACTERISTICS
FC	FORMS_CODE
JAD	JOB_ABORT_DISPOSITION
JC	JOB_CLASS
JM	JOB_MODE
JQ	JOB_QUALIFIER or JOB_QUALIFIERS
JRD	JOB_RECOVERY_DISPOSITION
JS	JOB_SIZE
JST	JOB_SUBMISSION_TIME or QT or QUEUED_TIME
LPT	LATEST_PRINT_TIME
LRT	LATEST_RUN_TIME
LA	LOGIN_ACCOUNT
LF	LOGIN_FAMILY
LP	LOGIN_PROJECT
LU	LOGIN_USER
MAXWS	MAXIMUM_WORKING_SET
MINWS	MINIMUM_WORKING_SET
OF	OPERATOR_FAMILY or DF or DESTINATION_FAMILY
OU	OPERATOR_USER or SO or STATION_OPERATOR
OAN	ORIGINATION_APPLICATION_NAME
OC	OUTPUT_CLASS
ODBU	OUTPUT_DEFERRED_BY_USER
ODE	OUTPUT_DESTINATION
ODU	OUTPUT_DESTINATION_USAGE or DU
ODI	OUTPUT_DISPOSITION
OP	OUTPUT_PRIORITY

PAI PAGE_AGING_INTERVAL
 PD PURGE_DELAY
 RHD REMOTE_HOST_DIRECTIVE
 RB ROUTING_BANNER
 SS SENSE_SWITCHES
 SC SERVICE_CLASS
 SI SITE_INFORMATION
 S STATION
 SJN SYSTEM_JOB_NAME
 UI USER_INFORMATION
 UJN USER_JOB_NAME
 VPD VERTICAL_PRINT_DENSITY
 VLP VFU_LOAD_PROCEDURE
 (default: ALL)

SCL function: \$JOB - determine the input attributes of your
 job

Examples: /disja

DISPLAY_JOB_ATTRIBUTE_DEFAULTS

Display the current defaults for the job attributes.

Syntax: DISPLAY_JOB_ATTRIBUTE_DEFAULT -or-
 DISPLAY_JOB_ATTRIBUTE_DEFAULTS -or-
 DISJAD

 JM or JOB_MODE=keyword or list of keyword
 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=list of keyword
 O or OUTPUT=file
 STATUS=status variable

Parameters: JM - job modes for the display

jm	meaning
ALL	batch and interactive
BATCH	batch only
INTERACTIVE	interactive only

DO - job attribute defaults to be displayed

ab	attribute
ALL	
CTL	CPU_TIME_LIMIT
JAD	JOB_ABORT_DISPOSITION
JC	JOB_CLASS
JBDO	JOB_DEFERRED_BY_OPERATOR
JM	JOB_MODE
JQ	JOB_QUALIFIER or JOB_QUALIFIERS
LF	LOGIN_FAMILY
MTL	MAGNETIC_TAPE_LIMIT
MAXWS	MAXIMUM_WORKING_SET

OC OUTPUT_CLASS
 ODBU OUTPUT_DEFERRED_BY_USER
 ODU OUTPUT_DESTINATION_USAGE or DU
 PD PURGE_DELAY
 SI SITE_INFORMATION
 SL SRU_LIMIT
 S STATION
 VPD VERTICAL_PRINT_DENSITY
 VLP VFU_LOAD_PROCEDURE
 (default: ALL)

SCL function: \$JOB_DEFAULT

See also: CHANGE_JOB_ATTRIBUTE

Examples: /disjad

DISPLAY_JOB_LIMITS

Display your job's limits.

Syntax: DISPLAY_JOB_LIMIT -or-
 DISPLAY_JOB_LIMITS -or-
 DISJL

 O or OUTPUT=file
 STATUS=status variable

Remarks: DISJL displays the count, warning and maximum
 limits for CP_TIME, MAGNETIC_TAPE, PERMANENT_
 FILE_SPACE, SRU, TASK, TEMPORARY_FILE_SPACE.

SCL function: \$JOB_LIMIT

Examples: /disjl

DISPLAY_JOB_STATUS

Display current status of queued or executing jobs.

Syntax: DISPLAY_JOB_STATUS -or-
 DISJS -or-
 DISPLAY_INPUT_STATUS -or-
 DISIS

 N or NAME or NAMES or JN or JOB_NAME or
 JOB_NAMES=keyword or list of name
 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=list of keyword
 O or OUTPUT=file
 STATUS=status variable

Parameters: N - one or more jobs whose status is to be displayed

DO - the information to be displayed

ab	option
----	-----
	ALL
CF	CONTROL_FAMILY
CU	CONTROL_USER
CTU	CPU_TIME_USED
DM	DISPLAY_MESSAGE
JC	JOB_CLASS
JCP	JOB_CLASS_POSITION
JDU	JOB_DESTINATION_USAGE
JIT	JOB_INITIATION_TIME
JM	JOB_MODE
JS	JOB_STATE (returns DEFERRED, QUEUED, INITIATED, TERMINATED)
LF	LOGIN_FAMILY
LU	LOGIN_USER
OAP	OPERATOR_ACTION_POSTED
PF	PAGE_FAULTS
SJN	SYSTEM_JOB_NAME
UJN	USER_JOB_NAME

(default: the value of OSD\$DISJS_DISPLAY_OPTIONS; if not defined: CTU, DM, JC, JF, SJN)

Remarks: ((ncc)S will do this at any time ((ncc) is your network command character).

((ncc)J will do this for all your jobs at any time.

System variable OSD\$DISJS_DISPLAY_OPTIONS has the default display options.

SCL function: \$JOB_STATUS

Similar commands: NOS: ENQUIRE, JSN
VMS: SHOW PROCESS; SHOW QUEUE; SHOW SYSTEM

Examples: /disjs

DISPLAY_LINK_ATTRIBUTES

Display individual link attribute values.

Syntax: DISPLAY_LINK_ATTRIBUTES -or-
DISLA

DO or DISPLAY_OPTION or
DISPLAY_OPTIONS=list of keyword
O or OUTPUT=file
STATUS=status variable

Parameters: DO - display options

ab	option	meaning
	ALL	all attributes
C	CHARGE	NOS charge number (not used at DTRC)
F	FAMILY	NOS family name
P	PROJECT	NOS project number (not used at DTRC)
U	USER	NOS username

Examples: /disla
CHARGE :
FAMILY : nlfam
PROJECT :
USER : AMDS
/

DISPLAY_LOG

Display the job log for the requesting job.

Syntax: DISPLAY_LOG -or-
DISL

DO or DISPLAY_OPTION or
DISPLAY_OPTIONS=keyword or integer
O or OUTPUT=file
STATUS=status variable

Parameters: DO - the portion of the log to be displayed

ab	option	meaning
A	ALL	start at the beginning of the log
L	LAST	start with the first message after the last DISL
	integer	the last n messages of the log and the current message

Remarks: The display includes:

- . time stamp
- . message origin
 - CI - command interpreted
 - CS - command skipped
 - PR - program generated
 - RC - job recovery
 - SY - system generated
- . message text

((ncc)L will do this at any time ((ncc) is your network command character).

Similar commands: NOS: DAYFILE
VMS: SET VERIFY; TYPE job.LOG

Examples: /disl 1

DISPLAY MESSAGE

Display a message in one or more logs.

```
Syntax:      DISPLAY_MESSAGE  -or-
             DISM
             M  or MESSAGE=string
             -----
             T  or TO=list of keyword
                  STATUS=status variable
```

```
Parameters: T - the message destination
```

option	how to view
JOB_MESSAGE	DISPLAY_JOB_STATUS
JOB	DISPLAY_LOG
JOB_STATISTIC	
ALL	
(default: JOB)	

Similar commands: NOS: COMMENT; DISPLAY

Examples: /dism "Running my prog"

DISPLAY OBJECT LIBRARY

Display information about an object library, object file, or procedure file.

```
Syntax:      DISPLAY_OBJECT_LIBRARY  -or-
             DISOL
             L  or LIBRARY=file
             -----
             M  or MODULE or
                MODULES=list of range of any
             DO or DISPLAY_OPTION or
                DISPLAY_OPTIONS=list of keyword
             O  or OUTPUT=file
             AO or ALPHABETICAL_ORDER=boclean
                STATUS=status variable
```

Parameters: L - the file to be displayed

M - the list of modules to display
(use a string variable for non-SCL name)

```

DO - display options
    ab  option
    ----
        NONE (type and name)
    DT  DATE_TIME
    EP  ENTRY_POINT
    H    HEADER
    L    LIBRARY or LIBRARIES
    R    REFERENCE
    C    COMPONENT
        ALL
    (default: DT)

```

```

AO - TRUE  => alphabetical listing
    FALSE => list in order the modules exist
              in the file
    (default: FALSE)

```

Remarks: Although modules may be mixed in a library, you may wish to put programs, subprograms, procedures, and procedure helps (or procedures and their helps) into separate libraries.

See also: COMPARE_OBJECT_LIBRARY; CREATE_OBJECT_LIBRARY;
DISPLAY_OBJECT_TEXT

Similar commands: COS, NOS: ITEMIZE
VMS: LIBRARY

Examples: /disol anylib
Display of object library - anylib 09:48:23 05/09/90 PAGE 1

LEN_TRIM	- load module	- 13:13:28 1990-02-28
GETAC	- load module	- 13:05:25 1990-03-02
GETUJN	- load module	- 13:05:25 1990-03-02
FICHE	- command procedure	- 12:05:18 1990-04-05
XEROX	- command procedure	- 12:06:24 1990-04-05
\$UNREAD_MAIL	- function procedure	- 14:23:57 1990-05-03
HELP_FICHE\$US_ENGLISH	- message module	- 11:08:37 1990-04-05
HELP_XEROX\$US_ENGLISH	- message module	- 11:08:40 1990-04-05

DISPLAY_OBJECT_TEXT

Display object records of an object library or file.

```

Syntax:  DISPLAY_OBJECT_TEXT -or-
         DISOT
         -----
         F    or FILE=file
         O    or OUTPUT=file
         DHR or DISPLAY_HEX_RECORDS=boolean
              STATUS=status variable

```


Parameters: F - (default: LGO)

DHR - controls whether large hexadecimal fields
(such as a code section) should be
displayed
(default: TRUE)

See also: COMPARE_OBJECT_LIBRARY; CREATE_OBJECT_LIBRARY;
DISPLAY_OBJECT_LIBRARY

Examples: /disot

DISPLAY_OUTPUT_ATTRIBUTE

Display attributes of an output file.

Syntax: DISPLAY_OUTPUT_ATTRIBUTE -or-
DISPLAY_OUTPUT_ATTRIBUTES -or-
DISOA
N or NAMES or NAME=file

DO or DISPLAY_OPTION or
DISPLAY_OPTIONS=keyword or
list of keyword
O or OUTPUT=file
STATUS=status variable

Parameters: N - (system- or user-supplied name of the file
to be displayed

DO - display options

ab	option
----	-----
	ALL
CB	COMMENT_BANNER
CF	CONTROL_FAMILY
CU	CONTROL_USER
C	COPIES
CP	COPIES_PRINTED
DM	DATA_MODE
D	DEVICE
DT	DEVICE_TYPE
EPT	EARLIEST_PRINT_TIME
EC	EXTERNAL_CHARACTERISTICS
FP	FILE_POSITION
FS	FILE_SIZE
FC	FORMS_CODE
LPT	LATEST_PRINT_TIME
LA	LOGIN_ACCOUNT
LF	LOGIN_FAMILY
LP	LOGIN_PROJECT
LU	LOGIN_USER

OF	OPERATOR_FAMILY or DF or DESTINATION_FAMILY
OU	OPERATOR_USER or SO or STATION_OPERATOR
OAN	ORIGINATING_APPLICATION_NAME
OC	OUTPUT_CLASS
ODBO	OUTPUT_DEFERRED_BY_OPERATOR
ODBU	OUTPUT_DEFERRED_BY_USER
ODE	OUTPUT_DESTINATION
ODU	OUTPUT_DESTINATION_USAGE or DU or DESTINATION_USAGE
OP	OUTPUT_PRIORITY
OST	OUTPUT_SUBMISSION_TIME or QT or QUEUED_TIME
PD	PURGE_DELAY
RHD	REMOTE_HOST_DIRECTIVE or DSRP or DUAL_STATE_ROUTE_PARAMETERS
RB	ROUTING_BANNER
SI	SITE_INFORMATION
S	STATION
SJN	STATION_JOB_NAME
UFN	USER_FILE_NAME
UI	USER_INFORMATION
UJN	USER_JOB_NAME
VPD	VERTICAL_PRINT_DENSITY
VLP	VFU_LOAD_PROCEDURE

SCL function: \$JOB OUTPUT

Examples: /disoa my job sijn

DISPLAY OUTPUT HISTORY

Display job log entries for output files.

Syntax: DISPLAY_OUTPUT_HISTORY -or-
 DISOH

```

OFN or OUTPUT_FILE_NAME=list of: keyword or
                                name
JN  or JOB_NAME=list of: keyword or name
LF  or LOGIN_FAMILY=list of: keyword or name
BLP or BEGINNING_LOG_POSITION=keyword
SO  or SORTED_ORDER=keyword
O   or OUTPUT=file
    STATUS=status variable

```

Examples: /disoh

DISPLAY_OUTPUT_STATUS

Display the status of a file in the output queue.

Syntax: DISPLAY_OUTPUT_STATUS -or-
 DISOS -or-
 DISPLAY_PRINT_STATUS -or-
 DISPS

 N or NAME=keyword or list of name
 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=keyword or list of
 keyword
 O or OUTPUT=file
 STATUS=status variable

Parameters: N - the files to be displayed

DO - display options

ab	option
----	-----
	ALL
CB	COMMENT_BANNER
CF	CONTROL_FAMILY
CU	CONTROL_USER
LF	LOGIN_FAMILY
LU	LOGIN_USER
ODU	OUTPUT_DESTINATION_USAGE
OP	OUTPUT_PRIORITY
OS	OUTPUT_STATE
	(returns DEFERRED, QUEUED, INITIATED, TERMINATING, COMPLETED)
SFN	SYSTEM_FILE_NAME
SJN	SYSTEM_JOB_NAME
UFN	USER_FILE_NAME
	(default: OS, SFN, UFN)

SCL function: \$OUTPUT_STATUS

Similar commands: COS: CRAY> STATUS /O (from VAXcluster)
 NOS: ENQUIRE, JSN; Q
 VMS: SHOW QUEUE

Examples: /disos

DISPLAY_PROGRAM_ATTRIBUTES

Display the current job library list and default execution option values.

Syntax: DISPLAY_PROGRAM_ATTRIBUTES -or-
 DISPLAY_PROGRAM_ATTRIBUTE -or-
 DISPA

 O or OUTPUT=file
 STATUS=status variable

See also: SET_PROGRAM_ATTRIBUTES

Examples: /dispa

DISPLAY_SCL_OPTIONS

Display the options set by CHANGE_SCL_OPTIONS.

Syntax: DISPLAY_SCL_OPTIONS -or-
 DISPLAY_SCL_OPTION -or-
 DISCLO -or-
 DISSO

 0 or OUTPUT=file
 STATUS=status variable

SCL function: \$SCL_OPTIONS - the values set by
 CHANGE_SCL_OPTIONS

Examples: /disso
 Line_Style_Correction_Prompts : line
 Screen_Style_Correction_Prompts : screen
 Name_Folding_Level : standard_folding
 Wild_Card_Pattern_Type : extended

DISPLAY_TAPE_LABEL_ATTRIBUTES

Display the current magnetic tape label attributes.

Syntax: DISPLAY_TAPE_LABEL_ATTRIBUTES -or-
 DISPLAY_TAPE_LABEL_ATTRIBUTE -or-
 DISTLA
 F or FILE=file

 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=list of name
 0 or OUTPUT=file
 STATUS=status variable

Parameters: DO - display options

ab	option
BT	BLOCK_TYPE
BO	BUFFER_OFFSET
CC	CHARACTER_CONVERSION
CS	CHARACTER_SET
CD	CREATION_DATE
ED	EXPIRATION_DATE
FAC	FILE_ACCESSIBILITY_CODE
FI	FILE_IDENTIFIER
FSN	FILE_SEQUENCE_NUMBER
FSI	FILE_SET_IDENTIFIER
FSP	FILE_SET_POSITION

GN	GENERATION_NUMBER
GVN	GENERATION_VERSION_NUMBER
MAXBL	MAXIMUM_BLOCK_LENGTH
MAXRL	MAXIMUM_RECORD_LENGTH
NF	NEXT_FILE
PC	PADDING_CHARACTER
RT	RECORD_TYPE
RL	REWRITE_LABELS
S	SIZE (file length in bytes)
	ALL

(default: ALL, NF)

See also: CHANGE_TAPE_LABEL_ATTRIBUTES

Examples: /distla na9876

DISPLAY_TASK_STATUS

Display the status of one or more tasks.

Syntax: DISPLAY_TASK_STATUS -or-
DISTS
TN or TASK_NAME or TASK_NAMES=list of:
keyword or name

O or OUTPUT=file
STATUS=status variable

Parameters: TN - the name of the task(s) whose status is
to be displayed
ALL - all tasks

SCL function: \$TASK_STATUS

Examples: /dists all

DISPLAY_TERMINAL_ATTRIBUTES

Display interactive terminal attribute values.

Syntax: DISPLAY_TERMINAL_ATTRIBUTES -or-
DISPLAY_TERMINAL_ATTRIBUTE -or-
DISTA

DO or DISPLAY_OPTION or
DISPLAY_OPTIONS=list of name
O or OUTPUT=file
STATUS=status variable

Parameters: DO - see CHANGE_TERMINAL_ATTRIBUTES

Similar commands: VMS: SHOW TERMINAL

Examples: /dista (pl,pw)

DISPLAY_TERM_CONN_DEFAULTS

Display the defaults for terminal connection attributes.

Syntax: DISPLAY_TERM_CONN_DEFAULTS -or-
 DISPLAY_TERM_CONN_DEFAULT -or-
 DISTCD

 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=list of name
 O or OUTPUT=file
 STATUS=status variable

Parameters: DO - the default attribute to be displayed
 (see CHANGE_TERM_CONN_DEFAULTS (CHATCD))
 (default: ALL)

Examples: /distcd

DISPLAY_TOPICS_FILE

Open a Topics online manual.

Syntax: DISPLAY_TOPICS_FILE -or-
 DISTF
 I or INPUT=file

 IE or INDEX_ENTRY=string
 ED or EXPAND_DEPTH=integer
 L or LIST=file
 IM or INDEX_MATCH=keyword
 STATUS=status variable

See also: CREATE_TOPICS_FILE; EXPLAIN; HELP

Examples: See Section 8-8.

DISPLAY_VALUE

Display the value of an expression.

Syntax: DISPLAY_VALUE -or-
 DISPLAY_VALUES -or-
 DISV
 V or VALUES or VALUE=any

 O or OUTPUT=file
 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=list of keyword
 STATUS=status variable

Parameters: V - the expression to be evaluated and displayed

DO - display options

ab	option
CLE	COMPRESSED_LABELED_ELEMENTS
DS	DATA_STRUCTURE
DE	DISPLAY_ELEMENTS
E	ELEMENTS or ELEMENT
LE	LABELED_ELEMENTS
S	SOURCE

SCL function: \$STRING

See also: PUT_LINE

Similar commands: NOS: DISPLAY
VMS: WRITE

Examples: /v = 154
/disv v
154
/

DISPLAY_VARIABLE_LIST

Display variables accessible to the current block.

Syntax: DISPLAY_VARIABLE_LIST -or-
DISVL

O or OUTPUT=file
STATUS=status variable

SCL function: \$VARIABLE

Similar commands: VMS: SHOW SYMBOL

Examples: /disvl
VARIABLES IN JOB

mvv\$have_mail	fdf\$library
c2f\$libmalloc	c2f\$run_time_library
c2f\$libm	c2f\$libc
c2f\$library	adf\$library
ftf\$library	tuf\$terminal_definitions
tuf\$library	osf\$system_library
paf\$library	bcf\$library
aaf\$44d_library	aaf\$4dd_library
smf\$library	mlf\$library

dbf\$library	cbf\$4dd_library
flf\$library	cyf\$run_time_library
osv\$status	

/

DISPLAY_WORKING_CATALOG

Display your working catalog.

Syntax: DISPLAY_WORKING_CATALOG -or-
 DISWC

 0 or OUTPUT=file
 STATUS=status variable

Similar commands: VMS: SHOW DEFAULT

Examples: /diswc
 :NVE.AMDS
 /

ECHO (DTRC) Control echoing of SCL statements to a file.

Syntax: ECHO
 0 or OPT or OPTION=keyword
 F or FILE=file

 STATUS=status variable

Parameters: 0 - one of:
 OFF - turn echoing off
 ON - turn echoing on

 F - the echo file

Remarks: Before an echo file can be examined, echoing to it
 must be off.

Similar commands: NOS: DAYFILE
 VMS: SET VERIFY

Examples: /ECHO ON E
 <do something>
 /ECHO OFF E
 /COPF E <-- echo to file E and then examine
 it

EDIT_CATALOG

Begin an EDIT_CATALOG utility session to edit, view, copy, move, print your files.

Syntax: EDIT_CATALOG -or-
EDIC

```

-----
C   or CATALOG=file
DO  or DISPLAY_OPTION or
      DISPLAY_OPTIONS=keyword
KDC or KEEP_DISPLAY_CURRENT=boolean
NDF or NO_DOLLAR_FILES
      STATUS=status variable

```

Parameters: C - the catalog to be displayed/edited.
(default: the current working catalog)

DO - display options

ab	option	meaning
A	ALL	display all file attributes
B	BRIEF	display the name and entry type (file or catalog)

(default: BRIEF)

KDC - TRUE => update display after each operation
FALSE => update only after home line
commands and some other operations
(default: FALSE)

NDF - TRUE => \$ files are not displayed
(synonyms: ON, YES)
FALSE => \$ files are displayed
(synonyms: OFF, NO)
(default: FALSE)

See also: ENTER_FILE_MANAGER

Examples: /edic do=a
<list of files with keypad options to manipulate them>

EDIT_DECK

(SCU subcommand) Open a deck in the working library for editing while maintaining your current position in other decks.

Syntax: EDIT_DECK -or-
 EDID -or-
 EDIT_LIBRARY -or-
 EDIL

 D or DECK=keyword or name
 M or MODIFICATION=name
 I or INPUT=file
 O or OUTPUT=file
 P or PROLOG=file
 DUC or DISPLAY_UNPRINTABLE_CHARACTERS=boolean
 STATUS=status variable

Parameters: D - the deck to be edited (required if you have never entered a deck on a DECK parameter); if it does not exist, it is created
 NONE - enter editing without a deck name (default: the last deck used)

Remarks: If you unintentionally create a deck, you can discard it with:
 end_deck write_deck=false -or- endd wd=false

EDIT_FILE

Begin a file editor (EDIT_FILE utility) session.

Syntax: EDIT_FILE -or-
 EDIF
 F or FILE=file

 I or INPUT=file
 O or OUTPUT=file
 P or PROLOG=file
 DUC or DISPLAY_UNPRINTABLE_CHARACTERS=boolean
 STATUS=status variable

Parameters: I - subcommands for editing the file (default: \$COMMAND)

 P - editing subcommands to be executed each time you start the editor (default: \$USER.SCU_EDITOR_PROLOG)

 DUC - TRUE => unprintable characters (ASCII 127 and 0-31) are displayed as <name> and written as the ASCII characters

FALSE => unprintable characters are
displayed as a space with a
warning message AND WRITTEN as a
space
(default: FALSE)

Remarks: To use an epilog file of subcommands, use
SET_EPILOG. You can also place this in your
prolog file.

To edit a second file while in the editor, use
EDIT_FILE.

Subcommand prompt: ef/

Similar commands: NOS: FSE
VMS: EDIT/TPU (EVE); EDT

Examples: /edif my_file

EMAIL Begin a Mail/VE session.

Syntax: EMAIL -or-
EMA

OP or OPERATION=keyword
O or OUTPUT=file
E or ERROR=file
P or PROLOG=file
STATUS=status variable

Parameters: OP - the operation to be performed

ab	op	meaning
C	CHECK	check your mailboxes for mail
R	READ	read your mail
W	WRITE	compose and send a letter

P - may contain Mail/VE subcommands or SCL
commands
(SCL default name variable MVD\$MAIL_PROLOG
may be created and set to the name of your
prolog file (as a string))

Remarks: EMAIL operates in line mode or screen mode
depending on the interactive style.

Subcommand prompt: Mail/

Similar commands: VMS: MAIL

Examples: /email <-- read your mail
/email write
^-- send mail

ENCRYPT_FILE

Encrypt a file using the National Bureau of Standards Data Encryption Standard algorithm.

Syntax: ENCRYPT_FILE -or- ENCF
 I or INPUT=file
 O or OUTPUT=file
 DK or DES_KEY=integer or string or name

 DM or DES_MODE=keyword
 DIV or DES_INITIAL_VECTOR=integer or string
 or name
 PA or PRESERVE_ATTRIBUTES=boolean
 STATUS=status variable

Parameters: DK - (attribute: SECURE)
 the cryptographic key

type	requirement
integer	each byte of the binary representation of the integer must have odd parity
string	exactly 8 ASCII-128 characters
name	a valid NOS/VE name with exactly 8 characters

DM - (attribute: SECURE)
 the decryption mode to be use

ab	option and meaning
ECB	ELECTRONIC_CODEBOOK use the basic Data Encryption Standard algorithm
CBC	CIPHER_BLOCK_CHAINING use the cipher block chaining mode of operation
CFB	CIPHER_FEEDBACK use the cipher feedback mode of operation (feedback unit data length is 64 bits)
OFB	OUTPUT_FEEDBACK use the output feedback mode of operation (feedback unit data length is 64 bits)

 (default: ELECTRONIC_CODEBOOK)

DIV - (attribute: SECURE)
 define a seed value for CBC, CFB, OFB
 (ignored for ECB)
 (same requirements as DK)

PA - TRUE => NOS/VE file attributes of the
input file are preserved
FALSE => the file attributes are not
preserved
(the length of the input file
must be a multiple of 8 bytes)

See also: DECRYPT_FILE

Similar commands: VMS: ENCRYPT (DTRC)

Examples: /encf plain_file crypt_fyle ABCD1234

ENTER_FILE_MANAGER

Begin a File Manager utility session to view, delete, copy,
print files.

Syntax: ENTER_FILE_MANAGER -or-
ENTFM

C or CATALOG or F or FILE=file
P or PROLOG=file
STATUS=status variable

Parameters: C - the catalog to be displayed as the main
display
(default: \$WORKING_CATALOG)

P - your prolog file for FM
(default: \$USER.\$FILE_MANAGER.PROLOG)

Remarks: You may use CREATE_DEFAULT_VARIABLE to define
EUD\$ENTFM_PROLOG as the prolog to be used.

If you switched your working catalog for this
command, you can keep it with the KEEP_CURRENT_
WORKING_CATALOG parameter of the QUIT subcommand.
Otherwise, your working catalog is restored to
the one before ENTFM was used.

See also: EDIT_CATALOG

Examples: /entfm

ENTER_PROGRAMMING_ENVIRONMENT

Access the Programming Environment.

Syntax: ENTER_PROGRAMMING_ENVIRONMENT -or-
ENTPE

DP or DEFAULT_PROCESSOR=keyword
EC or ENVIRONMENT_CATALOG=file
STATUS=status variable

Parameters: DP - one of: C, COBOL, CORAL, FORTRAN Version 1,
FORTRAN Version 2, PASCAL
(default: FORTRAN)

EC - the catalog in which PE is to maintain its
files
(default: \$USER.PROGRAMMING_ENVIRONMENT)

Examples: /entpe ec=\$user.proj_a

EXECUTE_COMMAND

Execute a single command asynchronously in a new task.

Syntax: EXECUTE_COMMAND -or-
EXEC
C or COMMAND=string

TN or TASK_NAME=name
CF or COMMAND_FILE=file
EE or ENABLE_ECHOING=boolean
STATUS=status variable

Parameters: C - the command to be executed

TN - the name of the task

CF - the input file for the command (used as
\$COMMAND)
(default: no file is used)

EE - YES => the command is echoed to the terminal
NO => no echoing

Remarks: Utility commands cannot be executed this way.

Examples: /exec 'copy_file file1 file2'

EXECUTE_INTERSTATE_COMMAND

Execute a NOS command.

Syntax: EXECUTE_INTERSTATE_COMMAND -or-
EXEIC

C or COMMAND=list of string
STATUS=status variable

Parameters: C - one or more NOS commands to be executed

See also: CREATE_INTERSTATE_CONNECTION; DELETE_INTERSTATE_
CONNECTION

```

Examples:  /creic      <-- open the connection
FA/exeic ('BEGIN,MSAUDIT,,A.', ..
          'REWIND,A.', ..
          'FILE,A,BT=C,RT=Z,MRL=80.')
```

^-- the NOS dayfile for the command
is displayed and put into file
\$LOCAL.ZSZ#Z@Z_170_DAYFILE --
file "A" is a local file on NOS

```

FA/fmu_dir = $unique
          ^-- get unique filename for FMU
          directives
FA/collect_text $local.fmu_dir
          ^-- create file of FMU (File
          Maintenance Utility) directives
          (prompt will be "ct? ")
ct? SET_INPUT_ATTRIBUTES A C170
          ^-- "A" is the NOS local file;
          "C170" means NOS machine format
ct? SET_OUTPUT_ATTRIBUTES MSAUDIT_OUT
          ^-- "MSAUDIT_OUT" is the NOS/VE file
**
FA/include_line 'fmu_dir=$local.temp'
          ^-- execute FMU to copy NOS local
          file to NOS/VE permanent file
FA/delic      <-- close the connection
```

EXECUTE_TASK

Execute a program.

```

Syntax:  EXECUTE_TASK  -or-
        EXET
        -----
        F    or FILES or FILE=list of file
        P    or PARAMETERS or PARAMETER=list of file
        L    or LIBRARIES or LIBRARY=string
        M    or MODULES or MODULE=list of: keyword or
              file
        SP   or STARTING_PROCEDURE=program_name
        LM   or LOAD_MAP=file
        LMO  or LOAD_MAP_OPTIONS or
              LOAD_MAP_OPTION=keyword or
              list of keyword
        PV   or PRESET_VALUE=keyword
        TEL  or TERMINATION_ERROR_LEVEL=keyword
        SS   or STACK_SIZE=integer
        DI   or DEBUG_INPUT=file
        DO   or DEBUG_OUTPUT=file
        AF   or ABORT_FILE=file
        DM   or DEBUG_MODE=boolean
        TN   or TASK_NAME=name
        AO   or ARITHMETIC_OVERFLOW=boolean
        ALOS or ARITHMETIC_LOSS_OF_SIGNIFICANCE=boolean
```

DF or DIVIDE_FAULT=boolean
 EO or EXPONENT_OVERFLOW=boolean
 EU or EXPONENT_UNDERFLOW=boolean
 FI or FPI or FP_INDEFINITE=boolean
 FLOS or FP_LOSS_OF_SIGNIFICANCE=boolean
 IBD or IBDPD or INVALID_BDP_DATA=boolean
 STATUS=status variable

- Parameters:
- F - object files or libraries to be loaded unconditionally
(default: determined by M and SP parameters; if all omitted: \$LOCAL.LGO)
 - P - passed to the program as its parameter list
 - L - object libraries to be searched (added to start of program library list)
OSF\$TASK_SERVICES_LIBRARY - add NOV/VE task services library to the list
(default: program library list contains:
 - . local library list
 - . NOS/VE task services library
 - . text-embedded libraries
 - . job library list
 - . job debug library list (if DM=ON))
 - M - modules to be loaded unconditionally from the program library list libraries
(default: determined by F and SP parameters; if F, M, SP all omitted: \$LOCAL.LGO)
 - SP - entry point to begin execution
(default: the last transfer symbol loaded)
 - LM - file to hold load map
(default: job default value (see DISPLAY_PROGRAM_ATTRIBUTES))
 - LMO - one or more of:

ab	lo	meaning
	NONE	no load map
S	SEGMENT	segment map
B	BLOCK	block map
EP	ENTRY_POINT	entry point map
CR	CROSS_REFERENCE	entry point cross-reference
	ALL	all maps

(default: default load map option (see DISPLAY_PROGRAM_ATTRIBUTES))

PV - value for presetting all uninitialized program memory

ab	pv	meaning
Z	ZERO	all zeros
FPI	FLOATING_POINT_	floating-point
	INDEFINITE	indefinite
I	INFINITY	floating-point
		infinite
AO	ALTERNATE_ONES	alternating 0 and
		1 bits (leftmost
		is 1)

(default: job default value (see DISPLAY_PROGRAM_ATTRIBUTES))

TEL - WARNING (W), ERROR (E), FATAL (F)

(default: job default value (see DISPLAY_PROGRAM_ATTRIBUTES))

(initial default: ERROR)

SS - maximum length (bytes) of the run-time stack

(default: job default value (see DISPLAY_PROGRAM_ATTRIBUTES))

DI - file with Debug commands

(default: job default value (see DISPLAY_PROGRAM_ATTRIBUTES))

DO - file to hold Debug output

(default: job default value (see DISPLAY_PROGRAM_ATTRIBUTES))

AF - file with Debug commands if program aborts

(default: job default value (see DISPLAY_PROGRAM_ATTRIBUTES))

DM - controls Debug mode (ON or OFF)

(default: job default value (see DISPLAY_PROGRAM_ATTRIBUTES))

TN - program is executed asynchronously and is referred to by this name

(default: executed synchronously)

AO - controls whether arithmetic overflow causes an interrupt (ON or OFF)

ALOS - controls whether arithmetic loss of significance causes an interrupt (ON or OFF)

DF - controls whether divide fault causes an interrupt (ON or OFF)

- EO - controls whether exponent overflow causes an interrupt (ON or OFF)
- EU - controls whether exponent underflow causes an interrupt (ON or OFF)
- FI - controls whether floating point indefinite causes an interrupt (ON or OFF)
- FLOS - controls whether floating point loss of significance causes an interrupt (ON or OFF)
- IBD - controls whether invalid BDP data causes an interrupt (ON or OFF)

See also: SET_PROGRAM_ATTRIBUTES

Examples: /execute_task (obj1,obj2)
/exet obj3 mdm=on

EXIT Transfer control out of a structured statement, command or function procedure, or command utility.

Syntax: EXIT
designator
WHEN boolean expression
WITH expression

Parameters: designator - the labeled structures statement, command procedure or utility to be exited

bool_exp - TRUE => the enclosing statement is exited
FALSE => it is not exited
(default: FALSE)

expression - specifies the completion status of a command procedure -or- the value returned by a function procedure:
cmd proc - must have type STATUS
(default: normal status)
fcn proc - any type value

Valid forms: EXIT designator
EXIT WHEN bool_exp
EXIT designator WHEN bool_exp
EXIT designator WITH expression
EXIT WHEN bool_exp WITH expression
EXIT designator WHEN bool_exp WITH expression

Remarks: The WHEN clause is evaluated at the point of reference, instead of continually (as with the WHEN statement).

WHEN and WITH clauses may appear separately or together (in either order).

Structured statements with outstanding calls to other blocks (such as one with an active INCLUDE_FILE) cannot be exited.

See also: WHEN

Similar commands: NOS: REVERT

Examples: PROCEDURE test (status)

```
...
EXIT test WITH ..
    $status(false,'US',1,'Test failed')
...
```

EXPAND_SOURCE_FILE

Expand a text file as though it were a deck in an SCU library.

Syntax: EXPAND_SOURCE_FILE -or-
EXPSF

F or FILE=file

C or COMPILE=file

DA or DEBUG_AIDS=keyword

ISM or INPUT_SOURCE_MAP=file

OSM or OUTPUT_SOURCE_MAP=file

SC or SELECTION_CRITERIA=file

W or WIDTH=integer

LI or LINE_IDENTIFIER=keyword

AB or ALTERNATE_BASE or
ALTERNATE_BASES=list of file

L or LIST=file

ED or EXPANSION_DEPTH=integer

DO or DISPLAY_OPTION or
DISPLAY_OPTIONS=keyword
STATUS=status variable

Parameters: C - output file with expanded text
(default: COMPILE)

DA - controls the disposition of screen debugging
information

dt	meaning
DT	write it to OSM file
NONE	no output

DT write it to OSM file

NONE no output

(default: NONE)

ISM - input screen debugging information for F
file (output source map from when the file
was created)
(default:)

OSM - file to receive screen debugging information
(default: OUTPUT_SOURCE_MAP)

SC - file with selection criteria subcommands
COMMAND - enter subcommands interactively
(default: no selection criteria)

W - length of expanded lines, excluding line
identifiers
(default: 0 (zero))

LI - line identifier placement

ab	option	meaning
R	RIGHT	to right of text
L	LEFT	to left of text
	NONE	no identifiers

(default: NONE)

AB - one or more additional libraries to be
searched for decks

ED - number of levels of COPY and COPYC
directives to process - those beyond are
not expanded (just copied)
(default: all are expanded)

DO - display options

ab	option	meaning
B	BRIEF	decks and library origins not listed
F	FULL	library origin listed if more than one library used

(default: BRIEF)

EXPLAIN Open an on-line manual.

Syntax: EXPLAIN -or-
EXP

S or SUBJECT=name
M or MANUAL=file
L or LIST=file
ED or EXPAND_DEPTH=integer
STATUS=status variable

Parameters: S - search string
(default: the main menu for a CONTEXT
manual and the table of contents
for a Topics manual)

M - the on-line manual to be displayed
(for a list of CDC_provided manuals, see
catalog \$SYSTEM.MANUALS)

L - (default: MANUAL_PAGES in your working
catalog)

ED - the number of the highest level of topics to
be displayed initially
(default: 1; range: 1 to 10)

Similar commands: NOS: EXPLAIN
VMS: HELP

Examples: /explain
/exp /m=fortran
/exp 'get1' scl

EXPLAIN_MESSAGE

Request a description of a system message.

Syntax: EXPLAIN_MESSAGE -or-
EXPM

C or CONDITION=integer
ID or I or IDENTIFIER=string
STATUS=status variable
CLV\$PREVIOUS_STATUS=status

Parameters: C - condition code
(default: the code for the last response
message you received -- if normal,
the first page of the on-line
manual is displayed)

ID - two-character product identifier

CLV\$PREVIOUS_STATUS - reserved

Examples: /expm

EXTRACT_SOURCE_LIBRARY

Extract decks from a base library to use as a separate library.

Syntax: **EXTRACT_SOURCE_LIBRARY** -or-
 EXTSL

```
-----  
      D  or DECKs or DECK=keyword or  
          list of: name or range of name  
-----  
req ->  I  or INTERLOCK=boolean or keyword or name  
-----  
      SC or SELECTION_CRITERIA=file  
      B  or BASE=file  
-----  
req ->  R  or RESULT=file  
-----  
          STATUS=status variable
```

Parameters: D - decks to be copied
 ALL - all decks
 (default: from the criteria file)

I - name of the user reserving the extracted
 decks - written into the subinterlock field
 for each extracted deck and in the original
 interlock field of each deck in the
 extracted library
 NONE - no interlock
 FALSE - no interlock

SC - file from which selection criteria records
 are read
 (default: decks are selected using the DECK
 parameter)

B - the source library
 (default: SOURCE_LIBRARY)

R - the new source library

Similar commands: COS, NOS: UPDATE
 VMS: LIBRARY

Examples: /extsl d=(dk1,dk4..dk6) ..
 ../i=false b=oldpl r=newlib

FICHE (DTRC) Print a NOS/VE file on microfiche (COM).

Syntax: FICHE

```

F or FILE=file
T or TITLE=string
-----
J or JOB=keyword
S or SHIFT=integer
H or HDR or HEADER=file
STATUS=status variable

```

Parameters: F - the file to be printed on microfiche

T - the eye-readable title, that is, the text to be printed in large letters across the top of the microfiche (1-32 characters)
(must be enclosed in apostrophes)

J - job on microfiche to process the file

job	meaning

COMP	compilation output (2 index frames)
COMPF	COMP with a box around each frame
COMQ	standard output without index frames
COMQF	COMQ with a box around each frame
COMDA	there is a 2-line eye-readable title in the file specified by the HEADER parameter

(Default: COMQ)

S - specifies whether the file is to be shifted one column to the right before printing

s	meaning

0	no shifting - the file already has Fortran carriage control characters in column 1 of each line
1	the file does not have Fortran carriage control in column 1 and will be shifted one column to the right to produce a single-spaced listing

H - file containing COM program name and eye-readable title records, such as is needed by COMDA (the COM program must have been defined by the Computer Center before it can be used)
(may NOT be file \$INPUT)

Remarks: FICHE passes the parameters to the BEGIN,FICHE command on NOS. It does this via a NOS/VE batch job which it creates.

You can monitor the jobs with DISPLAY_JOB_STATUS (DISJS). When it completes, the dayfiles will be

Similar commands: NOS: BEGIN, FICHE (DTRC)
VMS: FICHE (DTRC)

FILE MANAGEMENT UTILITY

```
Syntax:      FILE_MANAGEMENT_UTILITY  -or-
             FILMU                     -or-
             FMU
             I  or INPUT=file
             -----
             O  or OUTPUT=file
             D  or DIR or DIRECTIVE or DIRECTIVES=file
             L  or LIST=file
             ED or ERROR_DISPOSITION=keyword
             STATUS=status variable
```

```
Parameters: I - specify only if DIRECTIVES omitted.
```

```
O - specify only if DIRECTIVES omitted.
```

```
D - file containing FMU directives
```

```
L - diagnostic message and FMU summary
```

```
ED - controls termination if any output error
```

	ab	option	meaning
	--	-----	-----
	A	ABORT	FMU aborts
	NA	NO_ABORT	FMU continues writing other files

```
(default: ABORT)
```

Remarks: If INPUT and OUTPUT are specified, FMU does a simple copy.

If DIRECTIVES is specified, the directives give the input and output files and the reformatting to be done.

Examples: /fmu i=myinfyl o=myoutfyl
 ^-- simple file copy
 /fmu d=mydir
 ^-- reformat data

FOR Control repetition of a list of statements.

Syntax: label: FOR EACH variable IN list_expression DO
 statement list
 FOREND label

-or-

label: FOR
 variable=initial TO final BY step DO
 statement list
 FOREND label

Parameters: label - the name of the FOR block - may be used
 by CYCLE or EXIT

Remarks: The limits are determined at the start of the
 loop, so altering values of the list expression
 of the control variable, initial, final, or step
 elements in the statement list has no effect on
 the FOR/FOREND operation.

If INITIAL = FINAL or STEP = 0, the statement
 list is executed once.

FORMAT_SCL_PROCEDURE

Format a file of SCL commands to improve readability.

Syntax: FORMAT_SCL_PROCEDURE -or-
 FORMAT_SCL_PROC -or-
 FORMAT_SCL_PROCEDURES -or-
 FORMAT_SCL_PROCS -or-
 FORSCLP -or-
 FORSP
 I or INPUT=file
 O or OUTPUT=file

 PW or PAGE_WIDTH=integer
 IIC or INITIAL_INDENT_COLUMN=integer
 KC or KEY_CHARACTER=string
 UDF or UTILITY_DEFINITION_FILE=file
 PCT or PROCESS_COLLECT_TEXT=boolean
 T or TRANSLATE=boolean <-- not at DTRC
 STATUS=status variable

Parameters: I - must have the following file attribute values:

ATTRIBUTE	values
FILE_CONTENTS	LEGIBLE or UNKNOWN
FILE_PROCESSOR	SCL or UNKNOWN
FILE_STRUCTURE	DATA or UNKNOWN

O - must has the same attributes as I

PW - must be $\geq$ IIC + 64
(default: 110)

IIC - starting column of the first output line
(default: 1)

KC - this character in column 1 of a line
inhibits reformatting of the line
(default: *)

UDF - file listing the commands that start and
end command utilities - if you create your
own utilities, you need to create a file
that lists all the command utilities used
in the procedure
(default: only system-defined utility
names and terminators are
included in the formatted output)

PCT - TRUE => lines within COLLECT_TEXT and
similar commands are to be
formatted

FALSE => these lines are not formatted
(default: FALSE)

Examples: /forsp procl proclp

FORTTRAN Compile a Fortran program.

Syntax: FORTTRAN -or-
FTN

```

-----
I   or INPUT=file
B   or BINARY or BO or BINARY_OBJECT=file
L   or LIST=file
CD  or COMPILE_DIRECTIVES=boolean
DA  or DEBUG_AIDS=list of keyword
DC  or DEFAULT_COMPILE=keyword
E   or ERROR=file
EL  or ERROR_LEVEL=keyword
EE  or EXPRESSION_EVALUATION=keyword or
                                     list of keyword
FS  or FORCED_SAVE=boolean
ISM or INPUT_SOURCE_MAP=file

```

LO or LIST_OPTION or
 LIST_OPTIONS=keyword or list of keyword
 MD or MACHINE_DEPENDENT=keyword
 OTD or ONE_TRIP_DO=boolean
 OL or OPTIMIZATION_LEVEL or OPTIMIZATION or
 OPT=keyword
 OO or OPTIMIZATION_OPTIONS=keyword of
 list of keyword
 RC or RUNTIME_CHECKS=keyword or
 list of keyword
 SL or SEQUENCED_LINES=boolean
 SD or STANDARDS_DIAGNOSTICS=keyword
 TEL or TERMINATION_ERROR_LEVEL=keyword
 TM or TARGET_MAINFRAME=keyword <-- not at DTRC
 STATUS=status variable

Parameters: L - (default: batch: \$LIST; interactive: \$NULL)

CD - ON - process C\$ directives
 OFF - do not process C\$ directives
 (default: ON)

DA - debugging options
 option meaning

 NONE no debugging options
 ALL all debugging options (DT and PC)
 DT generate line number and symbol
 tables
 PC generate argument checking
 information in the object code
 (default: NONE)

DC - character weight table (collating sequence)
 for character expressions, CHAR, ICHAR
 ab option meaning
 --
 U USER user-specified weight table
 (DISPLAY)
 F FIXED fixed weight table (ASCII)
 (default: FIXED)

EL - (I, W, F, C)
 (default: WARNING)

EE - controls compiler expression evaluation
 ab option and meaning

 C CANONICAL
 expressions are evaluated according
 to precedence rules
 ME MAINTAIN_EXCEPTIONS
 don't eliminate of instructions
 which might cause run-time errors

MP MAINTAIN_PRECISION
 don't change floating-point
 operations to a mathematically
 equivalent form

OSM OVERLAPPING_STRINGS_MOVES
 character assignment v=exp may have
 same positions in v and exp

R REFERENCE
 call intrinsic functions by
 reference to get Fortran-generated
 error messages instead of system-
 generated messages
 (default: NONE; multiple: (op,...,op))

FS - ON - variables and arrays in subprograms
 retain values are saved on RETURN or
 END (same as SAVE in every subprogram)
 OFF - not saved
 (default: OFF)

ISM - file with the source map generated by the
 OUTPUT_SOURCE_MAP option of the SCU
 EXPAND_DECK command
 (default: \$NULL)

LO - compiler output listing options

lo	meaning
A	list symbolic name attributes
R	cross reference listing
M	A + DO loop and common block maps
S	list source program
SA	S + list lines turned off by C\$ LIST
O	object code listing (do not use at DTRC unless requested by User Services)
NONE	no output listing

(default: S; multiple: (op,...,op))

MD - controls diagnosis of machine-dependent
 capabilities

ab	option	meaning
	NONE	do not diagnose machine dependencies
I	INFORMATIONAL	treat as informational errors
W	WARNING	treat as warning errors
F	FATAL	treat as fatal errors

(default: NONE)

OTD - ON - DO loops executed at least once
 OFF - DO loops executed at least zero times
 (default: OFF)

OL - (OPT) controls optimization
 option meaning

option	meaning
HIGH	slow compile, fast execution
LOW	fast compile, slow execution
DEBUG	stylized for debugging

(default: LOW)

OO - not used at DTRC

SL - ON - source program is in sequenced format
 OFF - source program is in nonsequenced format

SD - controls non-ANSI diagnosis

ab	option	meaning
	NONE	do not flag nonstandard usages
I	INFORMATIONAL	treat as informational errors
W	WARNING	treat as warning errors
F	FATAL	treat as fatal errors

(default: NONE)

TEL - minimum severity level for returning abnormal status (I, W, F, C)

Similar commands: COS: CFT; CFT77; CF77
 NOS: FTN, FTN5
 VMS: FORTRAN

Examples: /fortran i=my_source lo=(s,m,r)

FUNCTION

First statement of an SCL function defining the function's names, attributes, and parameters.

Syntax: FUNCTION (attributes) function_names (
 parameter definition
 ...
)
 statement list
 FUNCEND function_name

Parameters: attributes - any of:

- . the name of a help module
- . XDCL - the function can be called from outside its object library and by commands within that library

. ADVANCED - the function is not listed by DISCLE unless requested by the DO parameter

. HIDDEN - the function is not listed by DISCLE

func_names - the function name and its aliases:
name, alias, ..., alias
(must begin with \$)

param_defs - define the function parameter:
parameter_name : (attributes) type
expression = default

Remarks: Function_name is optional on the FUNCEND statement -- if used, it must be the first function name:

```
function display_number, display_numbers, ..
        disn
...
FUNCEND display_number
```

The statement list may not contain PROCEDURE/PROCEND or another FUNCTION/FUNCEND.

An SCL function constitutes its own block.

A function must reside in an object library.

Functions cannot have attributes VAR or SECUDE.

When called, parameters are specified positionally, and, therefore, cannot have the BY_NAME attribute.

Similar commands: VMS: .COM file to define global variable

FUNCTION

(UTILITY subcommand) Declare an entry in a utility function table.

Syntax: FUNCTION
N or NAMES or NAME=data_name or
list of data_name

P or PROCESSOR=name
A or AVAILABILITY=keyword

Parameters: N - the name of the function

P - the name of the processor for the function
(default: the first name of the function is the name of the processor)

A - (attribute: BY_NAME)

specify whether the function appears in a
command list entry display

ab	option	meaning
NU	NORMAL_USAGE	function appears in a command list entry display
AU	ADVANCED_USAGE	it does not appear unless you specifically request it
H	HIDDEN	it does not appear

(default: NORMAL_USAGE)

GENERATE_COMMAND_TABLE

Generate command and function tables for a command environment.

Syntax: GENERATE_COMMAND_TABLE -or-
GENCT
I or INPUT=file
O or OUTPUT=file

PW or PAGE_WIDTH=integer
STATUS=status variable

Parameters: I - the file with the command table declaration
O - the file to receive the CYBIL representation
of the command table
PW - the output file page width (31..110)
(default: output file's page width
attribute forced to be in 31..110)

GENERATE_SCU_EDIT_COMMANDS

Compare a deck and a text source file and generate editing
commands to make the deck match the source file.

Syntax: GENERATE_SCU_EDIT_COMMANDS -or-
GENSEC
D or DECK=keyword or name
S or SOURCE=file
EC or EDIT_COMMANDS=file

B or BASE=file
TD or TERMINATING_DELIMITER=string
LSS or LEADING_SPACES_SIGNIFICANT=boolean
STATUS=status variable

Parameters: D - the deck to which the generated editing subcommands apply
 ALL - the source file is to include the *DECK directives

S - file with a modified version of the deck

EC - the file to receive the editing commands and text

B - the source library where the deck resides (default: SOURCE_LIBRARY)

TD - characters that mark the end of inserted text in the editor commands file (default: ///END\\)

LSS - TRUE => leading blanks are significant
 FALSE => leading blanks are not significant

Remarks: The source file text must not contain line identifiers.

The source file text must not contain DECK directives unless DECK=ALL is used.

If it doesn't matter how many blanks precede the text in a line, specify LSS=TRUE to inhibit generation of editing commands whose sole purpose is to change the number of leading blanks.

GET_FILE

Copy a NOS file to NOS/VE.

Syntax: GET_FILE -or-
 GETF
 T or TO=file

 F or FROM=name or string
 DC or DATA_CONVERSION=keyword
 U or ID or USER=name or string
 PW or PASSWORDS or PASSWORD=list of name
 C or CY or CYCLE=integer <-- not at DTRC
 STATUS=status variable

Parameters: T - the NOS/VE file (may be positioned)

F - the NOS file
 (an SCL name or 1- to 31-character string)
 (default: the file name component of TO)

DC - type of conversion to be done
 dc meaning

B60	60-bit NOS word placed into rightmost bits of NOS/VE word with leftmost 4 bits set to 0 (zero)
B56	rightmost 56 bits of NOS words are packed into contiguous NOS/VE bits ignoring the leftmost 4 NOS bits
A6	6/12 NOS display code converted to ASCII
A8	12-bit NOS ASCII converted to 7-bit ASCII
D63	6-bit NOS display code (63-character set) converted to 7-bit ASCII
D64	6-bit NOS display code (64-character set) converted to 7-bit ASCII

(default: A6)

U - alternate NOS user

PW - (attribute: SECURE)
 NOS password required to access the file

Remarks: SET_FILE_ATTRIBUTE may precede the GET_FILE to establish attribute values to be preserved.

The NOS file must not be on a cartridge.

CHANGE_LINK_ATTRIBUTE prior to GETF identifies the accounting and user information for accessing the file.

DC=D64, A6, or A8, requires zero-byte terminated (Z-type) NOS records.

Embedded NOS EORs are eliminated from the NOS/VE copy of the file.

See also: MSFETCH; MSSTORE; REPLACE_FILE

Similar commands: NOS: FCOPY
 VMS: HFT FETCH (DTRC); FTP

Examples: /getf ve_file nosfile

GET_LINE

Read a line from a file into a string variable.

Syntax: GET_LINE -or-
 GET_LINES -or-
 GETL -or-
 ACCEPT_LINE -or-
 ACCEPT_LINES -or-
 ACCL
 V or VARIABLE=string variable or
 array of string variables
 I or INPUT=file

 P or PROMPT=string
 LC or LINE_COUNT=integer variable
 STATUS=status variable

Parameters: V - the variable into which lines are read

 I - the file to be read

 P - the input prompt string
 (if I is not assigned to a terminal, there
 is no prompting)
 (default: ENTER <variable name>)

 LC - will contain the number of lines read by
 GETL

Remarks: Don't use standard files such as \$INPUT.

Similar commands: VMS: READ

Examples: PROCEDURE write_to_list (
 input, i: file = \$required
 status)
 VAR
 input_list: list of string
 VAREND
 get_line v=input_list i=input p='? '
 display_value input_list
 PROCEND write_to_list

 ...
 /write_to_1st i=input <-- input is the job file
 ? 123
 ? 456
 ? 789
 ? *EOI
 123
 456
 789

HELP Open an on-line manual.

Syntax: HELP -or-
 H

S or SUBJECT=data_name or string or
 application
M or MANUAL=file
L or LIST=file
 STATUS=status variable

Parameters: S - the index topic to be searched
 (enclose in apostrophes if any embedded
 blanks)
 (default: main menu (CONTEXT manual) or
 table of contents (Topics manual))

M - the on-line manual to be displayed
 (default: SCL (Commands and Functions);
 if in a utility with an on-line
 manual, that manual is displayed)

L - the file to receive the listable output
 (default: MANUAL_PAGES)

Remarks: HELP without parameters:
 . if after an error, the NOS/VE Diagnostic
 Messages manual (file name MESSAGES)
 . with a utility, the utility's on-line manual
 or the utility description in the NOS/VE
 Commands and Functions manual
 . neither of the above, the table of contents
 of the NOS/VE Commands and Functions manual

Similar commands: NOS: EXPLAIN; HELPME
 VMS: HELP

Examples: /help m=fortran
 ^-- the on-line Fortran manual
 /help forttran
 ^-- the NOS/VE Commands and
 Functions manual searching for
 FORTRAN
 /help \$quote
 ^-- display the description of the
 \$QUOTE function
 /ftn='get_line'
 /help s=ftn
 ^-- search for FTN
 (FTN not evaluated)
 /explain s=ftn m=scl
 ^-- search for GET_LINE
 (FTN evaluated)

/HELP CCRM .NSYS.CCRM

^-- the on-line Computer Center
Reference Manual

IF Conditional execution of one or more statement lists.

Syntax: IF boolean expression THEN
 statement list
 ELSEIF boolean expression THEN
 statement list
 ELSE
 statement list
 IFEND

Parameters: boolean expression - TRUE => execute the
 following statement
 list

Command prompt: IF/

Similar commands: COS: IF-ELSEIF-EXITIF-ENDIF
 NOS: IF
 VMS: IF-THEN-ELSEIF-ENDIF

INCLUDE_COMMAND

Include a string containing SCL statements into the command stream.

Syntax: INCLUDE_COMMAND -or-
 INCC
 C or COMMAND=string

 EE or ENABLE_ECHO or ENABLE_ECHOING=boolean
 STATUS=status variable

Parameters: C - string of semicolon-separated statements

EE - TRUE => echo the command
 FALSE => do not echo the command
 (default: TRUE)

See also: INCLUDE_FILE; INCLUDE_LINE

INCLUDE_FILE

Insert a text file of SCL commands into the command stream.

Syntax: INCLUDE_FILE -or-
 INCF
 F or FILE=file

 P or PROMPT=string
 U or UTILITY=keyword or name
 STATUS=status variable

Parameters: P - the input prompt string if F is assigned to
 an interactive terminal
 (if U is specified, the utility's prompt is
 used)
 (default: incf)

U - the utility to be associated with INCF
 (U is intended for use within UTILITY/
 UTILITYEND providing a mechanism for
 associating QUIT commands entered
 interactively with the specified utility
 block)
 NONE - use to avoid association with a
 utility
 (default: NONE)

Similar commands: VMS: @

Examples: /incf commands

INCLUDE_LINE

Insert a string containing SCL statements into the command stream.

Syntax: INCLUDE_LINE -or-
 INCL
 SL or STATEMENT_LIST=string

 EE or ENABLE_ECHO or ENABLE_ECHOING=boolean
 U or UTILITY=keyword or name
 STATUS=status variable

Parameters: SL - the string of semicolon-separated statements
 to be processed as a statement list

EE - TRUE => echo the command
 FALSE => do not echo the command
 (default: TRUE)

U - the utility to be associated with INCL
 (U is intended for use within UTILITY/
 UTILITYEND providing a mechanism for

associating QUIT commands entered
interactively with the specified utility
block)

NONE - use to avoid association with a
utility
(default: NONE)

Examples: /tim = '\$time(ampm)'
/dat = '\$date(month)'
/command_list = ..
../'display_vaule (' // tim // ',..
../' // dat // ')'
/include_line command_list
10:59 AM
January 24, 1990

INITIALIZE_TERMINAL

Change the terminal settings.

Syntax: INITIALIZE_TERMINAL -or-
INIT

STATUS=status variable

Parameters:

Remarks: For INIT to change the terminal settings, the
terminal definition must contain at least two
APPLICATION_STRING statements of the format:
APPLICATION_STRING NAME='LINE_INIT' ..
OUT='text sent to terminal'

APPLICATION_STRING NAME='SCREEN_INIT' ..
OUT='text sent to terminal'

where you supply the text strings.

JOB Generate and submit a batch job.

Syntax: JOB

UJN or USER_JOB_NAME or JN or JOB_NAME=name
CTL or CPU_TIME_LIMIT=keyword or integer
ERT or EARLIEST_RUN_TIME=date_time
JAD or JOB_ABORT_DISPOSITION=keyword
JC or JOB_CLASS=name
JDBU or JOB_DEFERRED_BY_USER=boolean
JER or JOB_EXECUTION_RING=integer
JQ or JOB_QUALIFIERS or
JOB_QUALIFIER=keyword or list of name
JRD or JOB_RECOVERY_DISPOSITION=keyword
LRT or LATEST_RUN_TIME=date_time

MAXWS or MAXIMUM_WORKING_SET=keyword or integer
 OF or OPERATOR_FAMILY=name
 OU or OPERATOR_USER=name
 OD or OUTPUT_DISPOSITION=keyword or file
 SL or SRU_LIMIT=keyword or integer
 S or STATION=keyword or name
 SM or SUBSTITUTION_MARK=keyword or string
 UI or USER_INFORMATION=string
 SJN or SYSTEM_JOB_NAME=name variable or
 string
 STATUS=status variable

Parameters: CTL - maximum CPU time for the job
 (job abort limit if job plus monitor time
 exceeds this value)

ctl	meaning
integer	maximum CPU seconds
SYSTEM_DEFAULT	system default used
UNLIMITED	maximum allowed by the system
(default: explicit JC: minimum of class' TL or user's validation TL	
CJ=AUTOMATIC: minimum of system default TL or user's validation TL)	

ERT - earliest time to execute -or-
 NONE - job may run anytime
 (default: NONE)

JAD - action if job aborts because of system failure

jad	meaning
RESTART	resubmit job to start at beginning
TERMINATE	discard the job

JC - the job class

jc	meaning
AUTOMATIC	automatically assign to job class depending on job's attributes

JDBU - TRUE => job is deferred (held in input queue indefinitely)
 FALSE => job is not deferred
 (default: JDBU job attribute used)

JER - the job's execution ring
(range: 4-13, must be at least at great as your minimum ring validation)

JQ - site-defined qualifiers for limiting a job to a specific job class or set of classes

jq	meaning
1-5 qualifiers	restrict the job
NONE	no job qualifiers
SYSTEM_DEFAULT	use the system default value

JRD - action if job aborts because of system interrupt

jrd	meaning
CONTINUE	attempt to resume the job - if not successful, use JAD
RESTART	resubmit job to start at beginning
TERMINATE	discard the job

LRT - latest time to execute (if not executed, job is dropped) -or-
NONE - job may run anytime
(default: NONE)

MTL - maximum number of tape drives used simultaneously

mtl	meaning
integer	1-2
SYSTEM_DEFAULT	system default used
UNLIMITED	maximum allowed by the system

(default: system default is used)

MAXWS - the maximum number of pages the job requires (may be restricted by job class)

maxws	meaning
integer	
SYSTEM_DEFAULT	system default used
UNLIMITED	maximum allowed by the system

(default: determined by job class)

OF -
OU -

ODI - the disposition of standard output
 ab odi

-----	-----
DAO	DISCARD_ALL_OUTPUT
DSO	DISCARD_STANDARD_OUTPUT
P	PRINTER
WQ	WAIT_QUEUE (output goes into
	user's \$WAIT_QUEUE)
(default: PRINTER)	

SL - maximum SRUs for the job
 (job abort limit if this is exceeded)
 sl meaning

-----	-----
integer	maximum SRUs
SYSTEM_DEFAULT	system default used
UNLIMITED	maximum allowed by the
	system
(default: explicit JC: minimum of class'	
	SL or user's
	validation SL
	CJ=AUTOMATIC: minimum of system
	default SL or
	user's
	validation SL)

S - the default I/O station name for the
 job's output files
 (if JOB_DESTINATION_USAGE is PRIVATE,
 this must specify the control facility
 name)
 AUTOMATIC - use the system default

SM - (substitution marks cannot be used to
 generate the JOBEND statement)

UI - up to 256 characters to be passed to the
 job and to all generated output files
 (default: the UI associated with the job)

Remarks: JOB is followed by SCL statements ending with
 JOBEND.

LOGIN and LOGOUT are added to your job by the
 system.

Command prompt: job/

See also: LOGIN; SUBMIT_JOB

Examples: /JOB AMDS001 15
 job/<SCL statements>
 job/JOBEND

KERMIT Begin a KERMIT file transfer session.

Syntax: KERMIT

T or TAKE=file
STATUS=status variable

Parameters: T - reserved

See also: XMODEM_RECEIVE; XMODEM_SEND; YMODEM_RECEIVE;
YMODEM_SEND

Similar commands: NOS: FTP; KERMIT-170; XMODEM
VMS: FTP; KERMIT

Examples: /kermit

LOGIN Provide access to NOS/VE batch services.

Syntax: LOGIN
LU or LOGIN_USER or U or USER=name
PW or PASSWORD=name

LF or LOGIN_FAMILY=name
LA or LOGIN_ACCOUNT=name
LP or LOGIN_PROJECT=name
CTL or CPU_TIME_LIMIT=keyword or integer
ERT or EARLIEST_RUN_TIME=date_time
JAD or JOB_ABORT_DISPOSITION=keyword
JC or JOB_CLASS=name
JDBU or JOB_DEFERRED_BY_USER=boolean
JER or JOB_EXECUTION_RING=integer
JQ or JOB_QUALIFIERS or
JOB_QUALIFIER=keyword or list of name
JRD or JOB_RECOVERY_DISPOSITION=keyword
LRT or LATEST_RUN_TIME=date_time
MTL or MAGNETIC_TAPE_LIMIT=keyword or integer
MAXWS or MAXIMUM_WORKING_SET=keyword or integer
SL or SRU_LIMIT=keyword or integer
UI or USER_INFORMATION=string
UJN or USER_JOB_NAME or JN or JOB_NAME=name

Parameters: See JOB

Remarks: ERT, LRT, JDBU are ignored for interactive jobs.

See also: SUBMIT_JOB

Similar commands: NOS: LOGIN; HELLO

Examples: /login amds my_password

LOGOUT Terminate a batch or interactive job.

Syntax: LOGOUT

See also: LOGIN; JOB; SUBMIT_JOB

Similar commands: NOS, VMS: LOGOUT

Examples: /logout

LOOP Repeat forever a statement list.

Syntax: label: LOOP
 statement list
 LOOPEND label

Parameters: label - the name of the LOOP block - may be used
 by CYCLE or EXIT

Remarks: LOOP may be exited only by an EXIT statement.

Similar commands: VMS: WHILE TRUE DO

MANAGE_FORM

Begin a MANAGE_FORMS utility session.

Syntax: MANAGE_FORM -or-
 MANAGE_FORMS -or-
 MANF

VC or VARIABLE_CREATION=keyword
VE or VARIABLE_EVALUATION=keyword
STATUS=status variable

Parameters: VC - controls creation of variables when a form
 is opened.

vc	meaning
FORM_VARIABLE	create one variable for each form
SINGLE	create one variable for each variable on the form
NONE	create no variables
(default: FORM_VARIABLE)	

VE - controls how the utility evaluates variables

ve	meaning
AUTOMATIC	READ_FORMS or SHOW_FORMS updates all variables

MANUAL you must use GET_FORM_VARIABLE
 and REPLACE_FORM_VARIABLE
 subcommands to update them

Subcommand prompt: mf/

MANAGE_JOB

Begin a utility that manages the selection and control of one or more jobs.

Syntax: MANAGE_JOB -or-
 MANAGE_JOBS -or-
 MANJ

 STATUS=status variable

MANAGE_OUTPUT

Begin a utility that manages the selection and control of one or more output files.

Syntax: MANAGE_OUTPUT -or-
 MANAGE_OUTPUTS -or-
 MANO

 STATUS=status variable

MEASURE_PROGRAM_EXECUTION

Begin a program measurement utility session.

Syntax: MEASURE_PROGRAM_EXECUTION -or-
 MEAPE

 STATUS=status variable

Subcommand prompt: MPE/

Similar commands: COS: SPY
 NOS: HOTSPOT
 VMS: PCA

Examples: /meape
 MPE/set_program_description target_text=lgo
 MPE/execute_instrumented_task
 MPE/display_program_profile output=my_file
 MPE/quit
 /

MERGE Merge sorted records from one or more files and write them in sorted order to a single output file.

Syntax: MERGE

```

-----
F      or FROM=list of file
T      or TO=file
K      or KEY=list of list of range of any
D      or DIR or DIRECTIVES or
        DIRECTIVES_FILE or DF=list of file
L      or LIST=file
LO     or LIST_OPTION or
        LIST_OPTIONS=list of keyword
E      or ERROR=file
EL     or ERROR_LEVEL=keyword
RP9    or RESERVED_POSITION_9=boolean
ENR    or ESTIMATED_NUMBER_RECORDS=range of
        integer
ERF    or EXCEPTION_RECORD_FILE=file
CC     or C170_COMPATIBLE=boolean
OD     or OMIT_DUPLICATIONS=boolean
OFL    or OWNFL or OWNCODE_FIXED_LENGTH=integer
OMRL   or OWNCODE_MAXIMUM_RECORD_LENGTH=integer
OP1    or OWN1 or OWNCODE_PROCEDURE_1=name
OP2    or OWN2 or OWNCODE_PROCEDURE_2=name
OP3    or OWN3 or OWNCODE_PROCEDURE_3=name
OP4    or OWN4 or OWNCODE_PROCEDURE_4=name
OP5    or OWN5 or OWNCODE_PROCEDURE_5=name
ROO    or RETAIN_ORIGINAL_ORDER or RETAIN or
        RET=boolean
CSN    or COLLATING_SEQUENCE_NAME or
        SEQN=name
CSS    or COLLATING_SEQUENCE_STEP or
        SEQ=list of range of any
CSR    or COLLATING_SEQUENCE_REMAINDER or
        SEQR=boolean
CSA    or COLLATING_SEQUENCE_ALTER or
        SEQA=boolean
        STATUS=status variable
S      or SUM=list of list of range of any
ZLR    or ZERO_LENGTH_RECORDS=keyword
VMIO   or VERIFY_MERGE_INPUT_ORDER or
        VERIFY or VER=boolean
LCT    or LOAD_COLLATING_TABLE=list of name
RA     or RESULT_ARRAY or RES=array of integer
        variables

```

Parameters: See SORT.

Similar commands: NOS: MERGE

Examples: /merge (\$user.infy11,\$user.infy12) ..
 ../\$user.outfyl ..
 ../key=((1,10)) verify=yes
 ^-- merge files infy11 and infy12
 producing infy13 -- leftmost
 10 characters are used as the
 key -- the sort is ascending
 ASCII -- the input record
 sort order is checked

MSACCES (DTRC) Access the Mass Storage System (MSS).

Syntax: MSACCES
 MPW=name

 STATUS=status variable

Parameters: MPW - (Attributes: SECURE)
 your MS access password

Remarks: MSACCES is required before using the MSx commands.

Similar commands: COS: MSACCES (DTRC)
 NOS: login process
 VMS: HFT ACCESS (DTRC)

Examples: /MSACCES mymssp

MSAUDIT (DTRC) Obtain a sorted audit of your MSS files.

Syntax: MSAUDIT

 O or OUTPUT=file
 LO or LIST_OPTION=keyword
 PW or SPW or SHOW_PW or SHOW_PASSWORD or
 SHOW_PASSWORDS=keyword
 UN or USER_NAME=name
 F or FILE or FILES=list of file
 STATUS=status variable

Parameters: LO - the list option

ab	option	meaning
F	FULL	4 lines per file + cost per day and per month
I	INTERMEDIATE	SHORT + date created, date last accessed, number of streams (direct files), password (if requested), charge number (your files), cost per day

N NAMES 7 filenames per line
separated by indirect/
direct, files on
cartridge are flagged

S SHORT length, filename, CT,
M (permissions), number
of uses, indirect/direct
(default: FULL)

PW - indicates whether individual file passwords
are to be included in the listing -- your
files only

pw	meaning
YES or YE or Y	list file passwords
NO or N	do not list them

(default: NO)

UN - alternate user name to display another user's
files (only those files you are permitted to
see)
(default: your own files)

F - a single filename or a filename with
wildcards
(must be enclosed in apostrophes)
(default: all filenames are listed)

Remarks: MSACCES is required before using the MSx commands.

See also: MSFILES

Similar commands: NOS: CATLIST
VMS: HFT DIRECTORY; MSSAUDIT (both DTRC)

Examples: /MSACCES,mymssp
 ^-- required to access MSS

/MSAUDIT <-- 4 lines per file on \$OUTPUT

/MSAUDIT,0=audout,LO=I,SPW=Y
 ^-- 1 line per file (including
 each file's password) written to
 file AUDOUT in your current
 catalog

/MSAUDIT out1 s UN=other
 ^-- short list of user OTHER's MSS
 files in file OUT1 in your
 current catalog

/MSAUDIT F='V*****'
 ^-- all files starting with "V"

MSCATLIST (DTRC) The NOS CATLIST command.

Syntax: MSCATLIST

 DN=integer
 EF=string
 FN=string
 L=file
 LO=keyword
 NA=keyword
 PN=name
 PW=keyword
 R=name
 UN=name
 WB=keyword
 STATUS=status variable

Parameters: See Appendix I: CATLIST

See also: MSAUDIT

Similar commands: NOS: CATLIST
 VMS: HFT DIRECTORY

MSCHANG (DTRC) Change the attributes of a Mass Storage System file.

Syntax: MSCHANG
 F or FN or FILE=name

 NFN or NEW_FILE_NAME=name or INPUT=file
 PW or PASSWORD=name
 CT or CATEGORY_TYPE=keyword
 M or FILE_ACCESS_MODE=keyword
 BR or BACKUP_REQUIREMENTS or
 BACKUP_REQUIREMENT=keyword
 CP=integer
 STATUS=status variable

Parameters: F - the MSS file whose attributes are to be changed

NFN - new file name
 (default: no change)

PW, CT, M, BR - same as MSSTORE
 (default for each: no change)

CP - controls changing the account number for the file

cp	meaning
0	do not change the account number
1	change the account number to the MSACCES account number

(default: 0)

Remarks: MSACCES is required before using the MSx commands.

Similar commands: COS: MSCHANG (DTRC)
 NOS: CHANGE
 VMS: HFT CHANGE (DTRC)

Examples: /MSCHANG myfile CT=PU
 ^-- make MYFILE a public file

 /MSCHANG,myfile,CP=1
 ^-- change the account number

 /MSCHANG myfile NFN=newfile M=E
 ^-- change the name of the file and
 make it execute-only

 /MSCHANG,myfile,BR=CR
 ^-- off-station backup at additional
 cost

MSFETCH (DTRC) Fetch a file from the Mass Storage System.

Syntax: MSFETCH
 F or MFN or FROM=name

 T or TO=file
 DC or DATA_CONVERSION=keyword
 U or USER or UN or ID=name
 PW or PASSWORD or PASSWORDS=list of name
 STATUS=status variable

Parameters: F - the MSS file to be fetched

T - the NOS/VE name for the file

DC - how the file is to be transferred

dc	meaning
B60	NOS word --> rightmost 60 NOS/VE bits; leftmost 4 NOS/VE bits set to 0
B56	rightmost 56 NOS bits --> contiguous NOS/VE bits; leftmost 4 NOS bits ignored (use for NOS/VE object libraries, SCU libraries, permanent file backup files)
A6	NOS 6/12 display code --> 7-bit ASCII
A8	NOS 8/12 display code --> 7-bit ASCII
D63	NOS 63-char " " --> 7-bit ASCII
D64	NOS 64-char " " --> 7-bit ASCII
(default: A6)	

U - alternate user name (to fetch another user's file)

PW - password for other user's file

Remarks: MSACCES is required before using the MSx commands.

Similar commands: COS: MSFETCH (DTRC)
 NOS: ATTACH; GET
 VMS: HFT FETCH (DTRC)

Examples: /MSACCES,MPW=mymssp
 /MSFETCH,mssfyl1,in1
 ^-- MSSFYL1 is retrieved and stored
 as file IN1 in your current
 catalog

/MSFETCH mssfyl2 in2 D64
 ^-- MSSFYL2 is retrieved, converted
 from 64-character NOS Display
 Code and stored as file IN2 in
 your current catalog

/MSFETCH thatfyl in3 UN=abcd
 ^-- fetch another user's file THATFYL
 and store as your file IN3

MSPASSW (DTRC) Change your Mass Storage System access password.

Syntax: MSPASSW
 O or OLD or OPW or OLD_PASSWORD=name
 N or NEW or NPW or NEW_PASSWORD=name

 STATUS=status variable

Parameters: O - (Attributes: SECURE)
 your current MSS access password

N - (Attributes: SECURE)
 your new MSS access password

Remarks: MSACCES is required before using the MSx commands.

Similar commands: COS: MSPASSW (DTRC)
 NOS: PASSWOR
 VMS: HFT PASSWORD (DTRC)

Examples: /MSACCES,MPW=mymssp
 ^-- access the MSS

/MSPASSW,OLD=mymssp,NEW=numssp
 -or-
 /MSPASSW mymssp numssp
 -or-
 /MSPASSW N=numssp O=mymssp
 ^-- the above are the same

A6 7-bit ASCII --> NOS 6/12 display code
 A8 7-bit ASCII --> NOS 8/12 display code
 D63 7-bit ASCII --> NOS 63-char " "
 D64 7-bit ASCII --> NOS 64-char " "
 (default: A6)

TY - the type of file on the MSS

type	meaning
D or DA or DIR or DIRECT	direct access
I or IA or IND or INDIRECT	indirect access
SAME	same as existing MSS file; if none, then direct access

(default: SAME)

CT - the new category type

ct	access
P or PR or PRIVATE	only you
PU or PUBLIC	everyone
S or SPRIV or SEMI_PRIVATE	everyone with a record of all accesses

(default: no change)

M - the file access mode

ab	m	meaning
E	EXECUTE	execute only
R	READ	read only
RU	READUP	read and update
RA	READAP	read and append
RM	READMD	read and modify
U	UPDATE	update only
A	APPEND	append only
M	MODIFY	modify only
W	WRITE	write only

(default: not specified - same as WRITE)

BR - the backup requirements for the file

br	meaning
CR	off-station
Y	on-station
MD	only if on disk
N	no backup

(MD and N are not recommended)
(default: not specified (same as Y))

PW - (Attributes: SECURE)
the new password -or-
CLEAR - remove the existing password
(default: no password for the file)

Similar commands: COS: MSSTORE (DTRC)
NOS: DEFINE; SAVE
VMS: HFT STORE; MSSBACKUP (both DTRC)

Examples: /MSACCES,MPW=mymsspw
 ^-- access the MSS
/MSSTORE,in1,mssfyl1
 ^-- IN1 is stored as private, direct
 file MSSFYL1

/MSSTORE in2 mssfyl2 D64 PW=fylepw
 ^-- IN2 is stored as private file
 MSSFYL2 (even if MSSFYL2 already
 exists) in CDC Display Code --
 FYLEPW is the password required
 for another user to access the
 file -- if MSSFYL2 does not
 exist, this will be a direct file

/MSSTORE in3 mssfyl3 A6 I PU
 ^-- IN3 is stored as public, indirect
 file MSSFYL3 in 64-character
 Display Code

NEW_ROUTINES

(DTRC) Display a description of recently-added routines or
copy the information into a file.

Syntax: NEW_ROUTINES -or-
NEWR

O or OUTPUT=file
 STATUS=status variable

Parameters: O - the file to which the information is to be
 copied
 (default: \$OUTPUT)

See also: NEWS; OLDNEWS

Similar commands: NOS: BEGIN,NEWRTNS (DTRC)
VMS: NEWROUTINES (DTRC)

Examples: /NEWR <-- display information about new
 routines at your terminal
/NEWR \$local.r
 ^-- copy the information into your
 local file R

NEWS (DTRC) Display the current news file or copy it into a file.

Syntax: NEWS

O or OUTPUT=file
STATUS=status variable

Parameters: O - the file to which the news file is to be
copied
(default: \$OUTPUT)

See also: NEW_ROUTINES;

Similar commands: NOS: BEGIN,NEWS (DTRC)
VMS: NEWS (DTRC)

Examples: /NEWS <-- display the news at your terminal
/NEWS \$local.n
^-- copy the news file into your
local file N

OLD_NEWS

(DTRC) Display the old news file or copy it into a file.

Syntax: OLD_NEWS -or-
OLDN

O or OUTPUT=file
STATUS=status variable

Parameters: O - the file to which the old news file is to be
copied
(default: \$OUTPUT)

See also: NEW_ROUTINES; NEWS

Similar commands: NOS: BEGIN,OLDNEWS (DTRC)
VMS: OLDNEWS (DTRC)

Examples: /OLD_NEWS <-- display the old news at your
terminal
/OPLDN \$local.o
^-- copy the news file into your
local file O

OPEN_FILE_MIGRATION_AID

Begin the file migration environment.

Syntax: OPEN_FILE_MIGRATION_AID -or-
OPEFMA

PJC or PARTNER_JOB_CARD=string
STATUS=status variable

Parameters: PJC - job statement parameters for NOS job in
 NOS syntax
 (defaults: batch: infinite time
 interactive: no job parameters)

Remarks: OPEFMA initiates a NOS batch job (partner job).

Subcommand prompt: fa/

Examples: /opefma
 fa/open_l70_state
 fa/execute_command 'attach,binfile'
 fa/ec 'file,binfile,fa=sq,bt=i,mrl=90.
 fa/close_environment
 fa/execute_migration_task ..
 fa/migration_file=(binfile, ..
 fa../newfile,cl70_to_cl80) file=lgo
 fa/close_environment

PASCAL Compile a PASCAL program.

Syntax: PASCAL

```

-----
I   or INPUT=file
B   or BINARY or BO or BINARY_OBJECT=file
L   or LIST=file
E   or ERROR=file
EL  or ERROR_LEVEL=keyword
LO  or LIST_OPTION or
     LIST_OPTIONS=keyword or list of keyword
OL  or OPTIMIZATION_LEVEL or OPTIMIZATION or
     OPT=keyword
DA  or DEBUG_AIDS=list of keyword
RC  or RUNTIME_CHECKS=keyword or
     list of keyword
SD  or STANDARDS_DIAGNOSTICS=keyword
TEL or TERMINATION_ERROR_LEVEL=keyword
ISM or INPUT_SOURCE_MAP=file
ISM or INPUT_SOURCE_MAP=file
     STATUS=status variable
  
```

Parameters: I - \$INPUT - you will be prompted for each
 source line -- end with *EOI

B - \$NULL - syntax check only

E - \$NULL - no error file is written

EL - (W, F, C)
 (default: WARNING)

LO - compiler output listing options

lo	meaning
A	list entity attributes
O	object code listing (do not use at DTRC unless requested by User Services)
R	cross reference listing
S	list source program
NONE	no output listing

(default: S; multiple: (op,...,op))

OL - controls optimization

option	meaning
LOW	keep constant values in registers
DEBUG	stylized for debugging

(default: LOW)

DA - debugging options

option	meaning
DT	generate line number and symbol tables
ALL	all debugging options (DT)
NONE	no debugging options

(default: NONE)

RC - controls execution-time checking

rc	meaning
F	errors in file and buffer variables
E	mis-use or pointer and buffer variables and NEW and DISPOSE procedures
R	range checking for subrange and set assignment and case variables
S	array subscript bounds
NONE	no checks
ALL	all run-time checks

(default: NONE)

SD - controls non-ISO/non-ANSI diagnosis
(level,std) meaning

(W)	ISO errors => WARNING
(F)	ISO errors => FATAL
(W,ISO)	ISO errors => WARNING
(F,ISO)	ISO errors => FATAL
(W,ANSI)	ANSI errors => WARNING
(F,ANSI)	ANSI errors => FATAL
(NONE)	standard errors not diagnosed

(default: NONE)

TEL - minimum severity level for returning
abnormal status (W, F, C)

ISM - (attribute: BY_NAME)
the file generated by the OSM option on the
SCU EXPAND_DECK command
(default: \$NULL)

Similar commands: NOS, VMS: PASCAL

Examples: /pascal pas_source lo=none

POP Delete the current version of an environment object or
variable and restore to its former value.

Syntax: POP environments

Parameters: environments - one or more (comma or space
separated) of:
environment object
environment variable
COMMAND_LIST
FILE_CONNECTIONS
INTERACTION_STYLE
MESSAGE_LEVEL
NATURAL_LANGUAGE
PROGRAM_ATTRIBUTES
WC or WORKING_CATALOG

Remarks: Error if environment object or variable was not
established (pushed).

See also: PUSH

PRINT_FILE

Print one or more files.

Syntax: PRINT_FILE -or-
PRINT_FILES -or-
PRIF
F or FILES or FILE=list of file

CB or COMMENT_BANNER=string
C or COPIES=integer
DM or DATA_MODE=keyword
D or DEVICE=keyword or name
EPT or EARLIEST_PRINT_TIME=keyword or
date_time
EC or EXTERNAL_CHARACTERISTICS=keyword or
name
FC or FORMS_CODE=keyword or string
LPT or LATEST_PRINT_TIME=keyword or
date_time

OF or OPERATOR_FAMILY or
 DESTINATION_FAMILY or DF=name
 OU or OPERATOR_USER or
 STATION_OPERATOR or SO=name
 OC or OUTPUT_CLASS=keyword
 ODBU or OUTPUT_DEFERRED_BY_USER=boolean
 ODE or OUTPUT_DESTINATION=name or string
 ODU or OUTPUT_DESTINATION_USAGE or
 DESTINATION_USAGE or DU=keyword or
 name
 OP or OUTPUT_PRIORITY=keyword
 PD or PURGE_DELAY=keyword or time_increment
 RHD or REMOTE_HOST_DIRECTIVE or
 DUAL_STATE_ROUTE_PARAMETERS or
 DSRP=string
 RB or ROUTING_BANNER=string
 S or STATION=keyword or name
 UFN or USER_FILE_NAMES or
 USER_FILE_NAME=list of file
 UI or USER_INFORMATION=string
 VPD or VERTICAL_PRINT_DENSITY=keyword
 VLP or VFU_LOAD_PROCEDURE=keyword or name
 SFN or SYSTEM_FILE_NAME=name variable or
 list of name variable
 STATUS=status variable

Parameters: CB - string to be displayed with the output
 (default: CB job attribute - if null, the
 file name)

C - number of copies

DM - one of:

ab	option	meaning
C	CODED	the file contains codes to be interpreted by the printer
T	TRANSPARENT	print the file without conversion or interpretation

(cannot be used if ODU=DUAL_STATE)

D - name that, together with STATION,
 identifies the printer -or-
 AUTOMATIC - any printer with the EC and
 FC needed

EPT - the earliest time to print the file -or-
 NONE - print any time

EC - 1-to 6-character string -or-
 NORMAL - a printer with EC=NORMAL
 (mapped to A9 on NOS)

- FC - 1-to 6-character string -or-
NORMAL - a printer with FC=NORMAL
- LPT - if not printed by this time, the file is discarded -or-
NONE - no restrictions on print time
- OF - family name of a private station or remote system operator that, together with OU, identifies the private or remote operator who can print or receive the file (requires ODU=PRIVATE or NTF)
- OU - family name of a private station or remote system operator that, together with OF, identifies the private or remote operator who can print or receive the file (requires ODU=PRIVATE or NTF)
- OC - define the initial and maximum priorities, and the aging interval and factor for the output file
NORMAL - only class available
- ODBU - TRUE => defer printing
FALSE => do not defer printing
- ODE - where the output file is to be sent for printing if ODU=QTF or NTF
- ODU - one of:
- . the kind of CDCnet print station
 - . the queue file transfer application to be used to send the file to a remote system
- | odu | meaning |
|------------|---|
| PUBLIC | a public CDCnet batch I/O station
(OF, OU, ODE, RHD ignored) |
| PRIVATE | a private CDCnet batch I/O station
(ODE, RHD ignored) |
| DUAL_STATE | printed by NOS
(only FC, C, RB, RHD used) |
| QTF | send the file to remote ODE station |
| NTF | send the file to remote ODE NTF system |
- (default: ODU used unless ODI=LOCAL, then system default used)

- OP - priority increment to add to the initial priority
- | value | increment |
|--------|-----------|
| LOW | 0 |
| MEDIUM | 1500 |
| HIGH | 3000 |
- PD - how long to retain the file in the output queue after printing -or-
NONE - no delay
- RHD - either:
- . a NOS/VE PRINT_FILE command
 - . a NOS ROUTE command
 - . the NOS ROUTE command parameters (ignored unless ODU has the appropriate value)
- RB - string to be displayed with the output (default: RB job attribute - if null, the user name for the file)
- S - the I/O station name for sending the file
AUTOMATIC - use the system default
- UFN - names to go with the files listed in FILE (default: each file's name)
- VPD - vertical print density
- | vpd | meaning |
|-------|-------------------------|
| SIX | 6 lines per inch |
| EIGHT | 8 lines per inch |
| NONE | any printer |
| FILE | vpd of source file used |
| | 6 - SIX |
| | 7-12 - EIGHT |
- VLP - name of a procedure with the vertical forms unit (VFU) image to be loaded to control the printer
- SFN - name of a variable of type LIST to receive the generated system names for the files (default: ignored)

Remarks: Except for UFN, DM and SFN, all defaults are the corresponding job attributes.

See also: CHANGE_OUTPUT_ATTRIBUTE, DISPLAY_OUTPUT_STATUS

Similar commands: COS: DISPOSE to a front-end
NOS: ROUTE
VMS: PRINT; QPRINT (DTRC)

```

Examples:  /print_file file=myout copies=5
           ^-- print 5 copies of file MYOUT
           /display_output_status all
           ^-- produces the following (sample)
Output_State      : printing
System_File_Name  : $0990_0102_aad_1439
User_File_Name    : myout
/disos name=all
           ^-- after printing produces
           None Were Found.
           = = = = =
           /prif fortran_listing /odu=dual_state ..
           ../rhd='dc=pr,ec=a9'
           ^-- print on NOS printer

```

PROCEDURE

First statement of an SCL procedure defining the procedure's names, attributes, and parameters.

```

Syntax:    PROCEDURE (attributes) procedure_names (
           parameter definition
           ...
           )
           statement list
PROCEND procedure_name

```

Parameters: attributes - any of:

- . the name of a help module
- . XDCL - the procedure can be called from outside its object library and by commands within that library
- . ADVANCED - the procedure is not listed by DISCLE unless requested by the DO parameter
- . HIDDEN - the procedure is not listed by DISCLE

proc_names - the procedure name and its aliases:
name, alias, ..., alias

param_defs - define the procedure parameter:
parameter_names : (attributes) type
expression = default

Remarks: Procedure_name is optional on the PROCEND statement -- if used, it must be the first procedure name:

```

PROCEDURE display_number, display_numbers, ...
disn
...
PROCEND display_number

```

The statement list may not contain another PROCEDURE/PROCEND.

An SCL procedure constitutes its own block.

WARNING: NOS/VE has two procedure interpreters. The old one is invoked by PROC and is not recommended because it does not support all features. The new one is invoked by PROCEDURE.

Similar commands: COS: PROC
NOS: .PROC
VMS: .COM file

Examples: PROCEDURE display_number, display_numbers, disn (
number, numbers, n : list 1..10 of range of ..
integer -100..100 = \$required
output, o : file = \$output
status)

...
PROCEND

^-- procedure name is DISPLAY_NUMBER
(alternate names DISPLAY_NUMBERS
and DISN) -- parameter NUMBER
(or NUMBERS or N) is required
and is a list of 1-10 value sets,
each having 1 or 2 values, each
value an integer in the range
-100 thru 100

PURGE (DTRC) Delete cycles of one or more files keeping the specified number of high cycles.

Syntax: PURGE
F or FILE or FILES=list of file

K or KEEP=integer
L or LOG=keyword
STATUS=status variable

Parameters: K - the number of high cycles of each file to keep
(default: 1)

L - controls logging the process
LOG - display the filenames as they are deleted
(synonym: L)
NOLOG - do not display them
(synonyms: N, NOL)
(default: NOLOG)

See also: DELETE; DELETE_ALL_CYCLES

Similar commands: COS: PURGE,KEEP=
NOS: PURGE
VMS: DELETE

Examples: /purge e*

PUSH Temporarily change environment objects or variables in an SCL procedure.

Syntax: PUSH environments

Parameters: environments - one or more (comma- or space-separated) of:
environment object
environment variable
COMMAND_LIST
FILE_CONNECTIONS
INTERACTION_STYLE

MESSAGE_LEVEL
NATURAL_LANGUAGE
PROGRAM_ATTRIBUTES
WC or WORKING_CATALOG

Remarks: Only the most recently pushed version of the system environment can be referenced or changed.

A specific object can be pushed only once in a procedure.

See also: POP

PUSH_COMMANDS

Move the command list entry containing the procedure that called this statement to the top of the command list.

Syntax: PUSH_COMMANDS

Remarks: The command list entry is returned to its original position when the procedure is exited or when the command list is restored using POP.

PUT_LINE

Write lines to a file.

Syntax: PUT_LINE -or-
PUT_LINES -or-
PUTL
L or LINES=list of string

O or OUTPUT=file
STATUS=status variable

Parameters: L - the lines to be written

Remarks: PUTL never adds page titles or format effectors. The first character is assumed to be a format effector (carriage control).

COLLECT_TEXT is faster for writing more than one line in succession since it opens the file once, while PUTL opens the file each time it is called.

PUTL allows substitution for variables and string expressions.

See also: COLLECT_TEXT

Similar commands: COS, NOS: NOTE
VMS: WRITE


```
Examples:  /put_lines lines=( ..
           ../' Today's date:  '//$date(month) ..
           ../' The current time:  '//$time(ampm) ..
           ../' Welcome to NOS/VE.')
```

Today's date: January 25, 1990
The current time: 11:10 AM
Welcome to NOS/VE.

RATES (DTRC) Display the current Computer Center rates or put them into a file.

```
Syntax:      RATES
              -----
              0 or OUTPUT=file
              STATUS=status variable
```

Parameters: 0 - the file to which the news file is to be copied
(default: \$OUTPUT)

Similar commands: NOS: BEGIN,RATES (DTRC)
VMS: HELP RATES (DTRC)

```
Examples:  /RATES      <-- Display the rates at your
           terminal

           /RATES $LOCAL.R
           ^-- Copy the rates to local file R
```

RELEASE_RESOURCE
Release tape reservations.

```
Syntax:      RELEASE_RESOURCE      -or-
              RELEASE_RESOURCES     -or-
              RELR
              -----
              MT9$800=keyword or integer
              MT9$1600=keyword or integer
              MT9$6250=keyword or integer
              STATUS=status variable
```

Parameters: MT9\$den - number of tape drives of density
 <den> which the job no longer needs
 ALL - all resource of this kind
 (default: 0)

See also: RESERVE RESOURCE

Similar commands: NOS: RETURN

```
Examples:  /reir mt9$1600=1 mt9$6250=2
           ^-- release 1 1600-cpi and
              2 6250-cpi tape drives
```

REPEAT Conditionally repeat a statement list.

Syntax: label: REPEAT
 statement list
 UNTIL boolean_expression

Parameters: label - the name of the REPEAT block - may be
 used by CYCLE or EXIT

 bool_exp - the terminating condition

Subcommand prompt: repeat/

Similar commands: VMS: DO WHILE

REPLACE_FILE

Copy a NOS/VE file to NOS direct file, replacing any existing file.

Syntax: REPLACE_FILE -or-
 REPF
 T or TO=file

 F or FROM=file
 DC or DATA_CONVERSION=keyword
 U or ID or USER=name or string
 PW or PASSWORD or TURNKEY or TK=name
 XR or EXCLUSIVE_ACCESS=name <-- not at DTRC
 C or CY or CYCLE=integer <-- not at DTRC
 STATUS=status variable

Parameters: F - the NOS/VE file (may be positioned)

T - the NOS file
 (default: the file name component of FROM)

DC - type of conversion to be done
 dc meaning

B60	rightmost 60 bits of each NOS/VE word into NOS word with leftmost 4 NOS/VE bits ignored
B56	contiguous NOS/VE bits packed into rightmost 56 bits of NOS words with leftmost 4 NOS bits set to 0 (zero)
A6	7/8 ASCII converted to 6/12 NOS display code
A8	7/8 ASCII converted to 8/12 NOS ASCII
D63	7/8 ASCII converted to 6-bit NOS display code (63-character set)

D64 7/8 ASCII converted to 6-bit NOS
display code (64-character set)
(default: A6)

U - alternate NOS user

PW - (attribute: SECURE)
NOS password only required if you don't
own the file

Remarks: CHANGE_LINK_ATTRIBUTUE prior to GETF identifies
the accounting and user information for accessing
the file.

DC=D64, A6, or A8 create zero-byte terminated (Z-type) NOS records.

See also: GET FILE; MSFETCH; MSSTORE

```
Similar commands:  COS: MSSTORE (DTRC)
                   NOS: PURGE/DEFINE; REPLACE
                   VMS: HFT STORE; MSSBACKUP (both DTRC)
```

```
Examples:  /repf  my_file  myfile
              ^-- copies NOS/VE file MY_FILE to
                 NOS as MYFILE, replacing any
                 existing NOS file with the same
                 name
```

REQUEST MAGNETIC TAPE

Associate a file with a magnetic tape.

```
Syntax:      REQUEST_MAGNETIC_TAPE  -or-
             REQMT
             F    or FILE=file
             -----
             EV   or EVSN or EXTERNAL_VSN=list of: string
                                     or name
             L    or LOG=boolean
             PW   or PASSWORD=keyword or name
             RV   or RVSN or RECORDED_VSN=list of: string
                                     or name
             RMG  or REMOVABLE_MEDIA_GROUP=keyword or name
             R    or RING=boolean
             T    or TYPE or DENSITY or D=keyword
             VOA  or VOLUME_OVERFLOW_ALLOWED=boolean
                 STATUS=status variable
```

Parameters: (attribute: BY NAME, all except F)

F - the file name for the tape -- should reside in \$LOCAL

- EV - the external sticker number (1-6 characters) -- if more than one, they are requested in the order specified (default: RECORDED_VSN is used)
- PW - (additional attributes: ADVANCED, SECURE) the password for all cycles of a file
- RV - the VSN written in the ANSI VOL1 label (1-6 characters) -- of more than one, they are located and verified in the order specified (default: EXTERNAL_VSN is used)
- (note: either EV or RV must be specified)
- RMG - (additional attribute: ADVANCED) 1- to 13-character name of a group of tape volumes managed by RMS that is used to assign volume to the file cycle
- R - specifies if the write ring is needed in each volume of the file
- | value | meaning |
|-------|-------------------------------------|
| TRUE | the ring must be in (write or read) |
| FALSE | the ring must be out (read only) |
- (default: FALSE)
- T - the tape type (density) of the tape(s)
- | value | meaning |
|-----------|-------------------|
| MT9\$800 | 9-track, 800 cpi |
| MT9\$1600 | 9-track, 1600 cpi |
| MT9\$6250 | 9-track, 6250 cpi |
- (default: value of OSD\$REQMT_DEFAULT_DENSITY; if it doesn't exist, MT9\$1600)
- VOA - (additional attribute: ADVANCED) controls whether, for this attachment only, this is a multi-reel volume (default: FALSE)

Remarks: Tape mounting is not requested until the file is used.

New tapes must be blank labeled by the operator.

CHATLA must set REWRITE_LABELS to TRUE for the first use.

Similar commands: NOS: LABEL
VMS: MOUNT

Examples: /reqmt file=\$local.tapel type=mt9\$6250 ..
 ../ev='NA9876' ring=true
 /chatla file=\$local.tapel rewrite_labels=true
 /copy_file file1 \$local.tapel
 /detach_file \$local.tapel

REQUEST_TERMINAL

Associate a file with a terminal in an interactive job.

Syntax: REQUEST_TERMINAL -or-
 REQT
 F or FILE=file

 STATUS=status variable

RESERVE_RESOURCE

Specify the number of tape units to be reserved.

Syntax: RESERVE_RESOURCE -or-
 RESERVE_RESOURCES -or-
 RESR

 MT9\$800=keyword or integer
 MT9\$1600=keyword or integer
 MT9\$6250=keyword or integer
 STATUS=status variable

Parameters: MT9\$den - number of tape drives of density
 <den> which the job needs
 (default: 0)

See also: RELEASE_RESOURCE

Similar commands: NOS: RESOURCE

Examples: /relr mt9\$1600=1 mt9\$6250=2
 ^-- release 1 1600-cpi and
 2 6250-cpi tape drives

RESTORE_FOREIGN_FILES

Begin a RESTORE_FOREIGN_FILES utility session to migrate
NOS tape files to NOS/VE.

Syntax: RESTORE_FOREIGN_FILES -or-
 RESFF

 L or LIST or OUTPUT or O=file
 STATUS=status variable

Parameters: L - (default: \$OUTPUT)

RESTORE_PERMANENT_FILES

Begin a RESTORE_PERMANENT_FILES utility session.

Syntax: RESTORE_PERMANENT_FILES -or-
 RESTORE_PERMANENT_FILE or
 RESPF

 1 or LIST=file
 STATUS=status variable

Remarks: Use SET_LIST_OPTION subcommand before the
 RESTORE_PERMANENT_FILE subcommand to define the
 contents of the list file. Default: modification
 date/time and size for each permanent file cycle.

Subcommand prompt: PUR/

Examples: /respf 1=\$local.taperd_log
 PUR/resec <parameters>
 PUR/quit
 /

RESUME_COMMAND

Resume a job activity interrupted by a pause break.

Syntax: RESUME_COMMAND -or-
 RESEC

 STATUS=status variable

Similar commands: VMS: CONTINUE

REWIND_FILE

Position a file at BOI (beginning-of-information).

Syntax: REWIND_FILE -or-
 RFWF
 F or FILE or FILES=file

 STATUS=status variable

Similar commands: COS, NOS: REWIND

Examples: /rewf source
 ^-- rewind one file
 /rewind_files f=(library,test,source)
 ^-- rewind 3 files

ROUTE_JOB

(for input from cards, or a PC or terminal supporting HASP)
Name a job and the output destination.

Syntax: ROUTE_JOB -or-
 ROUJ

 JN or JOB_NAME=name
 JD or JOB_DESTINATION=name
 JDO or JOB_OUTPUT_DESTINATION=name
 UN or USER_NAME=name
 UF or USER_FAMILY=name
 STATUS=status variable

SELECT_USER_MENU

Begin the NOS/VE user menu system.

Syntax: SELECT_USER_MENU -or-
 SELUM -or-
 MENJ

 PC or PROLOG_CALL=boolean
 NL or NATURAL_LANGUAGE=name
 STATUS=status variable

Parameters: PC - reserved

NL - (default: US_ENGLISH)

SEND_OPERATOR_MESSAGE

Send a message to an operator.

Syntax: SEND_OPERATOR_MESSAGE -or-
 SENOM
 M or MESSAGE=string

 R or RESPONSE=string variable
 OC or OPERATOR_CLASS=keyword
 STATUS=status variable

Parameters: M - the message to be send (1-256 characters)

R - an SCL variable to receive the operator's
 response (>= 256 characters)
 (default: the reply message is put into the
 job's log and, if interactive,
 SOUTPUT)

```

OC - the class of operator
      ab          option
-----
SO     SYSTEM_OPERATOR
RMO    REMOVABLE_MEDIA_OPERATOR
(default: SO)

```

Remarks: The job is suspended until the operator replies.

Similar commands: VMS: REPLY/REQUEST

```

Examples:  /reply=''
           /senom ..
           ../m='Is tape AB1234 in slot 99 yet?' ..
           ../r=reply
           ^-- the job is suspended
           /display_value reply
           SORRY, AB1234 IS NOT IN SLOT 99
           ^-- the operator's reply

```

SET_COMMAND_LIST

Add to and/or delete from the current command list, and/or alter the state of the search mode indicator.

```

Remarks:  Use instead:
           CREATE_COMMAND_LIST_ENTRY (CRECLE)
           DELETE_COMMAND_LIST_ENTRY (DELCLE)
           CHANGE_COMMAND_LIST_ENTRY (CHACLE).

```

SET_DEBUG_LIST

Add or delete debug object libraries to/from the job debug library list.

```

Syntax:    SET_DEBUG_LIST -or-
           SETFL
           -----
           DL or DELETE_LIBRARY or DELETE_LIBRARIES=
               list of: keyword or file
           AL or ADD_LIBRARY or ADD_LIBRARIES=list of
               file
           STATUS=status variable

```

Parameters: DL - list of file - libraries to be deleted
 ALL - delete all libraries
 (default: no libraries deleted)

AL - list of file - libraries to be added at the
 beginning of the job debug
 library list
 (default: no libraries added)

SET_DEBUG_RING

Specify the ring in which Debug is to execute.

Syntax: SET_DEBUG_RING -or-
 SETDR
 R or RING=integer

 STATUS=status variable

Parameters: R - the Debug ring number (1-13)

Examples: /setdr 10

SET_DEFAULT_LOGIN_PROJECT

(DTRC) Set the default login project number (job order number at DTRC) so that it need not be entered at future logins.

Syntax: SET_DEFAULT_LOGIN_PROJECT -or-
 SETDLP

 STATUS=status variable

Remarks: After your login project has been set, just press RETURN at the "Enter Project Number" prompt.

Similar commands: NOS: CHVAL,CN

Examples: /setdlp

SET_FILE_ATTRIBUTES

Set the attributes of a file.

Syntax: SET_FILE_ATTRIBUTES -or-
 SET_FILE_ATTRIBUTE -or-
 SETFA
 F or FILE=file

 AM or ACCESS_MODE or
 ACCESS_MODES=list of keyword
 ARL or AVERAGE_RECORD_LENGTH=integer
 BT or BLOCK_TYPE=keyword
 CC or CHARACTER_CONVERSION=boolean
 CTN or COLLATE_TABLE_NAME=keyword or name
 CPN or COMPRESSION_PROCEDURE_NAME=keyword
 or entry_point_reference
 DP or DATA_PADDING=integer
 EK or EMBEDDED_KEY=boolean
 EEN or ERROR_EXIT_NAME or
 EEPN or
 ERROR_EXIT_PROCEDURE_NAME=keyword
 or name

EL or ERROR_LIMIT=integer
 ERC or ESTIMATED_RECORD_COUNT=integer
 FAP or FILE_ACCESS_PROCEDURE or
 FAPN or
 FILE_ACCESS_PROCEDURE_NAME=keyword
 or name
 FC or FILE_CONTENTS=keyword or name
 FLT or FILE_LABEL_TYPE=keyword
 FL or FILE_LIMIT=integer
 FO or FILE_ORGANIZATION=keyword
 FP or FILE_PROCESSOR=keyword or name
 FS or FILE_STRUCTURE=keyword or name
 FW or FORCED_WRITE=keyword or boolean
 HPN or HASHING_PROCEDURE_NAME=keyword or
 entry_point_reference
 IL or INDEX_LEVEL=integer
 IP or INDEX_PADDING=integer
 IHBC or INITIAL_HOME_BLOCK_COUNT=integer
 IC or INTERNAL_CODE=keyword
 KL or KEY_LENGTH=integer
 KP or KEY_POSITION=integer
 KT or KEY_TYPE=keyword
 LN or LINE_NUMBER=record
 LET or LOCK_EXPIRATION_TIME=integer
 LO or LOGGING_OPTIONS or
 LOGGING_OPTION=list of keyword
 LR or LOG_RESIDENCE=keyword or file
 MAXBL or MAXIMUM_BLOCK_LENGTH=integer
 MAXRL or MAXIMUM_RECORD_LENGTH=integer
 MC or MESSAGE_CONTROL=list of keyword
 MINBL or MINIMUM_BLOCK_LENGTH=integer
 MINRL or MINIMUM_RECORD_LENGTH=integer
 OP or OPEN_POSITION=keyword
 PC or PADDING_CHARACTER=string
 PF or PAGE_FORMAT=keyword
 PL or PAGE_LENGTH=integer
 PW or PAGE_WIDTH=integer
 PV or PRESET_VALUE=integer
 RL or RECORD_LIMIT=integer
 RT or RECORD_TYPE=keyword
 RPB or RECORDS_PER_BLOCK=integer
 SI or STATEMENT_IDENTIFIER=record
 UI or USER_INFORMATION=string
 STATUS=status variable

Parameters: (attribute: all are BY_NAME)

AM - READ, WRITE (APPEND + MODIFY + SHORTEN),
 APPEND, SHORTEN, EXECUTE, NONE
 (default:)

BT - the block type
 ab bt
 -- -----
 SS SYSTEM_SPECIFIED
 US USER_SPECIFIED

CC - conversion between internal character
code and ASCII (TRUE or FALSE)
(default: FALSE)

MAXBL - maximum block length (0-65,497) in bytes

MAXRL - maximum record length (1-65,497) in bytes
(ANSI fixed-length records only)

MINBL - minimum block length (>17) in bytes
(default: 18 for new file)

Remarks: It is recommended that all relevant keyed-file
attributes be specifically set, since the
defaults are sometimes inappropriate.

See also: SET_FILE_ATTRIBUTES

Similar commands: NOS: CHANGE; FILE

Examples: /setfa my_file am=(read,execute)

SET_LINK_ATTRIBUTES

Use CHANGE_LINK_ATTRIBUTES.

SET_PROGRAM_ATTRIBUTES

Change default attributes for subsequent programs in a job..

Syntax: SET_PROGRAM_ATTRIBUTES -or-
SET_PROGRAM_ATTRIBUTE -or-
SETPA

```

-----
LM   or LOAD_MAP=file
LMO  or LOAD_MAP_OPTIONS or
      LOAD_MAP_OPTION=keyword or
      list of keyword
PV   or PRESET_VALUE=keyword
TEL  or TERMINATION_ERROR_LEVEL=keyword
DI   or DEBUG_INPUT=file
DO   or DEBUG_OUTPUT=file
AF   or ABORT_FILE=file
DM   or DEBUG_MODE=boolean
DL   or DELETE_LIBRARY or
      DELETE_LIBRARIES=keyword or
      list of: keyword or file
AL   or ADD_LIBRARY or
      ADD_LIBRARIES=list of: keyword or file
AO   or ARITHMETIC_OVERFLOW=boolean
ALOS or ARITHMETIC_LOSS_OF_SIGNIFICANCE=boolean

```

DF or DIVIDE_FAULT=boolean
 EO or EXPONENT_OVERFLOW=boolean
 EU or EXPONENT_UNDERFLOW=boolean
 FI or FPI or FP_INDEFINITE=boolean
 FLOS or FP_LOSS_OF_SIGNIFICANCE=boolean
 IBD or IBDPD or INVALID_BDP_DATA=boolean
 STATUS=status variable

Parameters: LM - (initial default: \$LOCAL.LOADMAP.\$BOI)

 LMO - NONE, SEGMENT (S), BLOCK (B),
 ENTRY_POINT (EP), CROSS_REFERENCE (CR),
 ALL
 (initial default: NONE)

 PV - ZERO (Z), FLOATING_POINT_INDEFINITE (FPI),
 INFINITY (I), ALTERNATE_ONES (AO)
 (initial default: ZERO)

 TEL - WARNING (W), ERROR (E), FATAL (F)
 (initial default: ERROR)

 DI - (initial default: COMMAND)

 DO - (initial default: \$OUTPUT)

 AF - (initial default: \$NULL)

 DM - default debug mode (ON, OFF)
 (initial default: OFF)

 DL - list of libraries to be deleted -or-
 ALL - reorder your library list
 (OSF\$TASK_SERVICES_LIBRARY - the system
 table)
 (initial default: none deleted)

 AL - list of libraries to be added
 (initial default: none added)

 AO - ON, OFF
 (initial default: ON)

 ALOS - ON, OFF
 (initial default: ON)

 DF - ON, OFF
 (initial default: ON)

 EO - ON, OFF
 (initial default: ON)

 EU - ON, OFF
 (initial default: ON)

FI - ON, OFF
(initial default: ON)

FLOS - ON, OFF
(initial default: OFF)

IBD - ON, OFF
(initial default: ON)

SET_SENSE_SWITCH

Sets sense switches on or off.

Syntax: SET_SENSE_SWITCH -or-
SET_SENSE_SWITCHS -or-
SETSS

N or NAME or JOB_NAME or JN=name
ON=list of range of integer
OFF=list of range of integer

Parameters: N - user- or system-supplied job name (you
control) whose sense switches are to be
altered
(default: the requesting job)

ON - the switches (1-8) to be turned on
(default: none are turned on)

OFF - the switches (1-8) to be turned off
(default: none are turned off)

Similar commands: NOS: SWITCH

Examples: /set_sense_switches on=(1,2,3)
 ^-- turn sense switched 1, 2 and 3
 on in the current job
/setss jn=\$0855_0104_pdq_0861 on=1 off=2
 ^-- turn SS 1 on and SS 2 off in a
 specific job you control
/setss jn=myjob off=(1,2,3,4,5,6,7,8)
 ^-- turn off all sense switches in
 a specific job you control

SHOW_FILE

Display a file in a pop up window that provides editing options.

Syntax: **SHOW_FILE** -or-
 SHOF
 F or FILE=file

 T or TITLE=string
 W or WIDTH=integer
 X=integer
 Y=integer
 STATUS=status variable

Parameters: T - (attribute: BY_NAME)
 the title of the window
 (default: T='Display of contents')

 W - (attribute: BY_NAME)
 the width of the window (characters)
 (default: 78; range: 1-80)

 X - (attribute: BY_NAME)
 X-coordinate of the upper left-hand corner of
 the window
 (default: 5; range: 2-80)

 Y - (attribute: BY_NAME)
 Y-coordinate of the upper left-hand corner of
 the window
 (default: 5; range: 2-30)

Default function keys:

Key label	description
-----	-----
OK	complete comand execution (also RETURN)
Help	display help (EUMSSH_SHOW_FILE)
Copy	copy marked lines to a file (user a file selection window having the same characetristics as the window generated by GET_FILE_SELECTION)
Edit	edit the file using EDIT_FILE
Find	locate text in the display (a dialog window prompts you for the search text)
Mark	toggle the marking of the current line
ExtMrk	extend the marking (or unmarking) from the last mark to the current line
SeeAll	return to the SHOW_FILE window after find
Fwd	page forward
Bkw	page backward
First	go to first line
Last	go to last line

Up put current line at top of screen
Down put current line at bottom of screen

Examples: /shof my_file

SHOW_VALUE

Display a value in a pop up window.

Syntax: SHOW_VALUE -or-
 SHOW_VALUES -or-
 SHOV
 V or VALUE or VALUES=any

 DO or DISPLAY_OPTION or
 DISPLAY_OPTIONS=keyword
 T or TITLE=string
 W or WIDTH=integer
 X=integer
 Y=integer
 STATUS=status variable

Parameters: V - the value to be displayed

DO - display options
 ab option

 CLE COMPRESSED_LABELED_ELEMENTS
 E ELEMENTS or ELEMENT
 LE LABELED_ELEMENTS
 (default: ELEMENTS)

T - (default: 'Display of value')

W - (default: 70; range: 1-80)

Examples: /v = 154

 /shov v

```
-----
|154|
-----Press RETURN to continue-----
/
```

SKIP_TAPE_MARK

Position a tape backward or forward.

Syntax: SKIP_TAPE_MARK -or-
 SKIP_TAPE_MARKS -or-
 SKITM
 F or FILE=file

 D or DIRECTION=keyword
 C or COUNT=integer
 STATUS=status variable

Parameters: D - the direction to skip
 dir position at completion

 FORWARD after <count> tapemarks
 BACKWARD before <count> tapemarks
 (default: FORWARD)

C - the number of tape marks to skip
 (default: 1)

Similar commands: NOS: SKIPEI; SKIPF; SKIPFB; SKIPR

Examples: /skip_tape_mark file=mytape count=4
 /skitm mytape forward 4
 ^-- these two are the same
 /skitm mytape1 d=backward
 ^-- skip backward 1 tapemark on
 tape MYTAPE1

SORT Sort records from one or more files and write them in sorted order to a single output file.

Syntax: SORT

 F or FROM=list of file
 T or TO=file
 K or KEY=list of list of range of any
 D or DIR or DIRECTIVES or
 DIRECTIVES_FILE or DF=list of file
 L or LIST=file
 LO or LIST_OPTION or
 LIST_OPTIONS=list of keyword
 E or ERROR=file
 EL or ERROR_LEVEL=keyword
 RP9 or RESERVED_POSITION_9=boolean
 ENR or ESTIMATED_NUMBER_RECORDS=range of
 integer
 ERF or EXCEPTION_RECORD_FILE=file
 CC or C170_COMPATIBLE=boolean
 OD or OMIT_DUPLICATIONS=boolean
 OFL or OWNFL or OWNCODE_FIXED_LENGTH=integer
 OMRL or OWNCODE_MAXIMUM_RECORD_LENGTH=integer
 OP1 or OWN1 or OWNCODE_PROCEDURE_1=name
 OP2 or OWN2 or OWNCODE_PROCEDURE_2=name
 OP3 or OWN3 or OWNCODE_PROCEDURE_3=name
 OP4 or OWN4 or OWNCODE_PROCEDURE_4=name
 OP5 or OWN5 or OWNCODE_PROCEDURE_5=name
 ROO or RETAIN_ORIGINAL_ORDER or RETAIN or
 RET=boolean
 CSN or COLLATING_SEQUENCE_NAME or
 SEQN=name
 CSS or COLLATING_SEQUENCE_STEP or
 SEQ=list of range of any

CSR or COLLATING_SEQUENCE_REMAINDER or
 SEQR=boolean
 CSA or COLLATING_SEQUENCE_ALTER or
 SEQA=boolean
 STATUS=status variable
 S or SUM=list of list of range of any
 ZLR or ZERO_LENGTH_RECORDS=keyword
 VMIO or VERIFY_MERGE_INPUT_ORDER or
 VERIFY or VER=boolean
 LCT or LOAD_COLLATING_TABLE=list of name
 RA or RESULT_ARRAY or RES=array of integer
 variables

Parameters: F - 1 to 100 input files
 (default: \$LOCAL.OLD)

T - output file
 (default: NEW or \$LOCAL.NEW1 owncode 3
 may override this)

K - 1 to 106 key field definitions:
 (first..last,type,order) -or-
 (first,length,type,order)

param	meaning
first	first byte of key field (leftmost byte is numbered 1)
last	last byte of key field
length	number of bytes in key field (default: 1)
type	name of numeric data format or collating sequence

numeric formats

BINARY	BINARY_BITS
INTEGER	INTEGER_BITS
NUMERIC_FS	NUMERIC_LO
NUMERIC_LS	NUMERIC_NS
NUMERIC_TO	NUMERIC_TS
PACKED	PACKED_NS
REAL	

collating sequences

ASCII	ASCII6
COBOL6	DISPLAY
EBCDIC	EBCDIC6

(default: ASCII)

```
order    sort order
          A (ascending)
          D (descending)
          (default: A)
(default: use the minimum record length
         (default: 0) as the key length)
```

DF - 1 to 100 directives files
(default: the merge is completely
specified in the command)

LO - list options
lo meaning
-- -----
OFF only minimum information
NONE same as OFF
S source directives
DE detailed exception information
(requires EXCEPTION_RECORD_FILE
parameter)
RS record statistics
MS merge statistics
(default: minimum information)

EL - (I, T, W, F, C, NONE)
(default: WARNING)

RP9 - reserved

ENR - integer 1 thru 16,777,215
(not used by MERGE)

ERF - invalid records are written to this file
instead of the output file
(default: no exception processing --
invalid records are written to
the output file)

CC - YES => convert lowercase letters in owncode
routines to uppercase
NO => owncode procedure names are not
converted
(default: NO)

OD - control processing of duplicate records
(those with matching keys)
YES => duplicate records are omitted
NO => duplicate records are not omitted
(default: processed by OWNCODE_
PROCEDURE_5, RETAIN_ORIGINAL_
order, or SUM parameter)

- OFL - integer 1 thru 4096
(default: OWNCODE_MAXIMUM_RECORD_LENGTH
is used)
(this or OMRL is required if TO or FROM
is omitted)
- OMRL - integer 1 thru 4096
(default: the record length attribute of
the first input file read or
output file written is used)
(this or OFL is required if TO or FROM
is omitted)
- OP1 - reserved
- OP2 - reserved
- OP3 - name of the routine to process each
output record
(default: no user-defined output record
postprocessing)
- OP4 - name of the routine which can insert a
record at the end of the output file
(default: no user-defined output file
postprocessing)
- OP5 - name of the routine to process duplicate
records (having equal keys)
(default: no user-defined processing of
matching records)
(OP5 and SUM are mutually exclusive)
- ROO - reserved
- CSN - name of the collating sequence defined by
subsequent collating sequence parameters
(default: no user-defined collating
sequence)
- CSS - one or more value step definitions (value
sets each defining one or more value steps
in the collating sequence (char is ASCII):
('char')
 ^-- one 1-character value step
('char', 'char', ...)
 ^-- one multi-character value step
('char' .. 'char')
 ^-- multiple 1-character value steps
('char' .. 'char', 'char' .. 'char', ...)
 ^-- multiple multi-character value
 steps (all ranges must be the
 same size)
(default: no specific collating sequence
value steps are defined)

CSR - TRUE => a special value step is defined
containing all characters not in
other value steps
FALSE => remaining characters keep their
ASCII positions
(default: FALSE)

CSA - TRUE => all characters in a value step
are changed to the first character
in the value step
FALSE => no character alteration
(default: FALSE)

S - 1 thru 100 sum field definitions:
(first..last,type,repeat_count) -or-
(first,length,type,repeat_count)

param	meaning
first	same as for KEY
last	same as for KEY
length	same as for KEY
type	name of numeric data format (same as for KEY)
repeat_count	number of consecutive sum fields defined by the value set (default: 1)

(OP5 and SUM are mutually exclusive)

ZLR - controls the disposition of zero-length
records read from input files

zlr	meaning
DELETE	the records are deleted and not written to the exception records file
PAD	each such record is processed as a short record
LAST	each such record is written at the end of the output file

VMIO - TRUE => check that all input records are
in sorted order
FALSE => do not check the input order
(default: FALSE)

LCT - load a collating table (predefined by
NOS/VE or user-defined) for use by one or
more keys:
(key type name,collating table name)

RA - a 6-element SCL variable for the return of statistics with the first element defined as the number of statistics to be returned (0 thru 15)

Similar commands: COS, NOS, VMS: SORT

Examples: /sort (\$user.infy11,\$user.infy12) ..
 ../\$user.outfy1 ..
 ../key=((1,10)) verify=yes
 ^-- merge files infy11 and infy12
 producing infy13 -- leftmost
 10 characters are used as the
 key -- the sort is ascending
 ASCII -- the input record
 sort order is checked
 =====
 /sort,key=((1,10))
 ^-- sort columns 1-10 of file
 \$LOCAL.OLD writing file
 \$LOCAL.NEW
 =====
 /sort fy14 fy15 ((9..16),integer,d),(3))
 ^-- sort using two keys: columns
 9-16 contain integers and are
 put into descending order;
 column 3 contains an ASCII
 character and is put into
 ascending order

SOURCE_CODE_UTILITY

Begin an SCU utility session.

Syntax: SOURCE_CODE_UTILITY -or-
 SCU -or-
 SOUCU

STATUS=status variable

Remarks: Use CREATE_LIBRARY or USE_LIBRARY subcommands to initialize the working library for the session. If omitted, SOURCE_LIBRARY is used (or created).

Subcommand prompt: sc/

See also: Section 8-5

Similar commands: COS, NOS: UPDATE
 VMS: LIBRARY/TEXT

```

Examples:  /source_code_utility
           sc/user_library base=oldpl result=oldpl.$next
           sc/quit
           ^-- creates new cycle of OLDPL in
              your working catalog (assumed
              not to be $LOCAL)

           = = = = =
           /scu
           sc/create_deck deck=deck1 ..
           sc../modification=version1
           sc/quit
           ^-- initializes the working library
              from file SOURCE_LIBRARY in your
              working catalog

```

STATUS_CRAY

(DTRC) Display the status of Cray jobs.

```

Syntax:    STATUS_CRAY -or-
           STAC
           -----
           Q or QUEUE=list of key
              MF=name
           O or L or OUTPUT=file
              STATUS=status variable

```

```

Parameters: Q - the queue to be displayed
              q   queue
              -   -----
              A   all
              E   execution
              I   input
              O   output
              R   receiving
              S   sending
              (default: A)

              MF - the NOS mainframe
              (default: MCR)

```

See also: SUBMIT_CRAY_JOB

Similar commands: NOS: CSTATUS
VMS: CRAY STATUS

```

Examples:  /stac
           /stac e
           /stac ers

```

SUBMIT_CRAY_JOB

(DTRC) Submit a job to the Cray.

Syntax: SUBMIT_CRAY_JOB -or-
 SUBCJ
 FILE=file

 STATUS=status variable

Parameters: F - the file containing the Cray job

Remarks: The job is submitted through the NOS Cray station.

By default, the output will go to the NOS print queue. To route the output to a NOS/VE file, dispose \$OUT to a permanent file (see second example). The output may also be saved as an 8/12 ASCII NOS file.

See also: STATUS_CRAY

Similar commands: COS: SUBMIT
 NOS: CSUBMIT
 VMS: CRAY SUBMIT; CSUBMIT (DTRC);
 RCSUBMIT (DTRC)

Examples: /submit_cray_job my_cray_job_1
 ^-- output printed by NOS

 File MY_CRAY_JOB_2 contains:
 JOB,JN=jobname.
 ACCOUNT,UN=username,UPW=password,AB=jobborderno.
 DISPOSE,DN=\$out,DC=ST,DEFER,WAITTEXT=,^
 'USER,username,password.CTASK,NVE.\$outfile\$'.
 ...
 /subjc my_cray_job_2
 ^-- the Cray output is put into
 NOS/VE file <outfile>

SUBMIT_DETACHED_JOB

Start a disconnected interactive job.

Syntax: SUBMIT_DETACHED_JOB -or-
 SUBDJ

 UI or USER_INFORMATION=string
 STATUS=status variable

Parameters: UI - a 0- to 255-character string with information for the detached job (default: '' - a null string)

Remarks: Validation to submit detached jobs is required.

The detached job acquires the user name, login family, and job class from the submitting job.

SUBMIT_JOB

Submit a NOS/VE batch job.

Syntax: SUBMIT_JOB -or-
SUBJ

or FILE=file

```

-----
CTL   or CPU_TIME_LIMIT=keyword or integer
ERT   or EARLIEST_RUN_TIME=date_time
JAD   or JOB_ABORT_DISPOSITION=keyword
JC    or JOB_CLASS=name
JDBU  or JOB_DEFERRED_BY_USER=boolean
JD    or JOB_DESTINATION=name or string
JDU   or JOB_DESTINATION_USAGE=keyword or name
JER   or JOB_EXECUTION_RING=integer
JQ    or JOB_QUALIFIERS or
        JOB_QUALIFIER=keyword or list of name
JRD   or JOB_RECOVERY_DISPOSITION=keyword
LRT   or LATEST_RUN_TIME=date_time
LF    or LOGIN_FAMILY=name
MTL   or MAGNETIC_TAPE_LIMIT=keyword or integer
MAXWS or MAXIMUM_WORKING_SET=keyword or integer
OF    or OPERATOR_FAMILY=name
OU    or OPERATOR_USER=name
RHD   or REMOTE_HOST_DIRECTIVE=string
SL    or SRU_LIMIT=keyword or integer
S     or STATION=keyword or name
UI    or USER_INFORMATION=string
UJN   or USER_JOB_NAME or JN or JOB_NAME=name
SJN   or SYSTEM_JOB_NAME=name variable or
        list of name
        STATUS=status variable

```

Remarks: The job begins with LOGIN and ends with LOGOUT.

See also: JOB

Similar commands: NOS: SUBMIT; ROUTE
VMS: SUBMIT

Examples: /colt job_file
 ct? login login_user=abcd ..
 password=mypass
 ct? display_command_list all
 ct? logout
 ct? **
 ^-- create the job file
 /subj f=job_file ujn=my_job
 ^-- submit the job
 /disjs jn=my_job do=all
 Name not found : my_job
 ^-- indicates the job has completed

TABLEND (UTILITY subcommand) End the collection of utility command table definitions.

Syntax: TABLEND

TASK Delimit a sequence of SCL statements to be as a synchronous or an asynchronous task.

Syntax: TASK

```

-----
TN or TASK_NAME=name
R or RING=integer
DM or DEBUG_MODE=boolean
SM or SUBSTITUTION_MARK=keyword or string
STATUS=status variable

```

Parameters: TN - causes asynchronous execution and identifies it for future reference (default: it executes synchronously)

R - the ring number (1-13) (default: the current ring)

SM - (substitution marks cannot be used to generate the TASKEND statement)

TERMINATE_COMMAND

Terminate the program interrupted by a pause break (%1).

Syntax: TERMINATE_COMMAND -or-
 TERC

 STATUS=status variable

Subccmmand prompt: p/

Similar commands: NOS: %2
VMS: ^Y; ^C

Examples: Enter new password
xxxxxxxxxxxxxxxxxxxxx <-- pause break entered
Suspended - 1
p/terminate_command
Command terminated.
/
^-- you decided not to change your
password, so you entered a pause
break (%1) to interrupt it, then
typed TERC to terminate the
command

TERMINATE_JOB

Terminate queued or executing jobs.

Syntax: TERMINATE_JOB -or-
TERMINATE_JOBS -or-
TERJ -or-
TERMINATE_INPUT -or-
TERI
N or NAME or NAMES or JOB_NAME or
JOB_NAMES or JN=list of name

JS or JOB_STATUS or STATE or S=keyword or
list of keyword
ODI or OUTPUT_DISPOSITION=keyword
STATUS=status variable

Parameters: N - the user- or system-supplies names of the
jobs to be terminated

JS - states for restricting termination

ab	option	meaning
	ALL	all jobs
D	DEFERRED	jobs not yet eligible to be initiated
Q	QUEUED	jobs waiting to be initiated
I	INITIATED	jobs that have been initiated
T	TERMINATED	jobs that are terminated

ODI - how the output is to be discarded

ab	option
DSO	DISCARD_STANDARD_OUTPUT
P	PRINTER
WQ	WAIT_QUEUE

Remarks: WHEN conditions are not processed.

Similar commands: NOS: DROP
VMS: DELETE/ENTRY

TERMINATE_OUTPUT

Delete output files.

Syntax: TERMINATE_OUTPUT -or-
TERMINATE_OUTPUTS -or-
TERO -or-
TERMINATE_PRINT -or-
TERMINATE_PRINTS -or-
TERP
N or NAME or NAMES or JOB_NAME or
JOB_NAMES or JN=list of name

OS or OUTPUT_STATUS=keyword
STATUS=status variable

Parameters: N - the user- or system-supplies names of the
output files to be deleted

OS - states for restricting termination

ab	option	meaning
--	-----	-----
	ALL	all jobs
D	DEFERRED	jobs not yet eligible to be printed
Q	QUEUED	jobs waiting to be printed
I	INITIATED	jobs that have been printed
C	COMPLETED	jobs that have printed but are still in the output queue

Similar commands: NOS: DROP
VMS: DELETE/ENTRY

TERMINATE_TASK

Delete asynchronous tasks (and all tasks generated by those
tasks).

Syntax: TERMINATE_TASK -or-
TERT
TN or TASK_NAME or TASK_NAMES=keyword or
list of name

STATUS=status variable

Parameters: TN - the names of the tasks to be terminated
 ALL - all tasks initiated by the requesting task

TYPE Create one or more user-defined data types.

Syntax: TYPE
 name : type expression
 ...
 TYPEND

UTILITY

Delimit the definition and execution of a command utility.

Syntax: UTILITY
 N or NAME=name

 ESL or ENABLE_SUBCOMMAND_LOGGING=string
 IIP or INTERACTIVE_INCLUDE_PROCESSING=keyword
 or name
 L or LIBRARY or LIBRARIES=list of file
 LP or LINE_PREPROCESSOR=keyword or name
 OM or INLINE_MANUAL=keyword or name
 P or PROMPT=string
 SM or SEARCH_MODE=keyword
 T or TABLE or TABLES=file
 TCN or TERMINATION_COMMAND_NAME=name
 STATUS=status variable

Parameters: N - the name of the utility

ESL - YES - utility subcommands are logged
 NO - utility subcommands are not logged
 (default: YES)

IIP - reserved

L - the object library/ies of utility
 subcommand processors
 (default: the library containing the
 utility)

LP - reserved

OM - the name of the on-line manual describing
 the utility to be used as the default for
 the MANUAL parameter of the HELP command
 entered from within the utility
 NONE - no on-line manual
 (default: NONE)

- P - the prompt string
 ("/" or "../" is added, as appropriate)
 (default: the utility name)
- SM - the command search mode used for the utility
- | ab | option | meaning |
|----|------------|---|
| G | GLOBAL | all commands in the command list can be searched - commands specified by command name or file reference can be executed |
| R | RESTRICTED | like G, except searching beyond the first entry in the command list requires a slash (/) before the command |
| E | EXCLUSIVE | only the first entry in the command list is searched - commands specified by command name or file reference are not allowed |
- (default: GLOBAL)
- T - the file with the table of utility subcommands defined by the COMMAND command
 (default: %COMMAND)
- TCN - the utility subcommand used to terminate the utility
 (default: QUIT)

VAR Create one or more variables.

Syntax: VAR
 name: (attributes) type expression = initial
 value
 ...
 VAREND

Parameters: attributes - general (either or both):

- READ - variable cannot be modified
 (an initial value is required)
- DEFER - evaluate the expression
 when the variable is referenced

scope (only one):
 LOCAL, XDCL, XREF, ENVIRONMENT,
 JOB, TASK, UTILITY, PUSH
 (default: LOCAL)

```

init_value - (defaults: boolean: TRUE
                integer: 0
                list:      () (empty list)
                real:      0.0
                status:    NORMAL field=TRUE
                string:    ' ' (null string))

```

WAIT Suspend command processing for a time period or until an event.

Syntax: WAIT

```

-----
T   or TIME=integer or time_increment
TN  or TASK_NAME or TASK_NAMES=list of name
QN  or QUEUE_NAME or QUEUE_NAMES=list of name
U   or UNTIL=keyword
      STATUS=status variable

```

Parameters: T - number of milliseconds to wait
(default: no wait)

TN - the tasks to complete before resuming

QN - the job queues from which a message must be
received before resuming

U - how many of the events must occur
option meaning

```

-----
ANY    any of the specified events
ALL    all of the specified events
(default: ANY)

```

Similar commands: VMS: WAIT

Examples: /wait 20000 <-- wait 20,000 milliseconds (20
seconds)

WHEN Delimit SCL statements to be executed when a specified
condition occurs.

Syntax: WHEN condition names DO
statement list
WHENEND

SCL functions: OSV\$STATUS - initialized by the program
that determined the
condition

OSVSCOMMAND_NAME - the name of the command
being processed

See also: EXIT

Similar commands: VMS: ON

WHILE Conditionally repeat a statement list.

Syntax: label: WHILE
 boolean expression DO
 statement list
 WHILEEND label

Parameters: label - the name of the WHILE block - may be
 used by CYCLE or EXIT

 bool_exp - the terminating condition

Subcommand prompt: while/

Similar commands: NOS: WHILE, ENDW
 VMS: DO WHILE

WHO (DTRC) List the executing jobs, including users currently
logged in.

Syntax: WHO

Remarks: WHO displays the following information:

 CPU Idle: <idle>/<i/o>
 - CPU has been idle <idle>% of the time
 since the last display -- of that, <i/o>%
 was waiting for I/O to complete, and
 <idle>-<i/o>% was waiting for events such
 as timed or interactive input

 NOS: <exec>
 - CPU has been executing NOS <exec>% of the
 time since the last display

 For each job:

 SSN - the system-supplied name for the job
 UJN - the user-supplied job name (first 8
 characters
 C - the service class

class	meaning
B	batch
I	interactive
M	maintenance
S	system

S - the job status

status	meaning
blank	job is swapped out because it is in a long wait
D	job is swapped out because of its dispatching priority (higher priority jobs are CPU-bound)
F	job is swapped due to an operator request
I	job is being swapped into memory
L	job is swapped out due to an IDLE_SYSTEM command
M	job is in memory
P	job is swapped out due to low priority
T	job is swapped out because of thrashing
U	job is swapped out for other reasons

PR - the job scheduling priority (/100)

If the C, S, and PR fields are asterisks, the system has forced the job out due to a system or hardware failure. The job is dead and cannot be continued.

Similar commands: NOS: WHO
VMS: SHOW USERS

Examples: /who

```

          SSN          UJN   C   S   PR
CPU Idle: 43/20          NOS: 46

$0860_0488_AAA_0000 $SYSTEM S   M   0
$0860_0488_AAM_0010 MAILVE$S B
$0860_0488_AAL_4002 HPA_VES$P M
$0860_0488_AAM_0081 CACO      I
$0860_0488_AAM_0079 AMDS      I   M 170
          ^-- the list is in two columns
/

```


WHOAMI (DTRC) Display the executing user's username and job order number.

Syntax: WHOAMI

STATUS=status variable

Similar commands: NOS: BEGIN,WHOAMI
VMS: MYACCOUNT

Examples: /whoami
Login_Project : \$1222233344
Login_User : ABCD

XEROX (DTRC) Print a NOS/VE file on the Xerox 8700.

Syntax: XEROX

F or FILE=file

C or COPIES=integer

J or JOB=keyword

P or PAPER=keyword

D or DUPLEX=keyword

FO or FORMS=keyword

S or SHIFT=integer

U or UJN=name

STATUS=status variable

Parameters: F - the file to be printed on Xerox

C - the number of copies (1-999)
(default: 1)

J - job on Xerox to process the file
(in the following: CC=carriage control
character, L=landscape, P=portrait)

job	L/P	width	lines	description
STD LND	L	CC+136	62	std L job
STD PRT	P	CC+72	62	std P job
ST3PRT	P	CC+94	62	3rd std P job (very large margins)
ST4PRT	P	CC+87	62	4th std P job (general documentation; manuals)
DOC PRT	P	CC+87	62	ST4PRT with '.' in col 76 to force new side
VAXDOC	P	CC+87	66	like ST4PRT with blank CC and exactly 66 lines per page

\$2UPPRT	P	CC+136	124	2 std L arranged vertically
\$2UCPRT	P	CC+136	132	like \$2UPPRT but continuous (good for printer plots)
LN8LND	L	CC+136	84	8 lpi L
\$2U8PRT	P	CC+136	168	\$2UPPRT at 8 lpi
\$2ACLND	L	2*(CC+72)	62	2 std P arranged L

(Default: STDLND)

P - the kind of paper to be used

p	meaning
\$3_HOLE	3-hole punched
PLAIN	unpunched

(Default: \$3_HOLE)

D - specify one- or two-sided printing

d	meaning
DUPLEX (D, YES, YE, Y)	two-sided
SIMPLEX (S, NO, N)	one-sided

(Default: YES) (two-sided)

FO - specify the forms overlay to be used
(in the following: CC=carriage control
character, L=landscape, P=portrait)

fo	L/P	description
NONE		no special forms overlay
FRAME1	L	box around page - 1 solid line
FRAME2	L	box around page - 2 solid lines
FRAME3	L	box around page - 3 solid lines
FRAME4	L	box around page - 1 solid/ shaded line
\$2UPL1	L	solid line dividing 2 pages
\$2UPB1	L	solid boxes around 2 pages
\$2UPL2	L	double solid lines dividing 2 pages
FRAMP1	P	box around page - 1 solid line
FRAMP2	P	box around page - 2 solid lines
FRAMP3	P	box around page - 3 solid lines
FRAMP4	P	box around page - 1 solid/ shaded line
\$2UP3L	p	triple solid lines dividing 2 pages
\$2UPSL	P	single shaded line dividing 2 pages

\$2UPSB P shaded bar dividing 2 pages
 \$2UPB2 P 2 separate solid line boxes on
 a page
 WEEKLY P checklist by days

(Default: NONE)

S - specifies whether the file is to be shifted
 one column to the right before printing
 s meaning

 0 no shifting - the file already has
 Fortran carriage control characters in
 column 1 of each line
 1 the file does not have Fortran carriage
 control in column 1 and will be shifted
 one column to the right to produce a
 single-spaced listing

U - NOS jobname for the output
 (up to 7 alphanumeric characters, first 4
 should be User Initials)

(default: your user initials)

Remarks: XEROX passes the parameters to the BEGIN,XEROX
 command on NOS. It does this via a NOS/VE batch
 job which it creates.

You can monitor the jobs with DISPLAY_JOB_STATUS
 (DISJS). When it completes, the dayfiles will be
 in files XEROX_DAYFILE_VE and XEROX_DAYFILE_NOS,
 respectively. If you execute XEROX more than
 once, additional cycles of these files will be
 created.

Similar commands: NOS: BEGIN,XEROX (DTRC)
 VMS: XEROX (DTRC)

Examples: /xerox my_output j=st4prt ujn=amdsxyj

XMODEM_RECEIVE

Receive a file send from a PC using XMODEM.

Syntax: XMODEM_RECEIVE -or-
 XMOR
 FN or FILE_NAME=file
 FT or FILE_TYPE=keyword

 EC or ERROR_CHECKING=keyword
 FM or FILE_MARKERS=keyword
 TF or TRACE_FILE=file
 STATUS=status variable

Parameters: FN - the name of the NOS/VE file

FT - the type of file

ab	option	meaning
T	TEXT	file is transferred line by line (for standard text files)
BS	BYTE_STREAM	file is transferred character by character (for microbinary files)
NVE	NV or NOS_VE	file is transferred character by character, and the file label and length previously saved are used to restore the file attributes and exact file length - assumed file was originally sent to microcomputer using a NOS/VE file transfer (for object binary, binary, or NOS/VE backup files)

EC - error checking method to be used

ab	option	meaning
C	CHECKSUM	
CRC or CC	CCITT_CRC	
(default: the value of XMD\$XMOR_ERROR_CHECKING (if it exists) or CRC)		

FM - specify file markers for TEXT file transfers

option	meaning
NOS/VE	#EOR - end-of-partition #EOI - end-of-information
MSDOS	#EOR - end-of-partition ^Z - end-of-information
(default: the value of XMD\$XMOR_FILE_MARKERS (if it exists) or \$NULL)	

TF - the file to receive a trace showing a hexadecimal dump of every protocol block
(default: the value of XMD\$XMOR_TRACE_FILE (if it exists) or \$NULL)

Similar commands: NOS: XMODEM

XMODEM_SEND

Send a file to a PC using XMODEM.

Syntax:

XMODEM_SEND -or-

XMOS

FN or FILE_NAME=file

FT or FILE_TYPE=keyword

PS or PACKET_SIZE=keyword or integer

ELS or END_LINE_SEQUENCE=keyword

FM or FILE_MARKERS=keyword

TF or TRACE_FILE=file

STATUS=status variable

Parameters: FN - the name of the NOS/VE file

FT - the type of file

ab	option	meaning
-----	-----	-----
T	TEXT	file is transferred line by line (for standard text files)
BS	BYTE_STREAM	file is transferred character by character (for microbinary files)
NVE	NV or NOS_VE	file is transferred character by character, and the file label and length are included to preserve the file attributes and exact file length for later transfer back (for object binary, binary, or NOS/VE backup files)

PS - the packet data field size

ps	meaning
-----	-----

128

1024

SMALL same as 128

LARGE same as 1024

(default: the value of XMD\$XMOS_PACKET_SIZE (if it exists) or 128)

ELS - the character sequence signalling end-of-line for text files

option	meaning
-----	-----

CR carriage return

CRLF carriage return and line feed
(usually for MSDOS)

(default: the value of XMD\$XMOS_END_LINE_SEQUENCE (if it exists) or CR)

FM - specify file markers for TEXT file transfers

option	meaning

NOS/VE	#/EOR - end-of-partition

	#/EOI - end-of-information

MSDOS	#/EOR - end-of-partition

	^Z - end-of-information

(default: the value of XMD\$XMOS_FILE_ MARKERS (if it exists) or \$NULL)

TF - the file to receive a trace showing a hexadecimal dump of every protocol block

(default: the value of XMD\$XMOS_TRACE_ FILE (if it exists) or \$NULL)

Similar commands: NOS: XMODEM

YMODEM_RECEIVE

Receive a file send from a PC using XMODEM extention YMODEM.

YMODEM_SEND

Send a file to a PC using XMODEM extention YMODEM.

***** Appendix I *****

*** CDC NOS JCL Commands ***

CDC NOS job control language (JCL) commands have the following general syntax:

```
verb,param1,param2,...    comments
verb(param1,param2,...)   comments
```

verb is the name of the routine to be executed. It consists of an alphabetic character (A-Z) followed by 0-6 alphanumeric characters for the name of the command.

param1 are parameters, which may be positional or keyword.

comments follow the terminator (a period "." or right parenthesis ")").

*** Strings ***

The following string representations are used in this appendix:

```
aa...a  1 or more alphabetic characters
axx...x 1 or more alphanumeric characters, the first alphabetic
xxx...x 1 or more alphanumeric characters
nn...n  1 or more decimal (unless otherwise stated) digits
nn...nB 1 or more octal digits
nn...nD 1 or more decimal digits
```

*** Some Common Parameters ***

The following parameters are used in many JCL commands. If they have a different meaning or a special condition, it will be mentioned in the individual description.

CM=nnnnnn Maximum central memory field length, octal unless D suffix
or 8 or 9 in number, maximum 376500

CT=ct	File permit Category Type										
	<table border="0"> <tr> <td>ct</td> <td>meaning</td> </tr> <tr> <td>-----</td> <td>-----</td> </tr> <tr> <td>P or PR or PRIVATE</td> <td>private</td> </tr> <tr> <td>S or SPRIV</td> <td>semiprivate</td> </tr> <tr> <td>PU or PUBLIC</td> <td>public</td> </tr> </table>	ct	meaning	-----	-----	P or PR or PRIVATE	private	S or SPRIV	semiprivate	PU or PUBLIC	public
ct	meaning										
-----	-----										
P or PR or PRIVATE	private										
S or SPRIV	semiprivate										
PU or PUBLIC	public										

JSN=jsn Job Sequence Name (aaaa)

lfn Local File Name (xxxxxxx, 7 maximum)
(lfn's starting with ZZ are reserved to NOS)

lfn_in Input local file name
(normal default is INPUT)

lfn_out Output local file name
(normal default is OUTPUT)

L=lfn Output listing file (xxxxxxx, 7 maximum)
(normal default is OUTPUT)

M=m

m	File permissions (file access mode)																						
	<table border="0"> <tr> <td>m</td> <td>meaning</td> </tr> <tr> <td>-----</td> <td>-----</td> </tr> <tr> <td>E (EXECUTE)</td> <td>you can execute; others can read or execute concurrently</td> </tr> <tr> <td>R (READ)</td> <td>all can read or execute concurrently</td> </tr> <tr> <td>RU (READUP)</td> <td>all can read or execute; one (other) user can rewrite the file</td> </tr> <tr> <td>RA (READAP)</td> <td>all can read or execute; one (other) user can lengthen the file</td> </tr> <tr> <td>RM (READMD)</td> <td>all can read or execute; one (other) user can lengthen or rewrite the file</td> </tr> <tr> <td>U (UPDATE)</td> <td>all can read or execute; you can rewrite the file</td> </tr> <tr> <td>A (APPEND)</td> <td>all can read or execute; you can lengthen the file</td> </tr> <tr> <td>M (MODIFY)</td> <td>all can read or execute; you can lengthen or rewrite the file</td> </tr> <tr> <td>W (WRITE)</td> <td>you can read, execute, lengthen, rewrite, or shorten the file; others have no concurrent access</td> </tr> </table>	m	meaning	-----	-----	E (EXECUTE)	you can execute; others can read or execute concurrently	R (READ)	all can read or execute concurrently	RU (READUP)	all can read or execute; one (other) user can rewrite the file	RA (READAP)	all can read or execute; one (other) user can lengthen the file	RM (READMD)	all can read or execute; one (other) user can lengthen or rewrite the file	U (UPDATE)	all can read or execute; you can rewrite the file	A (APPEND)	all can read or execute; you can lengthen the file	M (MODIFY)	all can read or execute; you can lengthen or rewrite the file	W (WRITE)	you can read, execute, lengthen, rewrite, or shorten the file; others have no concurrent access
m	meaning																						
-----	-----																						
E (EXECUTE)	you can execute; others can read or execute concurrently																						
R (READ)	all can read or execute concurrently																						
RU (READUP)	all can read or execute; one (other) user can rewrite the file																						
RA (READAP)	all can read or execute; one (other) user can lengthen the file																						
RM (READMD)	all can read or execute; one (other) user can lengthen or rewrite the file																						
U (UPDATE)	all can read or execute; you can rewrite the file																						
A (APPEND)	all can read or execute; you can lengthen the file																						
M (MODIFY)	all can read or execute; you can lengthen or rewrite the file																						
W (WRITE)	you can read, execute, lengthen, rewrite, or shorten the file; others have no concurrent access																						

NA No abort on errors

non-null Indicates that the parameter the presence of a string of
 characters (normally xxxxxxx, 7 maximum)

pfn Permanent File Name (xxxxxxx, 7 maximum)
 (normal default is the lfn)

PW=password password for access to another user's permanent file

UJN=ujn User Job Name (xxxxxxx, 7 maximum)

UN=un User Name (xxxxxxx, 7 maximum)

*** Summary of CDC NOS JCL Commands ***

The following are NOS JCL statements, except as indicated by:

- (DTRC) A command, procedure or program added at DTRC
- (IAF) InterActive Facility
- (ICF) Interactive Cray Facility (ICF);
begin with a slash (/), not to be confused with the IAF prompt

JCL statements for certain NOS features are indicated by:

- (CRM) Cyber Record Manager
- (Loader) Loader control statements

* Entire line is a comment.

Syntax: *comment

See also: COMMENT

Similar commands: COS: *
VMS: !

Examples: * This is a comment ---

ctl (IAF) Interrupt the current job step (user-break-1).

Syntax: ctl

Parameters: ct - the network control character
(normally percent (%))

Remarks: Some terminals require that ^S be entered before
and ^Q after this command.

Examples: %1

ct2 (IAF) Terminate the current job step (user-break-2); cancel
the output in progress.

Syntax: ct2

Parameters: ct - the network control character
(normally percent (%))

Remarks: Some terminals require that ^S be entered before
and ^Q after this command.

Examples: %2

ctD (IAF) Immediately detach a terminal job from the terminal.

Syntax: ctD

Parameters: ct - the network control character
(normally percent (%))

Remarks: To detach during output, interrupt the output,
then enter ctD.

Any type-ahead commands are discarded.

Examples: %D

ctE (IAF) Immediate detailed job status.

Syntax: ctE

Parameters: ct - the network control character
(normally percent (%))

Examples: %E

ctS (IAF) Immediate abbreviated job status.

Syntax: ctS

Parameters: ct - the network control character
(normally percent (%))

Remarks: Response is one of: EXECUTE, IDLE (waiting for
you), or WAIT (waiting for system resources).

Examples: %S

ACCESS (IAF) Select the ACCESS subsystem.

Syntax: ACCESS

Remarks: Required to communicate with another interactive
terminal (DIAL, WHATJSN)

RUN will not work in the access subsystem.

Examples: ACCESS
WHATJSN
DIAL,jsn,message
BATCH

<-- reset yourself

APPEND Append information to the end of an indirect file without retrieving the file.

Syntax: APPEND,pfn,lfn_1,lfn_2,...,lfn_n/UN=un,PW=pw,NA,WB.

Parameters: UN=un - required only for files in another catalog

PW=pw - Required for files with passwords in another catalog

WB - wait for a busy file

Remarks: You cannot append to a direct file.

You cannot append to a direct access (random) file.

Similar commands: VMS: APPEND

Examples: APPEND,myperm,new1,nu2.

APPSW (IAF) Switch temporarily to an altername NAM application program.

Syntax: APPSW,AP=appl,Z.data

APPSW,appl,Z,data

Parameters: appl - a NAM application

appl	meaning
ICF	Interactive Cray Facility
RBF	Remote Batch Facility

ICF Interactive Cray Facility

RBF Remote Batch Facility

Z - any characters following the terminator are passed to the secondary application as data

data - first 50 characters after the terminator are passed to the secondary application

Examples: APPSW,ICF. <-- go into Cray Interactive Facility

ATTACH Assign a direct permanent file to a job.

Syntax: ATTACH, lfn_1=pfn_1, lfn_2=pfn_2, ..., lfn_n=pfn_n
/M=m, UN=un, PW=pw, NA, RT, WB.

Parameters: lfn_i=pfn_i - if lfn_i is omitted, pfn_i is used;
if lfn_i exists, it is discarded

M=m - (default: M=READ)

UN=un - required for a file in another catalog

PW=pw - required if UN is specified and the file
has a password

RT - real-time processing (job continues after
requesting staging from the MSS - a
second ATTACH is required to access the
staged file; if no staging is necessary,
the file is assigned immediately)

WB - wait for a busy file

See also: GET, FETCH (Cray)

Similar commands: COS: ACCESS; ACQUIRE
NOS/VE: ATTACH_FILE
VMS: no local file concept

Examples: ATTACH, MYFILE/M=W.

^-- allows file MYFILE to be
overwritten (such as after
editing)

AUX (DTRC) Turn an auxiliary printer on or off, send control codes
to control character size and to eject a page.

Syntax: BEGIN, AUX, , OPT.

Parameters: OPT - option:

On/off:

ON - turn the auxiliary printer on

OFF - turn the auxiliary printer off

Character size (Brother 2024L):

BCC - set to condensed characters

BCCOFF - turn off condensed characters

BEC - set to Elite

BPC - set to Pica

BWC - set to wide characters

BWCOFF - turn off wide characters
(also turned off by an LF)

Character size (Okidata MicroLine 82 or 84

OC132 - set Okidata MicroLine 82 or 84
printer to 132 characters

OC80 - set Okidata MicroLine 82 or 84
printer to 80 characters

Page eject:

TOP - go to the top of the next page

```

Examples:  BEGIN,AUXPRNT,,myfile,,X.
            ^-- print file MYFILE, then skip to
              new page
            BEGIN,AUXPRNT,,myfile,BEC.
            ^-- print file MYFILE in Elite on a
              Brother 2024L
            BEGIN,AUXPRNT,,myfile,OC132,TOP.
            ^-- print file myfile in condensed
              characters on an Okidata
              MicroLine 82, then eject a page

```

BASIC (IAF) Select the BASIC subsystem.

Remarks: Use FSE and X,BASIC.

BASIC See X,BASIC to compile a BASIC program without entering the BASIC subsystem.

BATCH (IAF) Select the BATCH subsystem.

Syntax: BATCH,fl

Parameters: fl - initial running field length for subsequent commands
(default: 0)

Remarks: This is the default subsystem when you enter IAF.

Examples: BATCH

BEGIN Transfer control to a procedure.

Syntax: BEGIN,pname,pfile,p1,p2,...,pn.comment (1)

-pname,pfile,p1,p2,...,pn.comment (2)

pname,p1,p2,...,pn.comment (3)

pfile,p1,p2,...,pn.comment (4)

Parameters: pname - the name of the procedure to be executed
(default: the procedure at the current position in the file)

pfile - the file containing the procedure
(default search order:
1) local file PROCFIL
2) your permanent file PROCFIL
3) public-access procedure file PROCFIL/UN=LIBRARY)

(note: if you have a local or permanent file PROCFIL, you cannot access any of the public procedures, therefore, use another filename for your procedures)

pi - an optional parameter

comment - value associated with the CK keyword in the procedure header

Remarks: Except for interactive execution of a procedure, the BEGIN statement may be continued on more than one line.

For interactive procedures, a question mark "?" may be used:

- . for a list of parameters:
BEGIN,pname,pfile,?
- . for help with a parameter:
BEGIN,pname,pfile,p1,...,pi?
- . for help during interactive processing:
param?

To accept the default value for a parameter during interactive processing, press the carriage return.

To accept the default value for the current and all remaining parameters, enter a terminator. If any required parameters remain undefined, you will be prompted for them.

See also: REVERT

Similar commands: COS: CALL
NOS/VE: filename
VMS: @filename

Examples: GET,MYPF,MYPROCFIL/UN=xxxx.
BEGIN,MYPROC,MYPF.
 ^-- a user procedure

BEGIN,NEWCHRG.
 ^-- a public-access procedure in
 PROCFIL/UN=LIBRARY

BELOAD Selectively load files from a NOS/BE tape created by DUMPF or BEGIN,SELDUMP.

Syntax: BELOAD,I=lfni,L=lfnl,T=lfnt,OP=op.

Parameters: I= - input file of directives
(default: INPUT)

T= - local file name of the tape
(default: TAPE)

OP= - load option for directives with PF=

op	meaning
--	-----

N	normal restore (don't replace existing file)

R	replace specified file if it already exists

(default: N)

Directives: ID=id, FN=fn, PF=nospf, UN=un, CY=cy, CT=ct, TY=ty,
PW=pw, M=m.

ID= - the NOS/BE ID (required)
 FN= - the NOS/BE filename
 (also requires PF=)
 PF= - the NOS filename (consisting of letters
 and digits only!)
 UN= - NOS username (required if the NOS/BE ID is
 not your user ID)
 CY= - the NOS/BE cycle number
 (default: the highest cycle only)
 TY= - the NOS file type
 ty meaning
 -- -----
 D direct
 I indirect
 (default: D)
 PW= - the NOS password for the file
 M= - the NOS access mode
 (default: R)

Remarks: BELOAD is in library BETONOS/UN=LIBRARY.

Similar commands: VMS: BACKUP

Examples: ATTACH, BETONOS/UN=LIBRARY.
 LIBRARY, BETONOS/A.
 LABEL, TAPE, VSN=NA9876, D=GE, F=S, LB=KL, R, PO=R.
 BELOAD, I=fylist.
 UNLOAD, TAPE.
 LIBRARY, BETONOS/D.

where fylist contains:

ID=abcd, PF=myfy11, FN=mynosbefy11.
 ID=abcd, PF=myfy12, FN=mynosbefy12, CY=423, TY=I.
 ID=efgh, PF=myfy13, UN=abcd, FN=mynosbefy13, CT=PU.

BKSP Backspace a file (by logical records).

Syntax: BKSP, lfn, n, m.

Parameters: lfn - the file to be backspaced

n - decimal number of logical records to
 backspace
 (Default: 1; max: 262143)

m - file mode: C (coded) or B (binary)
 (Default: B)

Similar commands: COS: SKIPF, SKIPR

Examples: BKSP, myfile, 4.

BLANK Blank label a magnetic tape.

Syntax: BLANK,VSN=vs_n,MT|NT,D=den|den,CV=cv,FA=fa,
OFA=ofa,VA=va,OWNER=username/familyname,
LSL=ls_l,U.

Parameters: VSN= - 1- to 6-character volume serial number
(don't use a current local file name or
the file will be lost)

MT - 7-track tape

NT - 9-track tape

D=den - tape density

MT		NT	
den	density	den	density
LO	200 cpi	HD	800 cpi
HI	556 cpi	PE	1600 cpi
HY	800 cpi	GE	6250 cpi
200	200 cpi	800	800 cpi
556	556 cpi	1600	1600 cpi
800	800 cpi	6250	6250 cpi

CV= - conversion mode for 9-track labels
(do not use with MT)

cv	meaning
AS	ASCII/6-bit display code
US	same as AS
EB	EBCDIC/6-bit display code

FA= - File accessibility character

fa	meaning
blank	unlimited access
A	only the owner can access it
other	future accesses must specify this character

OFA= - old file accessibility character when
relabelling a tape with one

VA= - volume accessibility character

va	meaning
blank	unrestricted access
other	only a system job can destroy the VOL1 label

(must always be a labelled tape)

OWNER= - ownership identification - determines
file accessibility (FA)

LSL= - label standard level entered in VOL1
label

ls1	meaning
1	tape labels and data format for this volume are ANSI std
blank	may or may not be ANSI

(Default: 1)

U - unload after blank labelling
(Default: do not unload)

Similar commands: COS: no tapes
VMS: MOUNT

Examples: BLANK,VSN=NA9999,D=GE.

BLOCK Add one or more lines of 10x10 block letters to a file.

Syntax: BLOCK,lfm,rewind,cc./line_1/line_2/.../line_n

Parameters: rewind - rewind option
R - rewind lfm before writing
NR - do not rewind lfm
(Default: NR)

cc - carriage control character to be
inserted before the first line
(Default: 1)

/ - a delimiter character which separates
the lines in the command - may be any
character and must immediately follow
the terminator - successive delimiters
generate blank lines

line_i - a string of up to 10 characters for one
line of blocked characters - or one of
the following:
DATE - current date
TIME - current time
USER - current user name
UJN - user job name
JSN - job sequence number

Similar commands: VMS: VSYS:BANNER, VSYS:BANNER6

Examples: BLOCK,blockf.*myjob*date <-- * is the delimiter

BYE (IAF) Terminate an application.

Syntax: BYE,appl

Parameters: appl - if IAF is your primary application, appl is one of: IAF, RBF (see APPSW)

if IAF is your secondary application, appl is one of:

appl	meaning
omitted	end IAF and return to primary application
ABORT	end both primary and secondary applications
(anything else will be treated the same as if omitted)	

See also: GOODBYE, LOGOUT; HELLO, LOGIN

Similar commands: COS: ^Z, QUIT
NOS/VE, VMS: LOGOUT

Examples: BYE

CATALOG List information about each record in a file.

Syntax: CATALOG,lfn,p_1,p_2,...,p_n.

Parameters: lfn - the file to be cataloged.

pi - parameters

N=n - catalog n files
N=0 - catalog until double EOF
N - catalog until EOI
omitted - same as N=1

L=name - the output file
omitted - same as L=OUTPUT

T - list entire text for records starting with APRD, CMRD, EQPD, IPRD, LIBD

U - list all records in a user library
omitted - list ULIB record

D - suppress comments field and all but first page heading

CS - suppress character set indicator
(63/64) for OPL and OPLC records

R - rewind lfn before and after

Remarks: Don't use on S, L or F tapes. COPY them to a disk file, or to an I or SI tape before CATALOGing.

For terminal output, set to NORMAL mode.

See also: ITEMIZE

Similar commands: COS: ITEMIZE

Examples: CATALOG,myfile,N,R.

CATLIST List permanent file information.

Syntax: CATLIST,LO=lo,FN=pfn,UN=un,NA,L=lfm,PW.

Parameters: LO=lo - list options

lo	meaning
F	all information about one or all files (3 lines per file)
FP	access permissions of a file
0	(zero) alphabetical list of names of indirect and direct files
P	list only names of users who have access to a private file or who have accessed a semiprivate file
X	LO=F plus security access categories for one file

(default: 0)

FN=pfn - a single file specification (required for LO=FP, LO=P, and LO=X) - one or more single-character wildcards (*) are allowed (e.g., ABC****)

L=lfm - file to receive the CATLIST output
(Default: L=OUTPUT)

PW - display passwords in LO=F output

Remarks: In the CATLIST output, a filename enclosed in parentheses means the file is on the MSS.

Similar commands: COS: AUDIT
NOS/VE: BACKUP_PERMANENT_FILES; DISPLAY_CATALOG; DISPLAY_CATALOG_ENTRY
VMS: DIRECTORY

Examples: CATLIST.
CATLIST,LO=F,FN=myfile.
CATLIST,FN=ABC****. <-- all files starting
with ABC

CDEFINE Display or define the defaults to be used by the Cray Station commands.

Syntax: CDEFINE.

Parameters: UN= - user name
(default: your user initials)

PW= - your 860 password

PN= - your charge number (called project number
in the display)

DC= - output disposition code
(default: DC=BC)

RUN= - RBF output user name
(default: your user initials)

QTF= - QTF access
(default: OFF)

Remarks: Other parameters should not be redefined because they are not used at DTRC (CN=) or have only one value at DTRC (MF=, FN=, RFN=).

You must use CDEFINE each time you change your Cray password.

Examples: /CDEFINE.
^-- displays the current defaults
CDEFINE> help
^-- displays help information
(uppercase if in ASCII mode)
CDEFINE> exit
^-- leave CDEFINE
(uppercase if in ASCII mode)

CDROP (ICF) Abort an executing Cray job saving the output.

Syntax: CDROP,jsq.

Parameters: jsq - Cray job sequence number

See also: CSTATUS

Similar commands: VMS Cray Station:

Examples: CDROP,ABCD

CHANGE Change some characteristics of a permanent file.

Syntax: CHANGE,nfn_1=ofn_1,nfn_2=ofn_2,...,nfn_n=ofn_n
/PW=pw,CT=ct,M=m,BR=br,PR=pr,SS=ss,NA,CE,
AC=ac,CP.

Parameters: nfn_i=ofn_i - change old file name to new file
name

PW=pw - new password
0 - clear the password

M=m - alternate user permission mode for
semiprivate and public files

BR=br - backup requirements

br	meaning
CR	off-station backup
Y	on-station backup
MD	backed up only if on disk
N	no backup

(MD and N are not recommended)

PR=pr - preferred residence

pr	meaning
M	alternate storage - MSS
N	no preference

SS=ss - new interactive subsystem
(BASIC, BATCH, EXECUTE, FORTRAN, FTNLS,
NULL)

CE - clear file error code

AC=ac - may alternate users obtain information
about the file? (Y or N)

CP - account number is to be replaced by
the account number currently in effect

Similar commands: COS: ALTACN; MODIFY
NOS/VE: CHANGE_CATALOG_ENTRY; CREATE_FILE_
PERMIT; DELETE_FILE_PERMIT
VMS: SET PROTECTION

Examples: CHANGE,mynew=myold.
 ^-- rename a file
CHANGE,myfile/CT=PU.
 ^-- make a file public
CHANGE,myfile/CP.
 ^-- change account number to current
 value

CHARGE Validate charging information for the job.

Syntax: CHARGE,account_number.

Parameters: account_number - your Job Order Number

Remarks: In a batch job, the initial CHARGE statement must immediately follow the USER statement following the job statement.

The CHARGE statement may also be used to change the Job Order Number for subsequent file saves.

Similar commands: COS: ACCOUNT
VMS: your home directory defines the
job order number

Examples: jobname.
USER,ABCD,batch_password.
CHARGE,1222233344.
 ^-- job charged to 1-2222-333-44
 ...
CHARGE,5666677788.
DEFINE,NEWFILE.
 ^-- file charged to 5-6666-777-88

CHVAL,CN Make your current account number your login default.

Syntax: CHVAL,CN.
CHVAL,CN=0.

Parameters: CN - means "Charge Number"
(use CN=0 to turn this feature off)

Remarks: After executing this command, you will no longer need to supply your charge number when you login.

This is especially useful if you have only one charge number. If you have more than one, you should probably avoid this since the charge number used at login is used for charging the entire session.

See also: CHARGE

Similar commands: NOS/VE: SET_DEFAULT_LOGIN_PROJECT (DTRC)

Examples: CHVAL,CN.
 = = = = =
CHARGE,1222233344.
 ^-- change current charge number to
 another (must be one of your
 registered numbers) for making
 permanent files
CHVAL,CN. <-- this is now your login default

CJOB (ICF) Get the status of a specific Cray job.

Syntax: CJOB,jname,jsq,L=lfn,RT=rt.

Parameters: jname - job name from JOB statement (uppercase)

jsq - optional job sequence number (CSTATUS)

L= - local file to receive the status
(default: OUTPUT)

RT= - repeat time (seconds)
(default: do not repeat the command)

See also: CSTATUS

Similar commands: VMS Cray Station: STATUS

Examples: CJOB,MYJOB.

CKILL (ICF) Delete an input job, kill an executing job saving only the logfile, delete an output dataset

Syntax: CKILL,jsq.

Parameters: jsq - job sequence number

See also: CSTATUS

Similar commands: VMS Cray Station: DTCF

Examples: CKILL,abcd.

CKP Take a checkpoint dump.

Syntax: CKP,lfn_1,lfn_2,...,lfn_n.

Parameters: lfn_i - a file to be included in the checkpoint
dump
(Default: all local files)

Examples: CKP.

CLEAR Release all (or all but one or more specified) files assigned to the job.

Syntax: CLEAR. <-- all files

CLEAR,*,lfn1,lfn2,...,lfnn. <-- all except those
named

Remarks: Checkpoint and no-auto-drop files are not released.

See also: RETURN, SETFS

Examples: CLEAR.
CLEAR,*,keepfyl.

COBOL5 Compile COBOL 74 program.

Syntax: COBOL5,B=b,I=i,L=l,LO=lo,PD=pd,SY.

Parameters: B=PUNCHB Produce punched binary decks of all routines

B=lfm Put binary into a file

B Same as B=BIN

B=0 No binary output

omitted Same as B=LGO

I=lfm FORTRAN source input is in lfm

I Same as I=COMPILE

omitted Same as I=INPUT

L=lfm Output lists to file lfm

L=0 Listings are suppressed

L Same as L=LIST

omitted Same as L=OUTPUT

LO=op/op/... Listing options (see L parameter)

op	meaning
M	address map
O	object code listing (use only if requested by Code 3511)
R	cross-reference map
S	source code list
LO	Same as LO=M/S/R
LO=0	No M, O, R, S information
omitted	Same as LO=S
PD=8	Print density (listings at 8 lines per inch, single spacing)
PD=6	listings at 6 lpi single spacing
PD=4	listings at 8 lpi double spacing
PD=3	listings at 6 lpi double spacing
PD	Same as PD=8
omitted	Same as PD=6
SY	Syntax check only; do not produce object code (cuts compilation time roughly in half)

Similar commands: NOS/VE, VMS: COBOL

Examples: COBOL5. Defaults to: I=INPUT,L=OUTPUT,LO=S,
B=LGO

COBOL5,I=INP,L=OUTP,LO,SY,PD=8.

COMMENT Place a comment in the system dayfile and your job's dayfile.

Syntax: COMMENT,jsn.comment
COMMENT.comment
*comment

Parameters: jsn - jsn of job to receive the comment
Default: the current job

comment - the message to be put into the dayfile

See also: NOTE (for on-line messages from a procedure)

Similar commands: COS: *
NOS/VE: DISPLAY_MESSAGE
VMS: !

Examples: COMMENT.MARK 1 <-- these are
*MARK 1 <-- the same

COPY Copy data from one file to another.

Syntax: COPY,I=lfni,O=lfno,V=x,M=c,TC=tc,N=copycnt,
BS=bsize,CC=charcnt,EL=erlimit,
PO=plp2...pn,L=lfnl,NS=ns.

COPY,lfni,lfno,x,c,tc,copycnt,bsize,charcnt,
erlimit,plp2...pn,lfnl,ns.

Parameters: I= - file to be copied
(default: INPUT)

O= - output file
(default: OUTPUT)

V= - non-null to rewind, copy, rewind, verify,
rewind both files (x must not be 0)
(default: no verify)

M= - coded files

c	meaning
C1	only input is coded
C2	only output is coded
x	both input and output are coded
	(x is non-null, except C1, C2)
	(default: binary)

C1 only input is coded

C2 only output is coded

x both input and output are coded
(x is non-null, except C1, C2)

(default: binary)

TC= - termination condition with N=
 tc meaning

F or EOF	N is number of files
I or EOI	N is ignored (copy to end-of-information)
D or EOD	N is number of double EOFs to copy to (N>1, TC=D, VERIFY verifies only to the first empty file)

(default: TC=D)

N= - copy count (meaning determined by TC=)
 (default: N=1)

BS= - maximum block size (in CM words) of S or L
 tape PRU (cannot be used with CC=)
 (defaults: S tape: 1000 octal;
 L tape: 2000 octal)

CC= - maximum number of characters in S or L tape
 block
 (default: not used; size from BS=)

EL= - number of non-fatal errors before abort;
 EL=U for unlimited error processing
 (default: 0)

PO= - processing options:

pi	meaning
E	copy input blocks with parity or block-too-long errors (default: error blocks skipped)
D	delete noise blocks in copy from disk, I- or SI-tape to S- or L-tape (defaults: binary padded with 00B to noise block size; coded padded with blanks)
R	allow record splitting in copy from disk, I- or SI-tape to S- or L-tape (default: splitting not allowed)
M	copy according to TC; do not write EOFs (default: write EOF after each file)

L= - alternate file for parity error messages
 for EL<>0; cannot be same as I=, 0=
 (default: OUTPUT)

NS= - noise size for S, L or F input tapes
 (maximum: 41; NS=0 uses default of 18)

See also: TCOPY, SCOPY

Similar commands: COS: COPYD; COPYF; COPYR; COPYU
NOS/VE: COPY_FILE; FILE_MANAGEMENT_UTILITY
VMS: COPY

Examples: COPY,a,b.
COPY,a,b,verify.
= = = = =
COPY,a,b,TC=F,N=nfiles.
 ^-- copy S or L tape

COPYBF Copy a multi-file file in binary mode.
COPYBR Copy records from one file to another in binary mode.

Syntax: COPYBF,lfn_in,lfn_out,nfiles,c.
COPYBR,lfn_in,lfn_out,nrecs,c.

Parameters: nfiles - decimal number of files to copy
nrecs - decimal number of records to copy
c - non-null indicates coded S or L tape

Defaults: COPYBF,INPUT,OUTPUT,1.
COPYBR,INPUT,OUTPUT,1.

Remarks: Not recommended for S or L tapes.
Copies from current position.
If lfn_in=lfn_out, the file is read.

See also: COPYCF, COPYCR, COPYEI

Similar commands: COS: COPYD; COPYF; COPYU

Examples: COPYBF,fyl1,fyl2,4.
COPYBR,fyl1,fyl2,125.

COPYBFR (DTRC) Restore the "randomness" of a random file which was copied using a sequential copy such as COPYBF.

Syntax: BEGIN,COPYBFR,,file1,file2.

Parameters: file1 - input file
file2 - output file

Remarks: This most often happened when a NOS/BE UPDATE random PL was copied to tape using COPYBF instead of either COPYBR or UPDATE,B. When the file is copied back to disk under NOS, there is an EOF at the end of the file, making it unusable as an UPDATE PL. COPYBFR will undo this.

Examples: COPYBFR,nosbefyl,nosfyl.

COPYCF Copy a coded multi-file file.

COPYCR Copy records from one coded file to another.

Syntax: COPYCF,lfn_in,lfn_out,nfiles,
fchar,lchar,na.

COPYCR,lfn_in,lfn_out,nrecs,
fchar,lchar,na.

Parameters: nfiles - decimal number of files to copy

nrecs - decimal number of records to copy

fchar - first character position of line to copy

lchar - last character position of line to copy

na - non-null to not abort if no line
terminator before EOR

When used with 6/12-bit files, FCHAR and LCHAR may give unexpected results. For 6/12-bit files, LCHAR should be at least twice the number of characters in the longest line.

Defaults: COPYCF,INPUT,OUTPUT,1,1,136.

COPYCR,INPUT,OUTPUT,1,1,136.

Remarks: Not recommended for S or L tapes.

Copies from current position.

If lfn_in=lfn_out, the file is read.

A coded file contains lines of 500 or fewer characters, terminated with a zero-byte (12-bits).

Lines longer than 500 6-bit characters are truncated.

For coded SI tapes, use TCOPY.

See also: COPY, COPYBF, COPYBR, COPYEI, SCOPY, TCOPY

Similar commands: COS: COPYD; COPYF; COPYU

Examples: COPYCF,cfyl1,cfyl2.
COPYCR,cfyl1,cfyl2,2,7,35,x.

COPYEI Copy one file to another.

Syntax: COPYEI,lfm_in,lfm_out,x,c.

Parameters: lfm_in - file to be copied

lfm_out - the copy of the input file

x - non-null to rewind, copy, rewind,
verify, rewind both files

c - non-null indicates coded S or L tapes
(default: binary)

Defaults: COPYEI,INPUT,OUTPUT.

Remarks: Not recommended for S or L tapes.

Copies from current position.

If lfm_in=lfm_out, the file is read to
end-of-information.

See also: COPY, COPYBF, COPYCF

Similar commands: COS: COPYD
NOS/VE: COPY_FILE
VMS: COPY

Examples: COPYEI,myin,myout.

COPYL Selective single replacement of object modules.

Syntax: COPYL,oldfyl,repfyl,newfyl,last,flag.

Parameters: oldfyl - old master binary file of object modules
(Default: OLD)

repfyl - replacement file of object modules
(Default: LGO)

newfyl - new master binary file of object modules
(Default: NEW)

last - name of last record in oldfyl to be processed
(Default: all records processed from current position to EOF or EOI)

flag - processing options

- R - rewind oldfyl and newfyl before processing
(repfyl is always rewound)
- A - append to newfyl all records in repfyl which do not match any of oldfyl
(Default: non-matching records ignored)
- T - match record name but not type
(Default: must match record name and type)
- E - process oldfyl through EOI
(may be selected in any combination: RA, ART, TEAR, etc., up to four characters; extra characters ignored)
(default: option not selected)

Defaults: COPYL,OLD,LGO,NEW.

Remarks: Oldfyl is processed forward only, but binary will be searched as often as needed.

See also: COPYLM

Similar commands: COS: BUILD
NOS/VE: FILE_MANAGEMENT_UTILITY
VMS: LIBRARIAN

Examples: COPYL,oldlgo,lgo,newlgo.

COPYLM Selective multiple replacement of object modules.

Syntax: COPYLM,oldfyl,binary,newfyl.
COPYLM,oldfyl,binary,newfyl,last,flag.

Parameters: see COPYL; flags T, E do not apply to COPYLM

Defaults: COPYLM,OLD,LGO,NEW.

Remarks: All occurrences of a module in oldfyl will be replaced by the first occurrence of a module with the same name in binary.

See also: COPYL

Similar commands: COS: BUILD
NOS/VE: FILE_MANAGEMENT_UTILITY
VMS: LIBRARIAN

Examples: COPYLM,oldlgo,lgo,newlgo,R.

COPYSBF Copy a file, shifting the lines one character to the right for printing on a printer.

Syntax: COPYSBF, lfn_in, lfn_out, nfiles, na.

Parameters: lfn_in - file to be copied

lfn_out - the copy of the input file

nfiles - decimal number of files to copy

na - non-null to not abort if no line terminator before EOR

Defaults: COPYSBF, INPUT, OUTPUT, 1.

Remarks: Not recommended for S or L tapes.

Copies from current position.

If lfn_in=lfn_out, the file is read.

A coded file contains lines of 500 or fewer characters, terminated with a zero-byte (12-bits).

Lines longer than 500 6-bit characters are truncated.

A page eject is inserted at the start of each logical record.

See also: COPYSF8

Similar commands: COS: COPYD
VMS: VSYS:CPYSF (DTRC)

Examples: COPYSBF, myprog.

COPYSF8 (DTRC) Copy an ASCII8 file, shifting the lines one character to the right for printing on a printer.

Syntax: COPYSF8, FROM, TO, NFILES, REWIND.

Parameters: FROM - ASCII8 file to be copied

TO - the copy of the input file

NFILES - decimal number of files to copy
(default: 1)

REWIND - rewind option
rew meaning

R - rewind input and output before
and after the copy
(synonyms: REWIND, Y, YES)

N - copy from the current position
and leave the files at EOI after
the copy
(synonym: NO)

(default: N)

Remarks: COPYSF8 uses FCOPY to convert from ASCII8 to ASCII;
COPYSBF to shift; FCOPY to convert back to ASCII8.

See also: COPYSBF

Examples: COPYSF8, myfile, shftfyl, , R.

COPYX Copy a file until a user-specified condition is met.

Syntax: COPYX, lfn_in, lfn_out, x, b, c.

Parameters: lfn_in - file to be copied

lfn_out - the copy of the input file

x - copy specifications

x	meaning
n	decimal number of records
00	copy through first zero-length record
name	copy through this record
type/name	copy through this record

b - backspace control

x	meaning
0	no backspace
1	backspace input file one record after copy
2	backspace output file one record after copy
3	backspace both files one record after copy

(ignored if EOF or EOI before x met)

c - non-null indicates coded S or L tape
(default: binary)

Defaults: COPYSBF, INPUT, OUTPUT, 1.

Remarks: Not recommended for S or L tapes.

Copies from current position.

If lfn_in=lfn_out, the file is read.

A coded file contains lines of 500 or fewer characters, terminated with a zero-byte (12-bits).

Lines longer than 500 6-bit characters are truncated.

A page eject is inserted at the start of each logical record.

See also: COPYBR, COPYCF, COPYCR

Similar commands: COS: COPYF; COPYR; COPYU

Examples: COPYX, fyl1, fyl2, 125.

CRERUN (ICF) Immediately end processing of specified job, delete output, and resubmit to input queue, if allowed.

Syntax: CRERUN, jsq.

Parameters: jsq - job sequence number

Remarks: A job cannot be rerun if:

- . a dataset has been adjusted, modified, saved, deleted, or written on
- . RERUN, DISABLE has been executed

See also: RERUN

Examples: CRERUN, 9876.

CSET (IAF) Change the terminal's character set mode.

Syntax: CSET,mode

Parameters: mode - one of:

ASCII - ASCII 128-character set

NORMAL - ASCII graphic 64-character set

Remarks: CSET may appear in a procedure.

See also: CSET,NORMAL does not affect AUTO or BRIEF mode.

Examples: CSET,ASCII

CSTATUS (ICF) Get the status of jobs, and input and output datasets.

Syntax: CSTATUS,queues,ST=start,L=lfm,RT=rt.

Parameters: queues - one or more of:

value	meaning
-----	-----
E or EXECUTION	execution queue
I or INPUT	input queue
O or OUTPUT	output queue
R or RECEIVING	Cray mainframe receiving queue
S or SENDING	Cray mainframe sending queue
A or ALL	all of the above
(Default: ALL)	

ST= - decimal number of entries to skip before
starting the display
(default: 0)

L= - local file to receive the status
(default: OUTPUT)

RT= - repeat time (seconds)
(default: do not repeat the command)

Remarks: If RT is specified, %2 may be needed to cancel
the output.

Examples: CSTATUS.

CSUBMIT (ICF) Submit a job to a Cray mainframe.

Syntax: CSUBMIT,lfn,RB=user,NO,TO.

Parameters: lfn - local file containing the Cray job
(display code or 8/12-bit ASCII)

RB= - (remote batch submission) user to receive
the output

RB - put the output into the print queue for you
(default: print at Central Site)

NO - drop output at job termination

TO - put the output in the wait queue

Remarks: Other parameters are available for running
another user's job.

The control statement record (\$CS) determines the
transfer mode for FETCH, DISPOSE, MSFETCH,

MSSTORE, etc., in the job. For example, if \$CS
is 8/12 ASCII, all coded files transferred
must/will be 8/12 ASCII.

Similar commands: VMS: CRAY SUBMIT
VMS Cray Station: SUBMIT

Examples: CSUBMIT,mycray.
 ^-- output to 860 printer

CSUBMIT,mycray,RB=un.
 ^-- output to un's output queue
 (see QGET)

CSUBMIT,mycray,RB.
 ^-- output to your output queue
 (see QGET)

CTASK Transfers the file between the Cray and the CDC NOS front-end.

Syntax: CTASK,ALL,code.

Parameters: ALL - include the dayfile from the CDC NOS job
with the Cray logfile
(default: the dayfile is not included)

code - internal code of the CDC NOS file
 ASCII8 - 8/12-bit ASCII
 DIS - display code
(default: display code, unless the whole
job is ASCII8)

Remarks: The text field of a Cray ACQUIRE, DISPOSE or FETCH statement can include NOS commands to fetch or store the file. CTASK causes the file transfer to occur.

Examples: See pages 2-1-8, 3-1-3.

CTIME Put the accumulated CPU time (in seconds) into the job's dayfile.

Syntax: CTIME.

See also: RTIME, STIME

Similar commands: VMS: ^T

Examples: CTIME.

DATE (DTRC) Put the date and time into the dayfile.

Syntax: BEGIN,DATE.

Remarks: A typical dayfile message is:
TODAY IS 90/09/11 AT 1425

Examples: BEGIN,DATE.

DAYFILE Write the job's dayfile (or a subset) to a file.

Syntax: DAYFILE,L=lfn,FR=string,OP=op,PD=pd,PL=pl,
I=infile.
DAYFILE,lfn,string,op,pd,pl,infile.

Parameters: L= - the file to which the dayfile is to be
written (on a new page if OUTPUT or if PD
or PL specified)
(Default: L=OUTPUT)

FR= - search string in field OP for starting the
copy. \$-delimited if any non-alphanumerics
in the string (time starts with a blank;
interactive commands start with a "\$"; for
example: \$\$\$OLD\$). If found, the dayfile
is copied from this point. If not found,
a message (connected or disconnected L) and
the entire dayfile (connected L) are
written.

OP= - search option

op	meaning
T	search time field
M	search message field
I	incremental dump (from point of last DAYFILE command)
F	full dump

(Defaults: OP=M (FR, but no OP);
OP=F (L disconnected);
OP=I (L connected))

PD= - print density

pd	meaning
3	double space; 6 lpi
4	double space; 8 lpi
6	single space; 6 lpi
8	single space; 8 lpi

PL= - page size

pd	page size	defaults
3	p1 / 2	30 lines
4	p1 / 2	30 lines
6	p1	60 lines
8	p1	60 lines

I= - file containing a dayfile for input
(Default: the active dayfile)

Remarks: A paginated dayfile cannot be used.

Similar commands: COS: ECHO
NOS/VE: DISPLAY_LOG
VMS: /LOG qualifier

Examples: DAYFILE,, \$ 11.21.\$,T.
^-- start with the last occurrence of
11.21. in the time field

DAYFILE,I=DAY,FR=\$\$\$GET,STATS.\$.
^-- start with the last occurrence of
\$GET,STATS. in the message field
of the dayfile in file DAY

DEFINE Create an empty direct permanent file.

Syntax: DEFINE, lfn_1=pfm_1, lfn_2=pfm_2, ..., lfn_n=pfm_n
/PW=pw, CT=ct, M=m, BR=br, PR=pr, S=space, NA,
AC=ac.

Parameters: See CHANGE.

PW=pw - a 1- to 7-character password required
by others for access

BR=br - backup requirements

br	meaning
CR	off-station backup
Y	on-station backup
MD	backed up only if on disk
N	no backup

(MD and N are not recommended)
(Default: Y)

CT=ct - (Default: CT=PRIVATE)

PR=pr - preferred file residence

pr	meaning
D	disk
L	locked to disk
M	alternate storage (MSS)
N	no preference
T	tape alternate storage

(Default: PR=N)

S=space - number of PRUs requested for the file
(default: the minimum number of blocks
(a multiple of 704 PRUs)
needed to hold the file)

See also: REPLACE; SAVE

Similar commands: COS: SAVE
NOS/VE: CREATE_FILE
VMS: CREATE

Examples: DEFINE,myfile/CT=PU.

DIAL (IAF) Send a one-line message to another user.

Syntax: DIAL,jsn,sss

Parameters: jsn - job sequence number of the receiving
terminal

sss - the one-line message

Remarks: You must be in the ACCESS subsystem.

No queueing takes place if jsn is busy.

See also: WHATJSN

Similar commands: VMS: PHONE

Examples: ACCESS
 WHATJSN
 DIAL,jsn,message
 BATCH <-- reset yourself

DISPLAY Evaluate an expression and put the result into the job's dayfile in octal and decimal.

Syntax: DISPLAY,exp1,exp2,...,expn.

Parameters: expi - any valid symbolic name or expression
 (value may be up to 10 digits)

See also: SET

Similar commands: COS: PRINT
 NOS/VE: CREATE_FILE_CONNECTION;
 DISPLAY_VALUE
 VMS: WRITE SY\$OUTPUT

Examples: DISPLAY,TIME.
 1253 2345B <-- if time is 12:53
 = = = = =
 SET,R1=99.
 DISPLAY,R1. <-- register R1
 99 143B
 = = = = =
 DISPLAY,143B. <-- octal number with B
 99 143B
 = = = = =
 DISPLAY,3/2. <-- integer division
 1 1B

DMB Binary dump of exchange package.

Syntax: DMB,ordinal.

Parameters: ordinal - an octal number (0-777777) used to create the dump record number (D plus ordinal) - ordinal > 377777 aborts the job after the dump
 (Default: 0 ==> D000000)

Remarks: The dump is written to file ZZZZDMB (an unconnected local file), which is never rewound.

See also: DMD, DMP

Similar commands: COS: DUMPJOB/DUMP

Examples: DMB.

DMD Dump the exchange package or central memory in both octal and display code.

Syntax: DMD,fwa,lwa. (1)
DMD,lwa. (2)
DMD. (3)

Parameters: fwa - first word address of memory to be dumped
(relative to RA)

lwa - last word address to be dumped
(relative to RA)

Format 1: dump a specified range of memory

Format 2: dump from RA+0 thru specified lwa

Format 3: dump the exchange package and 40B
locations before and after the program
address

Remarks: The dump is written four words per line to file
OUTPUT. Interactively, it is generally written
to file ZZZDUMP, which is never rewound.

See also: DMP

Similar commands: COS: DUMPJOB/DUMP
VMS:

Examples: DMD,50000,60000.

DMP Dump the exchange package or central memory in octal.

Syntax: Same as DMD.

Similar commands: COS: DUMPJOB/DUMP
VMS:

Examples: DMP,47000.

DROP Drop any of your executing or queued files (except the job
issuing the DROP command).

Syntax: DROP,JSN=jsn,DC=q,UJN=ujn,OP=R.
DROP,jsn,q,ujn,R.

Parameters: JSN= - either or both
UJN= - may be specified

DC= - disposition code

dc	meaning
WT	waiting jobs
PR	print jobs
PU	punch jobs
PL	plot jobs
IN	input jobs
EX	executing jobs
ALL	all your jobs

(Defaults: none (JSN=,UJN= omitted);
ALL (JSN= or UJN= specified))

OP= - drop executing jobs without EXIT, but with
single-reprieve processing

Similar commands: NOS/VE: TERMINATE_INPUT; TERMINATE_JOB;
TERMINATE_OUTPUT; TERMINATE_PRINT
VMS: STOP

Examples: DROP,ABCD. <-- drop executing job ABCD
DROP,JSN=ABCD,OP=R.
 ^-- drop executing job with
 single-reprieve but no EXIT
DROP,,PR. <-- drop all print jobs
DROP. <-- invalid (JSN, UJN or DC
 required)

ELSE Terminate skipping (false IF command with same label), or
initiate skipping (true IF command with same label) to ENDIF
with same label.

Syntax: ELSE,label.

Parameters: label - alphanumeric string (xxxxxxxxxx, 1-10
characters)

See also: IF

Similar commands: COS, NOS/VE, VMS: ELSE; ELSEIF

Examples: SET,R1=1.
 ...
 IF,R1=1,DOIT.
 <statements to do if true>
 ELSE,DOIT.
 <statements to do if false>
 ENDIF,DOIT.

ENDIF Terminate skipping by a SKIP, IF, or ELSE command with a matching label.

Syntax: ENDIF,label.

Parameters: label - alphanumeric string (1-10 characters, starting with a letter)

See also: ELSE, IF

Similar commands: COS, VMS: ENDIF
NOS/VE: IFEND

Examples: See IF.

ENDW The end of a WHILE loop.

Syntax: ENDW,label.

Parameters: label - alphanumeric string (1-10 characters, starting with a letter)

See also: WHILE

Similar commands: COS: ENDLOOP
NOS/VE: WHILEND

Examples: WHILE,R1<5,DOIT.

...
SET,R1=R1+1.
ENDW,DOIT.

ENQUIRE Get information about your jobs.

Syntax: ENQUIRE,OP=plp2...pn,FN=1fn1,0=1fn2.
ENQUIRE,plp2...pn.
ENQUIRE,JSN=jsn,0=1fn2.
ENQUIRE,UJN=ujn,0=1fn2.
ENQUIRE.

Parameters: OP= - Up to 7 of the following options:

pi	meaning
A	same as BDRUJLF
B	identification and priority info
D	resources demanded and assigned
F	status of files assigned to your job
J	contents of control registers, error flags fields, succeeding commands
L	loader info including status of CID
R	amount of resources used (CPU time, mass storage, perm file, and adder activity, SRUs used)

S accumulated SRUs
 T accumulated cpu time
 U initial amount of resources
 available (seconds, job step RSU,
 account block SRU, remaining
 resources available for dayfile
 messages, commands, and mass
 storage)

(Default: OP=A)

JSN= - returns detailed report on this job

UJN= - returns one-line report for each of your
 jobs

FN= - returns the status of local file lfn1

O= - writes the output to file lfn2

(Default: interactive: a 2-line report on the
 current job)

Similar commands: VMS: SHOW SYSTEM
 NOS/VE: DISPLAY_JOB_STATUS;
 DISPLAY_OUTPUT_STATUS

Examples: /ENQUIRE,B <-- display system activity
 /ENQUIRE,JSN <-- display all of your jsn's
 /ENQUIRE,JSN=abcd,O=out
 ^-- display status and remaining
 commands for job ABCD; write
 display into file OUT
 /ENQUIRE <-- a 2-line report on the
 current job
 /ENQUIRE,UJN <-- display status of your jobs

ENTER Enter a series of commands on one line.

Syntax: ENTER./command1/command2/.../commandn

Parameters: / - delimiter - any character not in any
 commandi - immediately follows the
 terminator

commandi - any NOS command (except interactive
 commands with no batch counterpart)

Remarks: The system supplies a terminator if it is missing
 from any commandi.

Examples: BATCH <-- enter the batch subsystem
 \$RFL,O. <-- displayed on entry to batch
 /ENTER.\get,fprog\ftn5,i=fprog\map,part\lgo\
 \exit\dmp\rewind,zzzdmp\copy,zzzdmp
 ^-- (must fit on one line)
 compile and execute a program

ERRMSG Control the display of error messages in a procedure.

Syntax: ERRMSG,status.

Parameters: status - OFF - turn off display of messages
ON - turn on display of messages
(Default: ON)

Remarks: ERRMSG has no effect in a batch job.

Examples: ERRMSG,OFF.

EVICT Release file space but not the file assignment.

Syntax: EVICT,lfn1,lfn2,....

Remarks: If the file is a magnetic tape or a read-only disk file, the file assignment is also released.

Similar commands: VMS: create a new version (disk)

Examples: EVICT,myfyl.

EXECUTE (IAF) Select the execute subsystem.

Remarks: Not recommended at DTRC.

EXECUTE (Loader) Complete loading, generate load map (if requested), begin execution; or execute at a specific entry point.

Syntax: EXECUTE.

EXECUTE,pname,plist.

Parameters: pname - a specific entry point at which to begin execution

plist - list of parameters

See also: LGO, name

Examples: LOAD,lgo.
EXECUTE.

EXIT Resume processing commands after a previous error.

Syntax: EXIT.

Remarks: When used in a procedure, precede it with a SKIP or REVERT, because EXIT terminates the current and all calling procedures without restoring the registers.

See also: NOEXIT, ONEXIT

Similar commands: COS: EXIT
VMS: ON condition

Examples: FTN5.
LGO.
EXIT.
COMMENT. Program failed.

EXPLAIN (IAF) Retrieve an on-line version of a CDC manual.

Syntax: EXPLAIN,M=manual
EXPLAIN,manual

Parameters: manual - the desired manual

See also: HELP, HELPME

Similar commands: VMS, VMS Cray Station: HELP
NOS/VE: EXPLAIN; HELP

Examples: EXPLAIN.

FCOPY Convert a file from one character set to another.

Syntax: FCOPY,P=1fn,N=1fn,PC=cs,NC=cs,PL=1t,NL=1t,
FL=f1,LB=1b,R,A.

Parameters: P= - input file to be converted
(default: OLD)

N= - output converted file
(default: NEW)

PC= - character set of input file
cs meaning

ASCII	6/12-bit display code supporting ASCII 63- or 64-char on current system
ASCII8	7-bit ASCII, rt-just in 12 bits
ASCII88	8-bit ASCII, rt-just in 8 bits
ASCII63	6/12-bit display code supporting ASCII 63-character set
ASCII64	6/12-bit display code supporting ASCII 64-character set
ASCFL	8-bit ASCII on S tapes (fixed line length, no line terminators)

DIS 6-bit display code supporting
 CDC 63- or 64-char on current
 system
 DIS63 6-bit display code supporting
 CDC 63-character set
 DIS64 6-bit display code supporting
 CDC 64-character set
 EBCFL 8-bit EBCDIC on S tapes (fixed
 line length, no line terminators)
 (default: ASCII)

NC= - output character set
 (default: ASCII8)

PL= - input line terminator

value	meaning
ZB	zero byte
CR	carriage return
FF	form feed
LF	line feed
US	unit separator
RS	record separator
CRLF	carriage return and line feed
LFCR	line feed and carriage return
n	specified octal character

(defaults: most: ZB;
 ASCII88: US;
 ASCFL, EBCFL: no terminators)

NL= - output line terminator
 (defaults: same as for PL=)

FL= - length of fixed length lines for S tapes
 (default: 80; valid only for ASCFL, EBCFL)

LB= - number of lines per block
 (default: 3840/fl; valid only for ASCFL,
 EBCFL)

R - rewind input and output files before and
 after processing
 (default: no rewind)

A - abort on errors
 (default: no abort)

Remarks: Maximum line length is 160 (12-bit codes) or
 320 (6-bit codes). Longer lines are truncated.

Files converted to 7-bit ASCII can be printed
 on a Central Site printer and on some remote
 batch printers. They cannot be listed at an
 interactive printer.

Similar commands: NOS/VE: GET_FILE

```
BEGIN,FICHE,,mydata,$TEST DATA 90/09/20$,,1.
      ^-- a data file shifted
```

FILE (CRM) Describe the attributes of a file.

Syntax: FILE,lfn,keys.
FILE,lfn=xxxxxxx,keys.

Parameters: lfn - file to be described
=xxxxxxx - new name for file
keys - keyword parameters for the various attributes and their values -- some are:

- ASCII= - for interactive terminals
 - 0 - 64-char display code
 - 1 - 95-char ASCII
 - 2 - 128-char ASCII
 - (default: ASCII=0)
- BFS= - buffer size
 - 0 - system provides space
 - n - octal buffer size
 - (default: BFS=0)
- BT= - block type
 - I - internal
 - C - character count
 - K - record count
 - E - exact record count
 - (default: BT=I)
- CF= - close file action
 - N - no rewind
 - R - rewind
 - U - rewind and unload
 - (default: CF=N)
- CM= - conversion mode
 - NO - no conversion
 - YES - convert external to internal
 - (default: CM=NO)
- CNF= - connect file flag
 - NO - disk or tape file
 - YES - terminal file
 - (default: CNF=NO)
- DFC= - dayfile control
 - 0 - fatal errors to dayfile
 - 1 - errors to dayfile
 - 2 - notes to dayfile
 - 3 - errors and notes
 - (default: DFC=0)
- EFC= - error file control
 - (same as DFC, except errors written to error file ZZZZEG)

EO= - parity error processing
 T - terminate with fatal error
 D - drop bad data
 A - accept bad data
 TD - same as T, D, A plus
 DD display the error
 AD block on error file
 ZZZZZEG
 (default: EO=T)

ERL= - trivial error count
 0 - no limit
 n - number of trivial errors to accept
 (max: 551)
 (default: ERL=0)

FF= - flush sequential files on abnormal termination
 NO - buffers not flushed
 YES - output scratch file buffers flushed
 (default: FF=NO)

FL= - fixed length (RT=T) or full length (RT=Z)
 0 - must be defined for open
 n - decimal length
 RT=F: n is 10-1310710
 RT=Z: n is 1-1310710
 (default: FL=0)

FO= - file organization
 SQ - sequential
 WA - word addressable
 (default: FO=SQ)

LCR= - label creation flag
 CRT - create new label
 CHK - check existing label
 (default: LCR=CRT)

LFN= - new local file name

LT= - label type
 UL - unlabelled
 S - ANSI standard
 NS - nonstandard
 ANY - any label type (no user processing)
 (default: LT=UL)

MBL= - maximum block length (in characters)
 0 - BT=I - 5120
 BT=C - 5120 (S tapes)
 BFS-20(L tapes)
 other - error
 n - length
 BT=K,E,RT=Z - $\geq$ FL+10
 BT=I - MBL
 (default: MBL=0)
 MRL= - maximum record length
 0 - no maximum
 n - maximum number of characters
 (max: 1310710)
 OF= - open file action
 N - no rewind
 R - rewind
 (default: OF=N)
 PD= - I/O processing
 INPUT - read
 OUTPUT - write
 IO - read/write
 (default: PD=INPUT)
 RB= - records per block for BT=K
 0 - same as RB=1
 n - number (max: 4095)
 (default: RB=1)
 RT= - record type
 W - control word
 F - fixed length
 R - record mark
 Z - zero-byte terminated
 D - decimal character count
 T - trailer count
 U - undefined
 S - system-logical-records
 (default: RT=W)

Remarks: Other parameters include: BBH, B8F, CL, CP, Cl,
 HL, LBL, LL, LP, MFN, MNB, MNF, MUL, OMIT, PC,
 PNO, REL, RMK, SB, SBF, SPR, TL, ULP, USE, VF.

Similar commands: NOS/VE: ATTACH_FILE; CHANGE_FILE_ATTRIBUTES;
 SET_FILE_ATTRIBUTES

Examples: FILE,PRT,BT=C,RT=Z,MRL=150.

^-- zero-byte terminated print file

FILE,STRANGR,BT=K,RT=F,RB=10,MRL=80,MBL=800,EO=A,
 ERL=25,BFS=512,CM=YES.

^-- stranger blocked coded tape

FILE,INPUT,LFN=DATA.

^-- substitute alternate input file

FLR (DTRC) Compile FTN5, load with up to 2 libraries, and run.

Syntax: BEGIN,FLR,,fyle,abs,lib1,lib2,norun.

Parameters: fyle - Fortran 77 source file

abs - to hold the absolute program
(default: ABS)

lib1 - optional first library for use in loading

lib2 - optional second library

norun - non-null to inhibit program execution

Similar commands: VMS: FLR

Examples: BEGIN,FLR,,mysorc,myabs.

^-- create absolute and execute

BEGIN,FLR,,mysorc,myabs,,,x.

^-- make absolute but don't execute

BEGIN,FLR,,mysorc,,dtlib.

^-- load using library DTLIB (you do
not have to ATTACH this first)

ATTACH,mylib.

BEGIN,FLR,,mysorc,,mylib,DTLIB.

^-- load using libraries MYLIB and
DTLIB in that order

FORM File Organization and Record Manager.

Syntax: FORM,I=dirfyl,B=owncode.

Parameters: I= - directive file
(default: INPUT)

B= - owncode routines to be loaded
(default: B=0)

Directives: Parameters for sequential files are shown
(additional parameters are available):

Input file: INP(lfn,POS=+-n,MAX=n,REW=r)

lfn - the input file

POS= - skip n logical records forward or
backward before processing
(range: 1-16383; default: no limit)

MAX= - maximum number of records to be processed
(range: 1-8,388,607; default: no limit)

REW= - rewind at end
 N - no rewind
 R - rewind
 U - rewind and unload (tape)
 (default: R; ignored for INPUT)

Output file:

OUT(lfn,POS=+-n,MAX=n,REW=r,BGD=g,SEL=a)

lfn - the output file
 (up to 20 OUT files may be specified)
 (default: OUTPUT)
POS= - same as for INP
MAX= - same as for INP
REW= - same as for INP
 (default: R; ignored for OUTPUT, PUNCH,
 PUNCHB)
BGD= - preset output record
 X - blank (55B)
 N - display code zero (33B)
 B - binary zero
 E - floating point zero
 C - same as input record
 (default: B)
SEL= - selection criteria
 ALL - copy all records with QAL
 processing as requested
 QRO - copy only records meeting QAL
 criteria
 (default: SEL=QRO)

Non-standard label: NON(lfn,ORD=n,LEN=n,LBL=lit)

Record qualification: QAL(lfn,condition)

Record reformatting: REF(lfn,entry,entry,...)

entry - out_iTm=in_iTm - move input field to
 output field

out_iTm=literal - put literal into
 output field of all
 records

iTm - field specification
 i - initial position (decimal)
 T - field type
 X - character
 (also: E, D, U, I, S, N, Z)
 m - length

literal - dollar-delimited (\$...\$)

Print: PRT(lfn,FMT=f,PGL=n,TOP=n,TTL=lit)

lfn - the output file
 FMT= - line spacing
 1 - single space
 2 - double space
 3 - triple space
 A - first character of line is carriage control
 D - dump the lines single spaced, 100-character lines
 PGL= - number of print lines (including the title line) per page
 (max: 60 (f=1), 30 (f=2), 20 (f=3); default: 60/f)
 TOP= - line of page for the title
 (range: 2-60; defaults: 1 (TTL omitted), 2 (TTL given))
 TTL= - the page title (up to 136 characters) (ignored for f=D)

IBM S/360 magnetic tape conversion:
 CON(lfn,RID=lit,descr...,RID=lit,descr...)

Execute: XEQ(ERR=e,COL=lit,...,FIN)

ERR= - error processing for unrecovered tape parity errors:
 ASV - abandon FORM run
 ANO - abandon FORM run; get rid of output disk files, OUTPUT and any partially written tapes remain
 CSV - continue, accept bad block, dump bad block to ZZZZZEF
 CNO - continue, delete bad block, dump bad block to ZZZZZEF
 COL= - alternate collating sequence (up to 64 characters; those not specified collate equal and higher than the highest specified)

Remarks: FORM has many functions including reblocking sequential files.

FORM may also be called from a Fortran or Cobol program.

FILE statements are used to describe the file blocking.

See also: FCOPY (much easier to use)

Similar commands: NOS/VE: FILE_MANAGEMENT_UTILITY

FORTTRAN (IAF) Select the FORTRAN subsystem.

Remarks: Use FSE and FTN5.

FSE Invoke the full-screen editor.

Syntax: FSE, FN=file, OP=access, I=input, L=output,
IP=procedure, WF=workfile.directives

FSE, file, access, input, output, procedure,
workfile.directives

Parameters: FN= - local or permanent file to be edited
(Default: most recently edited file during
job)

OP= - character set code and file location

access	abb	meaning
DISPLAY or NORMAL	D N	6-bit display code (default if your terminal is in NORMAL mode)
ASCII	A	6/12-bit display code (default if your terminal is in ASCII mode)
ASCII8	8	7-bit ASCII right-justified in 12 bits)
GET	G	access an existing file (may be used with above)

I= - input directive file
(Default: INPUT)

O= - output listing file
(Default: OUTPUT)

IP= - alternate FSE procedure library
(Default: FSEPROC)

WF= - alternate FSE work file
(Default: ZZZWORK)

dir - initial directives
(use ";" to separate directives)

See also: DTRC/CMLD-88/15, "CDC NOS Full Screen Editor
(FSE) User's Guide"

Similar commands: NOS/VE: EDIT FILE
VMS: EDIT/EDT; EDIT/TPU (EVE)

Examples: FSE,myfile,G. <-- GET/ATTACH existing file to
edit
FSE. <-- resume previous editing
session

FTN5 Compile Fortran 77 program.

Syntax: FTN5,ANSI=ansi,B=b,BL=bl,CS=cs,DB=db,DO=do,DS=ds,
E=e,EL=el,ET=et,GO=go,I=i,L=l,LO=lo,MD=md,
OPT=opt,PD=pd,PL=pl,PN=pn,PS=ps,PW=pw,QC=qc,
REW=rew,ROUND=round,SEQ=seq.

Parameters: ANSI=T Flag Non-ANSI (trivial)
ANSI=F Flag Non-ANSI (fatal)
ANSI Same as ANSI=T
ANSI=0 No ANSI diagnostics
omitted Same as ANSI=0

B=PUNCHB Produce punched binary decks of all
routines
B=lfm Put binary into a file
B Same as B=BIN
B=0 No binary output
omitted Same as B=LGO

BL Burstable list (each major compilation
section starts on a new page)
BL=0 Compact list (new page for first page
only)
omitted Same as BL=0

CS=FIXED Collating sequence fixed weight table
(display code)
CS=USER Weight table is user-defined by
subroutines COLSEQ, WTSET, CSOWN
CS Same as CS=USER (at DTRC)
omitted Same as CS=FIXED (at DTRC)

DB=op/op/... Debugging options
(ARG=FIXED not allowed)

op	meaning
--	-----
ER	generate code for object-time retrieval of errors
ID	generate output for interactive debug (requires OPT=0)
PMD	post mortem dump facility is used
SB	check subscript bounds
SL	check character substring expressions
ST	same as DB=ID but no stylized object code
TB	full error traceback

DB	Same as DB=ER/ID/PMD/SB/SL/ST/TB						
DB=0	All options deselected (DB=-ER/-ID/-PMD/-SB/-SL/-ST/-TB)						
omitted	Same as DB=0 (if OPT=1,2,3) Same as DB=ER (if OPT=0)						
DO=op/op	DO-loop interpretation						
	<table border="0"> <thead> <tr> <th>op</th> <th>meaning</th> </tr> </thead> <tbody> <tr> <td>LONG</td> <td>trip count may be > 131,071</td> </tr> <tr> <td>OT</td> <td>at least once through each DO loop</td> </tr> </tbody> </table>	op	meaning	LONG	trip count may be > 131,071	OT	at least once through each DO loop
op	meaning						
LONG	trip count may be > 131,071						
OT	at least once through each DO loop						
DO	Same as DO=OT						
DO=0	Trip count must be <= 131,071 and minimum trip count is 0						
omitted	Same as DO=0						
DS	All C\$ directives ignored						
DS=0	All C\$ directives processed						
omitted	Same as DS=0						
E=lfm	Output error list (see EL) on file lfm						
E	Same as E=ERRS						
omitted	Same as E=OUTPUT						
EL=C	List catastrophic errors						
EL=F	EL=C plus fatal errors						
EL=W	EL=F plus warning errors						
EL=T	EL=W plus trivial errors						
EL	Same as EL=F						
omitted	Same as EL=T						
ET=C	Abort job if catastrophic errors during compilation; next control statement to be executed is the one after EXIT(S); if no EXIT(S), job ends						
ET=F	Abort job if fatal or higher errors						
ET=W	Abort job if warning or higher errors						
ET=T	Abort job if trivial or higher errors						
ET	Same as ET=F						
ET=0	Continue even if compilation errors						
omitted	Same as ET=F (at DTRC)						
GO	Load and execute object code without a separate LG0 (B=0 and QC not allowed)						
GO=0	Do not load and execute						
omitted	Same as GO=0						
I=lfm	FORTTRAN source input is in lfm						
I	Same as I=COMPILE						
omitted	Same as I=INPUT						

L=lfm	Output lists (BL, LO) to file lfm (see also E option)														
L=0	Listings are suppressed														
L	Same as L=LIST														
omitted	Same as L=OUTPUT														
LO=op/op/... Listing options (see L parameter)															
	<table border="0"> <thead> <tr> <th>op</th> <th>meaning</th> </tr> <tr> <th>--</th> <th>-----</th> </tr> </thead> <tbody> <tr> <td>A</td> <td>list of variables, common blocks and attributes</td> </tr> <tr> <td>M</td> <td>address map</td> </tr> <tr> <td>O</td> <td>object code list (use only at request of User Services)</td> </tr> <tr> <td>R</td> <td>cross-reference map</td> </tr> <tr> <td>S</td> <td>source code list</td> </tr> </tbody> </table>	op	meaning	--	-----	A	list of variables, common blocks and attributes	M	address map	O	object code list (use only at request of User Services)	R	cross-reference map	S	source code list
op	meaning														
--	-----														
A	list of variables, common blocks and attributes														
M	address map														
O	object code list (use only at request of User Services)														
R	cross-reference map														
S	source code list														
LO	Same as LO=S/A/R														
LO=0	No A, M, O, R, S information														
omitted	Same as LO=S/A														
MD=T	Flag machine-dependent usage (trivial)														
MD=F	Flag machine-dependent usage (fatal)														
MD=0	No machine-dependent diagnostics														
omitted	Same as MD=0														
OPT=0	Fast compile (required by DB=ID)														
OPT=1	Intermediate optimization														
OPT=2	High optimization, slow compile														
OPT=3	OPT=2 plus potentially unsafe optimization														
OPT	Same as OPT=2														
omitted	Same as OPT=0														
PD=8	Print density (E, L listings at 8 lines per inch)														
PD=6	E, L listings at 6 lpi														
PD	Same as PD=8														
omitted	Same as PD=6														
PL=<n>	Print limit (decimal maximum number of records to be written at execution time on file OUTPUT) -- may be reset at execution time by specifying *PL=n, where n is the new line limit, at the end of the execute statement (max: 9 999 999 999)														
PL	Same as PL=50000														
omitted	Same as PL=5000														
PN	Page numbering of output list is continuous														
PN=0	Each subprogram starts with page 1														
omitted	Same as PN=0														

GET Get copies of indirect permanent files as local files.

Syntax: GET,lfn_1=pfn_1,lfn_2=pfn_2,...,lfn_n=pfn_n
 /UN=un,PW=pw,NA,RT.

See also: ATTACH, FETCH (Cray)

Similar commands: NOS/VE: ATTACH_FILE

Examples: GET,myindf1.
 GET,herindf/UN=her_un,PW=her_pw.

GETASC (DTRC) Get an indirect or direct 8/12 ASCII file and convert
 to 6/12 Display Code.

Syntax: BEGIN,GETASC,,FN=fn,O=lfn.

Parameters: FN= - name of 8/12-ASCII permanent (indirect or
 direct) file

 O= - name of 6/12-bit Display Code local file

Remarks: This is useful in getting ASCII files from the
 VAXcluster which were stored on the MSS using
 HFT STORE /ASCII.

See also: PUTASC

Similar commands: VMS: HFT FETCH /ASCII

Examples: HFT STORE /ASCII myvaxfile.for myvaxf
 BEGIN,GETASC,,myvaxf,myvaxf.
 ^-- local file MYVAXF will contain a
 6/12-bit Display copy of the
 8/12-bit ASCII file MYVAXF

GETATT (DTRC) Get an indirect file or attach a direct file.

Syntax: BEGIN,GETATT,,FN,UN,PW,LOG.

Parameters: FN - the name of the file

UN - required for file in another catalog

PW - required if UN is specified and the file has a password

LOG - controls the display of a message telling whether the file is indirect or direct

log	meaning

---	-----

Y	display the message (synonyms: YE, YES)

N	do not display the message (synonym: NO)

(default: N; only keyword: Y)

See also: GET, ATTACH

Examples: /BEGIN,GETATT,,myfile,,,y.
MYFILE is a direct file
/BEGIN,GETATT,,hisfile,abcd,hispw,log.
HISFILE/UN=ABCD is an indirect file
/BEGIN,GETATT,,herfile,efgh.
/

GO Clear the pause bit of one of your jobs.

Syntax: GO,jsn.

Remarks: The pause bit may be set by one of your programs or by the PAUSE command.

See also: PAUSE

Examples: GO,AAXJ.

GOODBYE Terminate an application.

Syntax: GOODBYE,application

Remarks: GOODBYE can appear in a procedure or a batch job, where it terminates the job.

See also: BYE, LOGOUT; HELLO, LOGIN

Similar commands: COS: ^Z,QUIT
VMS: LOGOUT

Examples: GOODBYE

GRIPE (DTRC) On-line gripe/suggestion facility.

Syntax: BEGIN,GRIPE,,OPT,I,L=OUTPUT.

Parameters: OPT - option
 S - short interactive prompt messages
 (default: longer prompting messages)
 I - optional input local file containing your
 prepared comments
 (default: you will be asked later for a
 filename or that you will enter
 them at the keyboard)
 L - after all the information has been gathered,
 this file will contain a copy of your
 comments
 (default: OUTPUT)

Remarks: This procedure executes a program which gathers the information and sends it to us. It may be executed interactively or in a batch job. When executed in batch, the data record must contain the following in the indicated order:

line	cols	contents
1	1-4	your User Initials
2	1-40	your name
3	1-10	your DTRC code
		or
	1-37	your non-DTRC company and address
4	1-20	your telephone number
		4 digits - DTRC/Annapolis extension
		5 digits - DTRC/Carderock extension
		7 digits - DC area (202) phone #
		10 digits - area exchange number
		If you include an extension, use 10
		digits followed by the extension
5	1-10	blank line - not a refund request
		non-blank - 10-digit job order
		number for refund
		request
6	1-2	Cray Station ID (C1, N1, V, V3-V4)
		or
	1-5	Site Code (MFN, VAX, DT3-DT4)

7 1-40 Supporting documentation (such as
 "PRINTOUT AND DUMP")
 (blank line if none)
 8 on 1-70 your comments
 (up to 20 lines -- excess truncated)

Similar commands: VMS: GRIPE

Examples: Batch:
 /JOB
 myjob.
 /USER
 /CHARGE
 BEGIN,GRIPE.
 /EOR
 ZMDS <-- User Initials
 JOE USER <-- name
 3511 <-- DTRC code
 71234 <-- Carderock extension
 <-- not a refund request
 N1 <-- gripe about CDC 860
 <-- no supporting info
 THIS IS WHERE THE DESCRIPTION
 OF THE PROBLEM GOES - LIMITED TO
 20 LINES

GTR Selective extraction of records from a (library) file.

Syntax: GTR,1fn_1,1fn_2,d,NR,S,NA.dir_1,dir_2,...,dir_n

Parameters: 1fn_1 - source file
 (Default: OLD)

1fn_2 - output file (must be disk if "d" used)
 (Default: LGO, positioned at EOI (disk)
 or BOI (tape, unless NR)

d - random access directory
 omitted - no new random access
 directory; if ULIB type,
 first record not copied,
 rest of records copied; last
 record of OPLD copied without
 alteration; no EOF written at
 end of file
 U - no new random access
 directory; if ULIB type,
 first record copied without
 alteration with rest of
 library and OLPD; no EOF
 written at end of file

D - write random access directory
on lfn_2 with entries for
selected records; if ULIB
type, first record copied
without alteration with rest
of library and OPLD; EOF
written after new directory

other - same as D

NR - do not rewind lfn_1 after; do not rewind
lfn_2 before or after; copy an existing
directory from lfn_1 to lfn_2

S - search lfn_1 sequentially

NA - do not abort on error

dir_i - a record or group of records to get:
type/name
name
type_1/name_1-type_2/name_2
type_1/name_1-name_2
name_1-name_2
type/name-* (all type from name on)
name-* (all from name on)
type/* (all of type)
* (all records)
0 (insert zero-length record)

See also: LIBEDIT

Similar commands: COS: BUILD
VMS: LIBRARIAN

Examples: GTR,SYSTEM,BIN,D.PP/*
^-- copy all PP records, build random
access directory

GTR,OPL,NEW,,NR.OPLC/COMCARG,0,COMCCIO
^-- copy 2 records and put a zero-
length record in between them at
the current position of NEW; NEW
not rewound, OLD rewound before

GTR,SYSTEM,SYSLIB,D.ULIB/SYSLIB
^-- user library SYSLIB copied from
file SYSTEM to end of SYSLIB

GTR.REL/A <-- copy record A from OLD to LGO

HELLO (IAF) Logs you out of IAF and switches you to another
application, or starts another login.

Syntax: HELLO,application

Remarks: If IAF is a secondary application, HELLO,appl is
the same as BYE,appl.

If application is omitted, a new login is started.

See also: LOGIN; BYE, GOODBYE, LOGOUT

Similar commands: VMS: LOGOUT then reconnect

Examples: HELLO,ICF <-- switch to Interactive Cray Facility

HELP (IAF) Ask for help.

Syntax: HELP.

Remarks: Displays a menu and prompts for your selection.

Help features:

- . list of all NOS commands, except compiler calls
- . help in entering a command
- . access to on-line CDC manuals
- . list of NAM/CCP network commands

See also: EXPLAIN, HELPBE, HELPME

Similar commands: NOS/VE, VMS: HELP

Examples: HELP.

HELP (DTRC) Obtain help on DTRC additions to NOS or other topics.

Syntax: BEGIN,HELP,,T,L.

Parameters: T - the name of the command, (sub)program, procedure, or other topic for which help is desired -- the following provide help about this procedure:

CONTENT - a list of the available topics
? - how to use this procedure

L - output file
(Default: L=OUTPUT)

Remarks: Some help modules are pages from CCRM and may be printed or BEGIN,AUXPRNT-ed and inserted into the manual.

Most help modules are 6/12 ASCII.

Similar commands: VMS: HELP

Synonym: IFE

Parameters: condition - an expression evaluating to true or false

command - a valid command

label - alphanumeric string (1-10 characters, starting with a letter)

Similar commands: COS, NOS/VE, VMS: IF

Examples: IF(R1=1,there)
 ...
 ENDIF(there)
 = = = = =
 IF,R1=1.REWIND,fyle.

ITEMIZE List information about each record of a binary file.

Syntax: ITEMIZE,lfn,L=listfyl,BL,PW=n,PD,NR,N=n,E,U.

Parameters: params - optional parameters:

- BL - burstable listing (new page for each file output)
- E - list entry points for relocatable modules; list IDENTs for UPDATE sequential PL
- L= - the output listing, if other than OUTPUT
- N - process to end-of-information
- N=n - number of files to process (default: 1)
- NR - lfn not rewound before or after
- PD - print density is 8 lpi (default: 6 lpi)
- PW=pw - page width (defaults: batch: 136 or 72 IAF: 72 (output file connected) 136 (output file not connected))
- U - itemize all records in a user library (default: only user library directory)

Remarks: Output includes record number, name, length, prefix table for relocatable binary or user library.

For a sequential UPDATE PL, only deck names are listed.

Your terminal should be in NORMAL mode (not ASCII) before listing ITEMIZE output at your terminal.

Defaults: ITEMIZE,LGO,L=OUTPUT,N=1,PW=see above.

See also: CATALOG

Similar commands: COS: ITEMIZE
 NOS/VE: ANALYZE_OBJECT_LIBRARY;
 DISPLAY_OBJECT_LIBRARY
 VMS: LIBRARIAN

Examples: ITEMIZE,oldpl.
 ITEMIZE,userlib,U,L=uout.

job Identifies requirements for a batch job.

Syntax: ujn,SC=sc,T=t,CM=cm,ST=lid.punchmode
 ujn,SCsc,Tt,CMcm,STlid.punchmode
 ujn,sc,t,cm,lid.punchmode

ujn,P=p,T=t,CM=cm,ST=lid.punchmode
 ujn,Pp,Tt,CMcm,STlid.punchmode
 ujn,p,t,cm,lid.punchmode

Parameters: sc - do not use at DTRC

p - priority (2, 3, 4)
 (higher priority means higher cost)
 (default: 2)

t - job step time limit in CPU seconds
 (Range: 1 to 32767 decimal;
 1 to 77777B octal)
 (Default: 64 decimal)

cm - maximum octal field length required
 (Maximum: 376500 octal)
 (Default: 376500 octal)

lid - not used at DTRC

punchmode - in columns 79-80 of actual punched cards
 26 - 026 mode
 29 - 029 mode

Similar commands: COS: JOB

Examples: ABCDjb1,CM75000,T10. <-- these three
 ABCDjb1,CM=75000,T=10. <-- are the
 ABCDjb1,,10,75000. <-- same
 = = = = =
 ABCD. <-- all defaults

JOB COST (DTRC) Display the job cost, SRUs used, and CPU time in the dayfile and file OUTPUT.

Syntax: BEGIN,JOBCOST.

Similar commands: COS: JOBCOST
VMS: ^T

Examples: BEGIN,JOBCOST.
MYPROG.
BEGIN,JOBCOST.

^-- the difference is the cost of running program MYPROG

KERMIT (DTRC) Transfer files to/from microcomputers.

Syntax: GET,KERMIT/UN=NSYS.
KERMIT.

Remarks: You must have KERMIT on your PC (it might be a subset of PROCOMM).

See also: FTP; XMODEM

Similar commands: NOS/VE: KERMIT
VMS: FTP; KERMIT

LABEL Mount a magnetic tape and, if labelled, check the label.

Syntax: LABEL, lfn, VSN=vsn_1/vsn_2=...=vsn_n-1/vsn_n,
D=den, F=format, LB=lb,
FC=fcount, CV=cv, NS=ns, PO=plp2...pn, CK|CB,
SI=setid|M=setid, SN=secno, QN=seqno,
L=fileid, FA=fa, G=genno, E=gvn,
CR=cdate|C=cdate, RT=yyddd|T=ddd, W|R,
AC=ac, CT=ct, PW=pw,
TO=to, UN=username.

Required parameter: lfn - local file name for the tape -
if lfn is already a local disk
file, processing continues - if lfn
is already a mounted tape and R is
present, the label is checked

Parameters: AC= - alternate auditability

CK - lfn is to be used as a checkpoint file
CB CK - append dump to lfn
CB - write dump at BOI of lfn

CR= - creation date (yyddd)
C=

CT= - file category

CV=cv - conversion mode for 9-track (NT) labels

cv	meaning
AS	ASCII/6-bit display code
US	same as AS
EB	EBCDIC/6-bit display code

D=den - tape density

MT		NT	
den	density	den	density
LO	200 cpi	HD	800 cpi
HI	556 cpi	PE	1600 cpi
HY	800 cpi	GE	6250 cpi
200	200 cpi	800	800 cpi
556	556 cpi	1600	1600 cpi
800	800 cpi	6250	6250 cpi

E= - 1- to 4-digit generation version number
(Default: 0)

F= - data format

format	meaning
I	internal
SI	system internal (NOS/BE tape)
S	stranger
L	long block stranger
F	foreign (unknown data format)

(Default: F=I)

FA= - File accessibility character

fa	meaning
blank	unlimited access
A	only the owner can access it
other	future accesses must specify this character

FC= - maximum block size in frames (required
if F=F is specified)

G= - 1- to 4-digit generation number
(Default: 1; 0 not allowed)

L= - 1- to -17-character file label
(Default: blank)

LB=lb labelled or unlabelled tape

lb	meaning
KL	ANSI-labelled
KU	unlabelled
NS	nonstandard-labelled

(assumes data starts immediately
after the first tape mark)

NS=ns - noise size (any block shorted than ns frames is discarded); not valid for F=I, F=SI; D=PE, or D=GE
 (Maximum: 31; NS=0 ==> the default;
 Defaults: 0 (PE,GE), 18 (others))

PO= - processing options

po	meaning
--	-----
R	Read the tape (ring OUT)
W	Write the tape (ring IN)
L	List only first and last error messages for each bad tape block

Options may be combined (e.g., WL).
 Several other options are available.

PW= - password

QN= - 1- to 4-digit file section number of the file within a multi-file set
 (Default: 1; use QN=9999 to append a new file to a multi-file set)

R - read the label and check specified fields
 W - write a new label, if the existing label has expired
 (Default: R; ignored for F=SI and QN>1)

RT= - retention/expiration date (yyddd)
 T= - retention number of days (0 to 999)

SI= - 1- to 6-character file set identifier (required for file positioning in multi-file set) -- usually use the tape's VSN. (see page 1-5-1)

SN= - 1- to 4-digit file section number of the volume within a multivolume file set
 (Default: 1)

TO= - TMS option

UN= - user name

VSN= - 1- to 6-character volume serial number
 / - separates multiple reels
 = - use first available VSN
 (Default: from separate VSN statement preceding LABEL)

See also: VSN

Similar commands: COS: ACCESS
 NOS/VE: CHANGE_TAPE_LABEL_ATTRIBUTES;
 REQUEST_MAGNETIC_TAPE
 VMS: REQUEST, MOUNT

Examples: LABEL,tbe,VSN=NA9999,PO=R,F=SI,LB=KU,D=GE.
 ^-- read a NOS/BE unlabelled tape
 at 6250 cpi

LDSET (Loader) Set option(s) for the current load.

Syntax: LDSET(opt1,opt2,...,optn)
 LDSET.

Parameters: opti - option in one of the following forms:

key
 key=param
 key=param1/param2/.../paramn
 options include:
 FILES,lfn1/lfn2/.../lfnn
 probably not needed

LIB=lib1/lib2/.../libn
 LIBEDIT libraries to be searched

MAP=p/lfn

p - one or more of:

N - no map
 S - error messages and loader statistics
 B - block list and list of unsatisfied externals
 E - entry point list without cross reference
 X - entry point list with cross reference

(default: SB)

lfn - the file to hold the map
 (default: OUTPUT)

PRESET=p

PRESETA=p

preset memory as specified:

p	octal preset value
NONE	no presetting
ZERO	0000 0000 0000 0000 0000
DEBUG	6000 0000 0004 0040 0000
NGINF	4000 0000 0000 0000 0000
ALTZERO	2525 2525 2525 2525 2525
	(alternating 0 and 1)
n	a 1-20 digit octal constant
	with optional +, - and
	terminal B

(PRESETA puts each location's address
 in the 17 low order bits)

(default: PRESET=ZERO)

Remarks: MAP=S provides statistics: program length,
routines present and missing.

LDSET without parameters will prompt you for them.

See also: page 5-6-5

Similar commands: COS: SEGLDR, PRESET=
NOS/VE, VMS: virtual memory

Examples: LDSET, PRESETA=DEBUG, MAP=S.
= = = = =
/LDSET
LDR>? LDSET, MAP=S, PRESETA=DEBUG.
LDR>? LDSET, LIB=DTLIB. <-- previously attached
LDR>? LOAD, LGO.
LDR>? NOGO, ABS. <-- create absolute
/ABS. <-- execute

LENGTH Gives the current status of one of your local files.

Syntax: LENGTH, lfn.

Remarks: The information includes is length (PRUS), type,
current status.

See also: ENQUIRE, FN=lfn.

Similar commands: COS: DS

Examples: LENGTH, mylfn.

LGO (Loader) Load and execute the default compiler binary output
file.

Syntax: LGO.
LGO, plist.

Parameters: plist - list of positional and/or keyword
parameters for the program being executed

See also: name

Similar commands: COS: name
NOS/VE: LGO

Examples: FTN5.
LGO.

LIBEDIT Create and maintain a library of programs, subprograms, procedures, or text.

Syntax: LIBEDIT,P=1fn,N=1fn,I=1fn,Z,B=1fn,L=1fn,
LO=options,V,C,D,U=record,NA,NI,NR,NX=n.

Parameters: P= - the old file
P=0 => create new file from replacement files

N= - the new file

I= - input directive file
I=0 => no directives

Z - directives immediately follow the command terminator (the first character after the terminator is the directive separator; overrides I=)

B= - replacement and insertion records
B=0 => no replacement file

L - summary of LIBEDIT run and any requested listings
L=0 => suppress output

LO= - list options

option	meaning
C	list directives
E	list errors
M	list modifications
N	list records written to new file
F	full listing (same as LO=CEMN)

V - verify new file against old file by calling VFYLIB (U overrides V)

C - copy new file over old file when done

D - do not abort on processing errors

U= - old file is required to be a user library; new file is made a user library by call to LIBGEN (overrides V) - the value <record> becomes the name of the user library directory record
U or omitted - same as U=ULIB

NA - do not abort on directive errors

NI - do not insert unreplaceable records at EOF of new file

NR - old and new files not rewound before or after

NX= - n=0 - include cross references in library directory of new user library
n<>0 - do not include cross references
(has meaning only for U or the *LIBGEN directive)

Directives: See Section 5-5.

Defaults: interactive: LIBEDIT,P=OLD,N=NEW,I=INPUT,B=LGO,
L=OUTPUT,LO=EM,NX=0.
all others: LIBEDIT,P=OLD,N=NEW,I=INPUT,B=LGO,
L=OUTPUT,LO=F,NX=0.

Remarks: Either parameter U or the *LIBGEN directive is required to make your new library a user library.

If the output is to tape and the tape may be processed at a later date by GTR or MODIFY, put the new file on disk and COPY it to tape. This will insure that the directories have disk PRU random addresses and not tape PRU random addresses.

See also: LIBGEN

Similar commands: COS: BUILD
NOS/VE: CREATE_OBJECT_LIBRARY
VMS: LIBRARIAN

Examples: LIBEDIT,P=mysubs,N=mylib.

LIBGEN Create a new user library of routines for use by the loader.

Syntax: LIBGEN,F=1fn,P=1fn,N=name,NX=n.

Parameters: F= - source file containing absolute (ABS), overlay (OVL), procedure (PROC), relocatable (REL), or capsule (CAP) records
(no library generated if none of these records is in F)
(default: F=LGO)

P= - will contain the new user library
(default: P=ULIB)

N= - name of the generated user library; entered in ULIB and OPLD records
(default: N=value of P)

NX= - n=0 - include cross references in library
 directory of new user library
 n<>0 - do not include cross references
 (default: NX=0)

See also: LIBEDIT

Similar commands: COS: BUILD
 NOS/VE: CREATE_OBJECT_LIBRARY
 VMS: LIBRARIAN

Examples: ATTACH,subs.
 FTN5,I=subs.
 LIBGEN,P=mylib.

LIBLOAD (Loader) Load modules from a library which contain the specified entry points.

Syntax: LIBLOAD,libname,eptnamel,eptname2,...,eptnamen.

Parameters: libname - the library containing the desired entry points

eptnamei - a specific entry point to be loaded
 (only one entry point for core image load)

See also: LOAD

Similar commands: VMS: LINK ...,library/LIB/INCLUDE=

Examples: LIBGEN,F=LGO,P=mysubs.

LIBRARY (Loader) Specify a set of global libraries to be searched for externals and programs and the order in which they are to be considered.

Syntax: LIBRARY,file1,file2,...,filen/directive.

Parameters: filei - system or user library
 (a maximum of 2 user libraries)

directive - specify if the files are to be added to, deleted from, or replace your global library set.

directive	meaning
A	add
D	delete
R	replace

(default: R)

Omit all parameters to clear your global library set.

Remarks: The order of search for externals is:
global (those on most recent LIBRARY)
local (those in LDSET, LIB= or in LDSET tables
in the loaded modules)
system (SYSLIB)

The order of search for programs is:
local files; global, local and system (NUCLEUS)
libraries

LIBRARY may not occur in a load sequence.

A no-auto-drop status is set for these files
while they are in the global set. See SETFS.

Similar commands: COS: LIBRARY
NOS/VE: CHANGE_COMMAND_SEARCH_MODE;
CREATE_COMMAND_LIST_ENTRY
VMS: LINK ...,library/LIB

Examples: LIBRARY,MYLIB. <-- global set has MYLIB
LIBRARY,YOURLIB/A. <-- global set has MYLIB
and YOURLIB
LIBRARY. <-- global set empty

LIMITS List your validation limits.

Syntax: LIMITS,L=1fn.

Examples: LIMITS.

LINE (IAF) Set your terminal for line mode.

Syntax: LINE.
LINE,TM=model.
LINE,model.

Parameters: See SCREEN.

Remarks: LINE may be included in a procedure.

LINE is the default setting unless SCREEN is
included in your LOGINPR file.

Affects FSE, HELPME, screen formatting, and the
display of NOS procedure parameters.

See also: SCREEN

Similar commands: NOS/VE: CHANGE_INTERACTION_STYLE

Examples: LINE.

LIST (IAF) List lines of a local file.

Syntax: LIST,F=lf_n

Parameters: F= - the local file to be listed
(default: the primary file)

Similar commands: VMS: VSYS:LISTN (DTRC); TYPE

Examples: LIST,F=MYFILE
 ^-- list local file MYFILE

LISTLB List labels of an ANSI-labelled multi-file tape.

Syntax: LISTLB,lf_n,SI=setid,QN=seqno,LO=ltype,L=out.

Parameters: SI= - 1- to 6-character file set identifier

QN= - 1- to 4-character file sequence identifier

LO= - label type(s) to be listed

A - all labels

R - required labels

O - optional labels

V - VOL_n labels

H - HDR_n labels

F - EOF_n labels

E - EOV_n labels

U - uv_l_n, uh_l_n, ut_l_n labels

(default: A)

Examples: LABEL,tape,SI=NA9876....
LISTLB,tape,SI=NA9876.

LISTLID List network configuration and host availability information.

Syntax: LISTLID,LID=lid,PID=pid,L=lf_n.
LISTLID,ST=lid,PID=pid,L=lf_n.

Parameters: LID= - a specific logical identifier
ST=

PID= - a specific physical identifier

Similar commands: VMS: SHOW NETWORK

Examples: LISTLID.

LOAD (Loader) List of object files whose contents are to be loaded.

Syntax: LOAD,lfnl,lfn2,...,lfnn.

Parameters: lfni - rewind (except INPUT) before loading
lfni/R - rewind before loading
lfni/NR - do not rewind before loading

See also: LIBLOAD, SLOAD

Similar commands: COS: SEGLDR BIN=dn1,dn2,...
VMS: LINK f1,f2,...

Examples: LOAD,LGO,BIN.

LOCK Prevent writing on a file.

Syntax: LOCK,lfnl,lfn2,...,lfnn.

Parameters: lfni - a local file

Remarks: Used to prevent writing on a local file.

See also: UNLOCK

Similar commands: COS:
VMS: OPEN(...,READONLY) in Fortran
program

Examples: ... <-- create a new file
LOCK,newfile.
^-- inhibit further writing on file
NEWFILE
... <-- other commands
UNLOCK,newfile.
^-- all writing on file NEWFILE

LOGIN (IAF) Terminate your current application and start another.

Syntax: LOGIN,application

Remarks: LOGIN may be used in a procedure or batch job,
where it terminates the job.

See also: HELLO; LOGOUT, BYE, GOODBYE

Similar commands: NOS/VE: LOGIN

Examples: LOGIN,ICF <-- switch to ICF

LOGOUT (IAF) Terminate an application.

Syntax: LOGOUT,application

Remarks: LOGOUT may be used in a procedure or batch job, where it terminates the job.

See also: BYE, GOODBYE; LOGIN, HELLO

Similar commands: COS: ^Z,QUIT
NOS/VE, VMS: LOGOUT

Examples: LOGOUT

L072 Reformat files.

Syntax: L072,p1,p2,...,pn.

Parameters: I=lfm - file with reformat parameters
I - same as I=INPUT
I=0 - no file of reformat parameters
omitted - same as I=0

S=lfm - input file to be reformatted
S - same as S=SCR
omitted - same as S=SCR

L=lfm - output reformatted file
L - same as L=OUTPUT
omitted - same as L=OUTPUT

T=x - type of file being reformatted
x meaning

M Modify source data
C COMPASS source data
B other source data

T - same as T=B
omitted - same as T=B

H=xxx - (truncation) length of output line
H - same as H=72
omitted - same as H=72
(max: 150; must be >= Nx+0x)
(see Remarks below)

LP - format for line printer

NR - do not rewind S file

Nx=y - number of characters to be moved
x - field number (1-6)
y - number of characters being moved
(see Remarks below)

Ix=y - input data field
 x - field number (1-6)
 y - starting column
 (see Remarks below)

Ox=y - output data field
 x - field number (1-6)
 y - starting column
 (see Remarks below)

IT - suppress query before each change
 omitted - query before each change
 (interactive jobs only)

Remarks: Restrictions on H, N, I, O:
 (Nx+Ix)>150 --> error for 1<=x<=6
 (Nx+Ox)>H --> error for 1<=x<=6
 H >150 --> error

Defaults for N, O, I:

type	N1	I1	O1	N2	I2	O2	N3	I3	O3
B	72	1	1	0	0	0	0	0	0
C	7	9	1	50	41	8	15	112	58
M	2	6	1	48	10	3	22	82	51

Ni, Ii, Oi=0 for 4<=i<=6.

Most useful in compressing compiler list output
 to fit into 72 columns

Similar commands: VMS: VSYS:CPYEXT (DTRC)

Examples: L072,S=myin,L=myout,I1=2,O1=1.
 ^-- restore a file that was shifted
 one column to the right, perhaps
 by COPYSBF

MAP (Loader) Specify the global default option for load maps.

Syntax: MAP.
 MAP,p.

Parameters: p - the desired load map

p	meaning
OFF	no map (same as LDSET,MAP=N)
PART	statistics and block map (same as LDSET,MAP=SB)
ON	PART plus entry point cross- reference (same as LDSET,MAP=SBX)
FULL	ON plus entry point map (same as LDSET,MAP=SBEX)

(default at DTRC: OFF)

Remarks: MAP without a parameter resets to the default.

MAP remains in effect until changed by another MAP statement. It may be overridden for the next load by using LDSET,MAP=.

The more map requested, the more CP time and memory is required to generate it.

See also: LDSET

Similar commands: COS: SEGLDR
VMS: LINK qualifiers

Examples: MAP(PART)

MERGE Merge files.

Syntax: FILE,lfnin1,....
FILE,lfnin2,....
...
FILE,lfnout,....
MERGE.p1,p2,...,pn
or
MERGE.p1 p2 ... pn

Positional: MERGE.from,to,key,dir,l,,e,e1,
dialog,end,,,ownf,ownf1,
ownmrl,,own1,own2,own3,
own4,own5,retain,seqa,seqn,
seqr,seqs,status,sum,,
verify,fastio.

Interactive: MERGE.DIALOG=YES

Directive file: MERGE.DIR=1fn
MERGE.params,DIR=1fn
MERGE.DIR=1fn,params
MERGE.params,DIR=1fn,..
.more_params

Parameters: See SORT5.

VERIFY - Verify that each input file is sorted
before merging them.

Remarks: Files to be merged must be presorted.

See SORT5 remarks.

See also: SORT5

Similar commands: COS, VMS: SORT
NOS/VE: MERGE

Examples: FILE,in1,BT=C,RT=Z,MRL=80.
FILE,in2,BT=C,RT=Z,MRL=80.
FILE,outfyl,BT=C,RT=Z,MRL=640.
MERGE.(in1,in2),outfyl
 ^-- merge two files
MERGE.FROM=(in1,in2),TO=outfyl
 ^-- same

MFL Reset maximum field length for subsequent job steps.

Syntax: MFL,CM=nnnnnn.
MFL,nnnnnn.

Remarks: MFL clears RFL and allows the system to determine the FL for each job step.

MFL cannot exceed the job statement CM or 376500 octal, whichever is lower.

See also: RFL.

Examples: MFL,200000.

MODE Mode error bypass should not be used at DTRC. An attempt to ignore Error Mode 1 may cause an Error Mode 0.

MODIFY Edit a Modify-formatted program library.

Remarks: Use UPDATE.

name (Loader) Load and execute binary program or procedure in local file "name".

Syntax: name.
name,plist.

Parameters: plist - list of positional and/or keyword parameters for the program or procedure being executed

See also: BEGIN, LGO

Similar commands: NOS/VE, COS: name
VMS: RUN; logical name

Examples: ATTACH,myprog.
myprog.

NEWCHRG (DTRC) Set the current charge number of some or all files.

Syntax: BEGIN,NEWCHRG,,fn.

Parameters: fn - optional filename (wildcards allowed)
(default: all files)

Remarks: Useful at the start of the Fiscal Year or any
other time you have to change a lot of files.

See also: CHANGE

Similar commands: COS: NEWCHRG

Examples: BEGIN,NEWCHRG.

^-- change all files

BEGIN,NEWCHRG,,a*****.

^-- change all files starting with
an "A"

NEWRTNS (DTRC) Display a brief description of new routines added at
DTRC.

Syntax: BEGIN,NEWRTNS,,OUTPUT.

Parameters: OUTPUT - output file
(default: OUTPUT)

See also: NEWS, OLDNEWS

Similar commands: NOS/VE: NEW_ROUTINES
VMS: NEWROUTINES

Examples: BEGIN,NEWRTNS.

^-- display on your terminal

BEGIN,NEWRTNS,,myout.

^-- put into file MYOUT

NEWS (DTRC) Display the current news items.

Syntax: BEGIN,NEWS,,OUTPUT.

Parameters: OUTPUT - output file
(default: OUTPUT)

See also: OLDNEWS, NEWRTNS

Similar commands: NOS/VE, VMS: NEWS

Examples: BEGIN,NEWS.

^-- display on your terminal

```
BEGIN,NEWS,,myout.  
      ^-- put into file MYOUT
```

NOEXIT Continue processing with the next command even if an error has occurred (suppress EXIT processing).

Syntax: NOEXIT.

See also: EXIT, ONEXIT

Similar commands: VMS: ON condition

Examples: NOEXIT. <-- Exit processing off
FTN5.
ONEXIT. <-- restore exit processing
LGO. <-- executed even if compile errors
... <-- not executed if execution errors

NOGO (Loader) Complete loading of a program, generate load map (if requested), put absolute into a file (if requested), but do not execute.

Syntax: NOGO.
NOGO,abs.

Parameters: abs - will contain the loaded program as a single core image module (non-segmented/non-overlay loads only)
(<abs> is suitable for inclusion in a LIBEDIT library)

See also: LDSET

Similar commands: COS: SEGLDR (ABS= directive)
VMS: LINK

Examples: DEFINE,myprog/NA.
LOAD,LGO.
NOGO,myprog.

NORERUN Clear the job rerun status.

Syntax: NORERUN.

Remarks: May be useful to prevent updating a file when the job would ordinarily be rerun.

See also: RERUN

Similar commands: COS: RERUN

Examples: NORERUN.

NORMAL (IAF) Reverse the effect of ASCII, AUTO, BRIEF, CSET, ASCII, and NOSORT commands.

Syntax: NORMAL

See also: ASCII, AUTO, BRIEF, CSET, NOSORT

Examples: NORMAL

NOTE Create a file with the command line containing the lines for the new file.

Syntax: NOTE, lfn, NR./line_1/line_2/.../line_n

Parameters: lfn - (default: OUTPUT)

NR - Do not rewind lfn before and after
(default: rewind)

/ - a delimiter (any character) denoting the
start of a new line for the file
(the character immediately following the
terminator is the delimiter)

line_i - the contents of the i-th line of the
new file

Similar commands: COS: NOTE
NOS/VE: COLLECT_TEXT; PUT_LINE
VMS: OPEN, WRITE, CLOSE

Examples: NOTE, DATA./ 1 2.4/LINE OF TEXT/0.1 1E-4/END
^-- create a new file DATA

Local file DATA contains:

1 2.4
LINE OF TEXT
0.1 1E-4
END

= = = = =

NOTE, DATA, NR./ 1 2.4/LINE OF TEXT/0.1 1E-4/END
NOTE, DATA, NR./ 2 3.6/ANOTHER LINE OF TEXT
NOTE, DATA, NR./0.1 1E-4/END

^-- create file with many lines

PACK, DATA. <-- remove embedded EORS

Local file DATA contains:

1 2.4
LINE OF TEXT
0.1 1E-4
END
2 3.6
ANOTHER LINE OF TEXT
0.1 1E-4
END

= = = = =


```
NOTE,UIN./*compile prog1,sub1,sub2
UPDATE,I=UIN.
=====
NOTE,,NR./THE PROGRAM FINISHED
      ^-- useful for displaying messages
          (comments) from a procedure
```

NULL (IAF) Select the NULL subsystem.

Syntax: NULL

Remarks: This is the default subsystem in a batch job.

RUN will not work in the NULL subsystem.

Examples: NULL

OFFSW Clear sense switches.

Syntax: OFFSW,switch_1,switch_2,...,switch_n,jsn.

Parameters: switch_i - a sense switch to be cleared (1-6)
 0 - clear all sense switches

jsn - since a jsn is recognized by its
 alphabetic characters, jsn may appear
 anywhere in the parameters list

See also: ONSW

Similar commands: COS: SWITCH

Examples: OFFSW,0,ABCD.
 ^-- clear all sense switches for job
 ABCD

OFLREQ (DTRC) Generate an Off-Line Request to process tapes for the Calcomp, Xerox or Microfiche.

Syntax: BEGIN,OFLREQ,,type.

Parameters: type - type of request
 936 - Calcomp 936 plots
 1051 - Calcomp 1051 plots
 CALCOMP - same as 936
 COM - microfiche
 FICHE - same as COM
 XEROX - Xerox 8700

Remarks: This is for use with Calcomp output and for microfiche and Xerox output which cannot be handled by the standard BEGIN,FICHE and BEGIN,XEROX.

The procedure will execute a program which will prompt you for the information needed to prepare the Off-Line Request. When complete, the Request will be printed at Central Site for the Off-Line operator. A copy will be put into your local file FORM.

When running OFLREQ in a batch job, the data must be in local file KEYBRD. Any messages from the program will be in file CONSOL.

See also: BEGIN,FICHE, BEGIN,XEROX

Examples: BEGIN,OFLREQ,,1051.

OLDNEWS (DTRC) Display the old news items.

Syntax: BEGIN,OLDNEWS,,OUTPUT.

Parameters: OUTPUT - output file
(default: OUTPUT)

See also: NEWS, NEWRTNS

Similar commands: NOS/VE: OLD_NEWS
VMS: OLDNEWS

Examples: BEGIN,OLDNEWS.
 ^-- display on your terminal

BEGIN,OLDNEWS,,myout.
 ^-- put into file MYOUT

ONEXIT Reverse the effect of NOEXIT.

Syntax: ONEXIT.

See also: EXIT, NOEXIT

Similar commands: VMS: NOON; ON condition THEN CONTINUE

Examples: See NOEXIT.

ONSW Set sense switches.

Syntax: ONSW,switch_1,switch_2,...,switch_n,jsn.

Parameters: switch_i - a sense switch to be set (1-6)
 0 - set all sense switches

jsn - since a jsn is recognized by its alphabetic characters, jsn may appear anywhere in the parameters list

See also: OFFSW, SWITCH

Similar commands: COS: SWITCH

Examples: ONSW,ABCD,4,5.

^-- turn sense switches 4 and 5 on
in job ABCD

OPLEDIT Remove modification decks and identifiers from a MODIFY library.

Remarks: Use UPDATE instead of MODIFY.

OUT Send deferred output files to the print or punch queue immediately.

Syntax: OUT. <-- queue all files
OUT,*,lfn1,lfn2,...,lfnn. <-- queue all files,
except those listed

Remarks: OUT processes any file given deferred ROUTE-ing as well as OUTPUT, PUNCH, PUNCHB, P8.

See also: ROUTE

Similar commands: COS: DISPOSE

Examples: OUT.

OVWRITE Overwrite files to erase (destroy) their contents.

Syntax: OVWRITE,lfn1,lfn2,...,lfnn/OP=plp2.
^-- overwrite specified files
OVWRITE,*,lfn1,lfn2,...,lfnn/OP=plp2.
^-- overwrite all but specified files

Parameters: OP= - how files are to be overwritten and whether they are to be released

pi	meaning
Z	overwrite with zeros
X	overwrite with zeros, then ones, then alternating zeros and ones
R	release files after overwriting

(default: OP=Z)

Similar commands: COS: SCRUBDS; WRITEDS

Examples: OVWRITE,fyll,OP=XR.
 ^-- clear a file, then release it

PACK Combine all records/files in a file by removing all EORs and EOFs.

Syntax: PACK,lfn_in,lfn_out,x.

Parameters: lfn_in - (not rewound after)
 lfn_out - (default: lfnout=lfnin;
 rewound after, but not before)
 x - non-null to not rewind lfn_in before
 packing

Remarks: Do not use with S, L, or F tapes.

Examples: PACK,infyl,pkdfyl.

PASSWOR Change your password.

Syntax: PASSWOR,oldpw,newpw .
 PASSWOR.

Parameters: oldpw - old password
 newpw - new password (4-7 alphanumeric characters
 with at least one digit)

Remarks: At DTRC, there is only one password for both
 batch and interactive.

Similar commands: COS: ACCOUNT,NUPW=nupw,...;
 CNEWPW; CSUBMIT/NUPW (from VMS)
 (DTRC)
 NOS/VE: CHANGE_LOGIN_PASSWORD; SET_PASSWORD
 VMS: SET PASSWORD

Examples: PASSWOR,old,new.

PAUSE Set the pause bit of one of your executing jobs.

Syntax: PAUSE,jsn.

See also: GO

Examples: PAUSE,ABCD.

PDU (Panel Definition Utility) Compile a panel definition and store the compiled panel in a user library.

Syntax: PDU,I=panel,L=listing,C=capsule,LIE=library.

PDU,panel,listing,capsule,library.

PDU? <-- prompt for the parameters

Parameters: panel - the local panel definition file
(6/12 display code; must be same as the panel name)

listing - the output listing
(default: OUTPUT)

L=0 - inhibits the listing with the message
PANEL-CAN'T OPEN FILE 0

capsule - the name of the generated capsule file
C=0 - syntax check only

library - the name of the (local) panel library
(default: PANELIB)

LIB=0 - do not change a library file

Remarks: Panels are part of Screen Formatting. See CDC publication: 60460430 - NOS Version 2 Screen Formatting Reference Manual.

See also: SHOW, ULIB

Similar commands: VMS: FMS

Examples: In local file TEST:

```
{ PANEL TEST PRIMARY
  VAR RSIDE1 T=REAL F=E R=(0. 999999999.)
    HELP='Enter positive integer or real value'
  VAR RSIDE2 T=REAL F=E R=(0. 999999999.)
    HELP='Enter positive integer or real value'
  VAR RSIDE3 T=REAL F=E R=(0. 999999999.)
    HELP='Enter positive integer or real value'
  KEY HELP=(F5)
  KEY NORMAL=(NEXT)
  KEY ABNORMAL=(F6) }
```

To find the area of a triangle:

Enter values for Side A: _____

Side B: _____

Side C: _____

Press: NEXT to continue
 F5 for help
 F6 to quit

PDU,test. <-- compile the panel into PANELIB
 SHOW,test. <-- test the panel

PERMIT Explicitly permit another user to access one of your private or semi-private files.

Syntax: PERMIT,pfn,un_1=m_1,un_2=m_2,...,un_n=m_n/NA.

Similar commands: COS: permit lists
 NOS/VE: CREATE_CATALOG_PERMIT; CREATE_FILE_PERMIT;
 DELETE_CATALOG_PERMIT;
 DELETE_FILE_PERMIT
 VMS: Access Control Lists

Examples: PERMIT,myfile,ABCD=R.
 ^-- allow ABCD to read the file

PURGALL Purge all your files which match the parameters.

Syntax: PURGALL,TY=ty,CT=ct,AD=ad,MD=md,CD=cd,AF,TM=tm,NA.

Parameters: ty - file type

ty.	meaning
I (INDIR)	all indirect files
D (DIRECT)	all direct files
A (ALL)	all files

(Default: TY=A, if any other parameter is specified)

ad - all files last accessed before (after, if AF)
 this date (yymmdd)

md - all files last modified before (after, if AF)
 this date (yymmdd)

cd - all files created before (after, if AF)
 this date (yymmdd)

AF - all files after AD, MD, or CD dates

tm - time-of-day on the AD, MD, CD date (hhmmss)

Remarks: AF, CT, DN, MA, TY, TM, and one date (either AC, MD, CD) fit on a single PURGALL command.

Similar commands: VMS: DELETE; PURGE

Examples: PURGALL,AD=881001.

^-- purge all (your) files not
accessed in 2 or more years
(assuming today is October 1,
1990)

PURGE Purge one or more direct or indirect permanent files.

Syntax: PURGE,pfn_1,pfn_2,...,pfn_n/UN=un,PW=pw,NA.

Remarks: If the file is attached, it remains as a local
file until RETURNed or LOGOUT.

Similar commands: COS: DELETE
NOS/VE: DELETE_FILE; DELETE_ALL_CYCLES
'DTRC)
VMS: DELETE; PURGE

Examples: PURGE,myobj/NA. <-- be sure file is not present
DEFINE,myobj. <-- before creating a new one

PUTASC (DTRC) Convert a 6/12-bit Display Code file to 8/12-bit ASCII
and make it a permanent indirect or direct file.

Syntax: BEGIN,PUTASC,,I=lfm,FN=fn.

Parameters: I= - name of 8/12-bit ASCII local file

FN= - name for the permanent (indirect or direct)
8/12-bit ASCII file

Remarks: This is useful in sending ASCII files to the
VAXcluster to be retrieved using HFT FETCH /ASCII.

An attempt is made to SAVE the file as an indirect
file. If it is too large (greater than 696 PRUS),
it will be DEFINed as a direct file.

See also: GETASC

Similar commands: VMS: HFT STORE /ASCII

Examples: BEGIN,PUTASC,,myfile,myfile.

^-- local file MYFILE will be
converted from 6/12-bit Display
Code to 8/12-bit ASCII and
saved as an indirect or direct
file

QGET Assign a queued file to your job.

Syntax: QGET, JSN=j sn, DC=q, UJN=uj n, FN=l fn.
QGET, j sn, q, uj n, l fn.

```
Parameters:  DC=q      - the queue containing the file
              q        meaning
              --        -----
              PR        print
              PU        punch
              PL        plot
              WT        wait
              IN        input
              (default: WT)
```

FN=lfname - the local file name to be given to the file

```
Examples:  SUBMIT,myjob,TO.      -or-      CSUBMIT,....
           ENQUIRE,JSN.        <-- get jsn of job
           QGET,jsn.           <-- get the file from the
                                wait queue
```

RATES (DTRC) Display the current computer rates.

Syntax: BEGIN,RATES,,OUTPUT.

Parameters: OUTPUT - output file
(default: OUTPUT)

Similar commands: NOS/VE: RATES
VMS: HELP RATES

```
Examples:  BEGIN,RATES.
              ^-- display on your terminal

           BEGIN,RATES,,myout.
              ^-- put into file MYOUT
```


RECLAIM Selectively backup and reload local and permanent files.

Syntax: RECLAIM,p1,p2,...,pn./dir1,opts1/dir2,opts2/...

Parameters: pi - parameter
 diri - directive
 opti - option

Remarks: No REQUEST is needed for a magnetic tape.

Dump tapes MUST be labelled.

RECLAIM tapes are compatible with PFDUMP and PFLOAD.

See also: See NOS 2 Reference Set Volume 3: System Commands
 for a 15-page discussion of the RECLAIM utility.

Similar commands: NOS/VE: BACKUP_PERMANENT_FILES;
 DISPLAY_RESTORE_PERMANENT_FILES
 VMS: BACKUP

Examples: RECLAIM,L=OUT.DB=0,Z./DUMP,TN=NA9876,EI=NO.
 ^-- simple dump of all files to tape
 without a database
 PURGALL,TY=A.
 ^-- AFTER you are sure the dump is
 correct!
 = = = = =
 RECLAIM,DB=0,Z./LOAD,TN=NA9876,PF=myfile.
 ^-- Reload a file

RECOVER (IAF) Recover a detached job or interrupted terminal session.

Syntax: RECOVER,JSN=jsn,OP=T

RECOVER,jsn,T

RECOVER

Parameters: jsn - job sequence number of the detached job
 T - abort recovery if no recoverable files
 (else start a recovery dialog)

Examples: /RECOVER,ABCD

REDO (IAF) Recall a previously entered command to modify and re-execute it without having to retype the entire command.

Syntax: REDO,string/GO
 R,string/GO

Parameters: string - the first up-to-10 characters or the command to be KEDOne (a blank or terminator in string ends the command) (default: the most recent command)

GO - re-execute without modification (OLD:, MOD:, NEW: prompts are suppressed)

Edit chars: space - leave character unchanged
 # - delete character any shift line to left
 & - replace character with a space
 ^ - insert characters before the marked character (end the inserted string with a #; ^ RETURN displays the command line as edited so far)
 ! - delete to the end of line
 other - replaces the original character

Similar commands: NOS/VE, VMS, VMS Cray Station: <UP arrow>

Examples: /REDO
 OLD: CATLIST,LO=F,FN=ABCDEFGF
 MOD: hij!
 NEW: CATLIST,LO=F,FN=HIJ
 = = = = =
 /REDO
 OLD: CATLIST,LO=X,FN=ABCDEFGF
 MOD: f hij!^
 NEW: CATLIST,LO=F,FN=HIJ <--- changes so far
 MOD: ^P
 NEW: CATLIST,LO=FP,FN=HIJ

REDUCE (Loader) Turn the reduce flag on or off.

Syntax: REDUCE. <--- turn reduce flag on
 REDUCE(-) <--- turn reduce flag off

Remarks: When on, the loader determines the field length assigned.

When off, you determine the field length with RFL statements.

See also: RFL

Examples: FTN5.
 LGO. <--- program executes in the FL needed
 RFL,50000.
 REDUCE(-)
 LGO. <--- program executes in 50000 words
 REDUCE. <--- next load executes in what is needed

RENAME Change the name of a local file.

Syntax: RENAME,nfn1=ofn1,nfn2=ofn2,...,nfnn=ofnn.

Parameters: nfni - the new name

ofni - the existing name

Remarks: Does not change the name in the permanent file directory.

Similar commands: NOS/VE: CHANGE_CATALOG_ENTRY
VMS: no local file concept

Examples: RENAME,that=this.

^-- change local file name THIS to
THAT

REPLACE Purge an indirect file and replace it with a copy of a local file; save a copy of a local file as a new indirect file.

Syntax: REPLACE,lfn_1=pfn_1,lfn_2=pfn_2,...,lfn_n=pfn_n/
UN=un,PW=pw,M=m,NA.

Remarks: If the file already exists, the catalog type (CT=) and all other information about the file is preserved; if it does not, a new file is created with CT=PRIVATE.

See also: SAVE; DEFINE

Similar commands: NOS/VE: COPY_FILE; CREATE_FILE

Examples: REPLACE,mylfn=myprog

^-- replaces indirect file MYPROG
with the contents of local file
MYLFN

REQUEST Request a tape be mounted.

Remarks: Use LABEL.

REQUEST Assign a file to receive checkpoint dumps, or send a message to the operator.

Syntax: REQUEST,lfn,ckpt.comment
 REQUEST,NE.message_to_operator

Parameters: ckpt - lfn is to be a checkpoint file

ckpt	meaning
CK	put each dump at end of lfn
CB	put each dump at beginning of lfn

comment - message to the operator about device assignment

Similar commands: NOS/VE: REQUEST_MAGNETIC_TAPE

Examples: REQUEST,NE.message
 ^-- <message> is displayed at the
 operator's console
 = = = = =
 REQUEST,lfn,CK.
 ^-- save all checkpoints
 REQUEST,lfn,CB.
 ^-- save the last checkpoint
 REQUEST,lfn1,CB.
 REQUEST,lfn2,CB.
 ^-- save consecutive checkpoints by
 alternating two checkpoint
 files.
 DEFINE,lfn.
 REQUEST,lfn,CK. -or- ASSIGN
 CKP.
 ^-- make checkpoint file permanent

RERUN Allow a job to be rerun if necessary.

Syntax: RERUN.

Remarks: A job is normally rerunable unless it does something which might make a rerun fail, such as creating, modifying or deleting a file, etc.

See also: NORERUN

Similar commands: COS: RERUN

Examples: RERUN.

RESOURC Specify that more than one tape drive is required.

Syntax: RESOURC,rt1=u1,rt2=u2,...,rtn=un

Parameters: rti - resource type

- LO - 7-track tape, 200 cpi
- HI - 7-track tape, 556 cpi
- HY - 7-track tape, 800 cpi
- HD - 9-track tape, 800 cpi
- PE - 9-track tape, 1600 cpi
- GE - 9-track tape, 6250 cpi

ui - maximum number of units this job will use concurrently

- 0 - clear a resource type that is no longer required
- 2 - maximum at DTRC (LO, HI, HY, HD)
- 3 - maximum at DTRC (PE, GE)

Remarks: Jobs needing only a single 9-track tape drive at a time, even for a multi-reel file, do not need a RESOURC statement.

RESOURC should precede the first tape request.
Subsequent RESOURC statements may change any RT=U.

This statement helps prevent deadlock.

Similar commands: NOS/VE: RESERVE_RESOURCES
VMS: ALLOCATE

Examples: RESOURC,GE=2. <-- two 6250 cpi, 9-track tapes
are required at once

```

=====
JOB123.
USER,....
CHARGE,....
RESOURC,PE=2. <-- 2 1600-cpi tapes needed
LABEL,T1,D=PE,VSN=tape1.
LABEL,T2,D=PE,VSN=tape2.
...
RETURN,T1,T2.
RESOURC,PE=1,GE=1. <-- 1 1600 and 1 6250
...

```

RESTART Restart a checkpointed job.

Syntax: RESTART,lfn,nnnn,x_i.

Parameters: lfn - the checkpoint file
(must have write permission)

nnnn - number of the checkpoint
 (Default: 1; use * for last checkpoint)

x_i - RI - do not restore command file on lfn
 NA - do not abort if a required file is
 not available; if read parity while
 restoring a file in checkpoint nnnn,
 use checkpoint nnnn-1
 FC - do not restore files ZZZZC0, C1, C2,
 if already local

Examples: RESTART,ckpfyl,*.

RETURN Release files (and file space depending on file type) assigned
 to a job.

Syntax: RETURN,lfn1,lfn2,...,lfnn. <-- all listed
 files

RETURN,*,lfn1,lfn2,...,lfnn. <-- all but listed
 files

Parameters: lfni - a file assigned to your job

See also: CLEAR, EVICT, UNLOAD

Similar commands: COS: RELEASE
 NOS/VE: DETACH_FILE; RELEASE_RESOURCES
 VMS: CLOSE it in a program

Examples: RETURN,dtlib,sublib,work1,out.

REVERT Return from a procedure.

Syntax: REVERT,opt.com

Parameters: opt - revert option

opt	meaning
ABORT	return to next EXIT, unless NOEXIT (REVERT appears at your terminal and in the job dayfile)
EX	return to calling procedure and execute command <com> (REVERT appears in the job dayfile but not at your terminal)
NOLIST	return to calling procedure (REVERT does not appear at your terminal or in the job dayfile)

com - for opt EX: the command to be executed
 for other opt: a comment

Remarks: The following statements are supplied automatically at the end of a procedure to insure that a REVERT is present:

\$REVERT.CCL
\$EXIT.CCL
\$REVERT,ABORT.CCL

Similar commands: COS: RETURN
NOS/VE, VMS: EXIT

Examples: .PROC,MYPROC.
.* the body of your procedure
REVERT,NOLIST.
EXIT.
DMP,30000.
REVERT,ABORT.

REWIND Position files at beginning-of-information (BOI).

Syntax: REWIND,1fn1,1fn2,...,1fnn. <-- all listed
files

REWIND,*,1fn1,1fn2,...,1fnn. <-- all but listed
files

Parameters: 1fni - file assigned to your job

Similar commands: COS: REWIND
NOS/VE: REWIND_FILE

Examples: REWIND,myfile.

RFL Set field length for the next program execution.

Syntax: RFL,CM=nnnnnn.
RFL,nnnnnn.

Remarks: nnnnnn may not exceed the last MFL or job
statement setting.

See also: MFL, REDUCE

Examples: FTN5.
LGO.
RFL,50000.
REDUCE(-)
LGO.
REDUCE.

ROUTE Direct the disposition of an indirect file and define its characteristics.

Syntax: ROUTE, lfn, parameters.

Parameters: DC=dc - disposition code
 IN - input queue (output to central site or the user specified by UN)
 LP - any printer
 PL - plot
 PR - same as LP
 PU - punch coded
 P8 - punch 80-column binary
 SB - punch system binary
 SC - rescind prior routine and make the file type local
 TO - input queue (output to wait queue)

(default: same as in previous ROUTE for this lfn; if none, DC=SC, except these special names:

name	DC
-----	--
OUTPUT	PR
PUNCH	PU
PUNCHB	SB
P8	P8)

DC - same as DC=SC

DEF - defer routing until end-of-job
 (default: do it now;
 not allowed with DC=IN, NO, TO)

EC=ec - external characteristics for print and punch files
 print:

ec	meaning
-----	-----
A6	ASCII graphic 63/64-char set
A9	ASCII graphic 95-char set (lfn must be 7-bit ASCII8)

FC=fc - forms code

FC - use standard print or card forms

FID=ujn - NOS/BE compatibility (same as UJN=)

REP=rep - number of extra copies
 (default: 0 (only 1 copy printed);
 maximum: 31 (37B))

TID= - see UN

UJN=ujn - user job name for the file (not input)

UN=un - user name of the receiving remote batch
 or interactive user
UN - implicit remote routing

Remarks: In general, if a parameter is omitted, it
 retains the definition from the last ROUTE
 command which referenced that "lfn". The
 exception is DEF.

Similar commands: COS: DISPOSE
 NOS/VE: PRINT_FILE; SUBMIT_JOB
 VMS: SUBMIT; PRINT; XEROX (DTRC);
 FICHE (DTRC)

Examples: CATLIST,LO=F,L=out1.
 ROUTE,out1,DC=PR. <-- print at Central Site
 = = = = =
 ROUTE,mydata,DC=PU,UJN=xxxxx.
 ^-- punch deck with banner
 card of "xxxxx"

RTIME Put the real-time clock time into the dayfile.

Syntax: RTIME.

See also: CTIME, STIME

Similar commands: VMS: ^T

Examples: RTIME.

SATISFY (Loader) Satisfy unsatisfied externals now, instead of at the
 end of the loading.

Syntax: SATISFY.
 SATISFY,lib1,lib2,...,libn.

Parameters: libi - a specific library to be searched in the
 listed order
 (default: all known libraries are searched)

Examples: LOAD,bin1.
 SATISFY(mylib)
 LOAD,bin2.
 SATISFY.
 bin3.

SAVE Put a copy of a local file on disk as an indirect file.

Syntax: SAVE, lfn_1=pfn_1, lfn_2=pfn_2, ..., lfn_n=pfn_n
/PW=pw, CT=ct, M=m, SS=ss, BR=br, PR=pr, NA=ac.

See also: REPLACE; DEFINE

Similar commands: NOS/VE: COPY_FILE; CREATE_FILE

Examples: SAVE, mytemp=keepit/CT=PU.
 ^-- save local file MYTEMP as a
 public file unless KEEPIT
 already exists

SCOPY Copy coded file(s) displaying EORs and EOFs in the receiving file.

Syntax: SCOPY, lfn_in, lfn_out, n, fchar, lchar, na, R, fcs,
 fline, lline, ns.

Parameters: n - decimal number of files to copy
 (default: copy to EOI)

fchar - first character position of line to copy

lchar - last character position of line to copy

na - do not abort if no line terminator
 before EOR

R - rewind lfnin and lfnout before copying

fcs - character set of lfnin
 0 - display code or 6/12-bit display
 code
 (default: 0; no other value allowed)

fline - first line of (sequenced) file to be
 copied
 (default: 1)

lline - last line of (sequenced) file to be
 copied
 (default: controlled by n)

ns - any non-null value to suppress EOR/EOF
 display in lfnout

Remarks: Do not use with S, L, F or SI tapes.

A file without an EOR will have one added to the
end of the listing.

See also: COPY (S, L, F tapes), TCOPY (SI tapes)

Examples: SCOPY,myfile.

SCREEN (IAF) Set your terminal for screen mode.

Syntax: SCREEN,TM=model.

SCREEN,model.

Parameters: model - the terminal mode

model	meaning
DT100	DEC VT100-compatible for FSE at DTRC
VT100	alternate DEC VT100-compatible
other	call User Services
omitted	no change

modelT - append T for type-ahead capability

Remarks: SCREEN may be included in a procedure.

Affects FSE, HELPME, screen formatting, and the display of NOS procedure parameters.

You may wish to put "SCREEN,DT100." into your LOGINPR file.

See also: LINE

Similar commands: NOS/VE: CHANGE_INTERACTION_STYLE

Examples: SCREEN,DT100 <-- set for full-screen editing
FSE,myfile,G. <-- edit in full-screen mode

SET Assign a value to a control register, an error flag, or the enter-skipped-commands-in-the-dayfile flag; change the current interactive subsystem.

Syntax: SET,symb1=exp1,symb2=exp2,...,symbn=expn.

Parameters: symbi - one of:

name	meaning
R1, R2, R3	local control registers (initial value: 0)
R1G	global control register (initial value: 0)
EF	local error flag (initial value: 0)

EFG	global error flag (initial value: 0)
DSC	dayfile_skipped_command flag 0 - do not put skipped commands into dayfile 1 - put skipped commands into dayfile (initial value: 0)
PL or PS	page length (or page size) (default: 60)
PW	page width (default: 136)
PD	page density (default: 6 lines / inch)
SS	interactive subsystem

expi - any valid expression

symbol	range
R1, R2, R3, R1G	-131071 to 131071
EF, EFG	0 to 63
DSC	1 or 0
PL or PS	16 to 255
PW	40 to 255
PD	6 or 8
SS	ACCESS, BASIC, BATCH, EXECUTE, FORTRAN, FTNTS, NULL

Similar commands: COS: SET
NOS/VE: var = value (SCL)
VMS: \$ var = value (DCL)

Examples: SET,R1=0.
WHILE,R1=5,LOOPEND.
...
SET,R1=R1+1.
ENDW,LOOPEND.

SETASL Set the SRU limit for an accounting block.

Syntax: SETASL,s.
SETASL,*<--- set to your maximum SRU limit

Parameters: s - maximum number of SRUs allowed
(decimal, or octal with B suffix)
(generally, s must be >= the current job step
SRU count and <= your SRU limit)

See also: SETJSL

Similar commands: NOS/VE: CHANGE_JOB_LIMIT

Examples: SETASL,2000.

SETCORE Preset each word of the field length except for RA+2.

Syntax: SETCORE,p.
SETCORE,-p.

Parameters: p - desired setting (-p sets the complement of p)
p fill characters

-----	-----
0	0
ZERO	zeros (0)
INDEF	indefinite (1777 0000 ... 0000)
INF	infinite (3777 0000 ... 0000)
(Default: 0)	

Remarks: To preset memory with a load sequence, use
LDSET,PRESET.

Examples: RFL,100000.
SETCORE. <-- immediately clear FL to 0.

SETFS Set the auto-drop/no-auto-drop status of files assigned to
your job.

Syntax: SETFS,1fn1,1fn2,...,1fnn/FS=fs.

Parameters: FS= - auto-drop status
fs meaning

AD auto-drop
NAD no-auto-drop

Remarks: Files with no-auto-drop set are not returned by
CLEAR, RETURN(*), or UNLOAD(*) .

Examples: SETFS,fyl1,fyl2/FS=NAD.

SETJOB Change some of the current job's attributes.

Syntax: SETJOB,UJN=ujn,DC=dc,OP=op.
SETJOB,ujn,dc,op.

Parameters: ujn - new job name

dc - disposition code

dc	meaning
---	-----
TO	queue output with wait disposition
NO	discard output (no dayfile)
DF	default output processing

op - job processing option	
op	meaning

SU	job remains suspended until recovered or timed out
TJ	system terminates the job

See also: RECOVER

Similar commands: NOS/VE: CHANGE_JOB_ATTRIBUTES

Examples: SETJOB,xxxxxx.
 ^-- set UJN for banner pages (batch
 and interactive)

SETJOB,,NO. <-- discard file OUTPUT (including
 the dayfile) after the end of a
 batch job

SETJSL Set the SRU limit for each subsequent job step.

Syntax: SETJSL,s.
 SETJSL,* <-- set to your maximum SRU limit

Parameters: s - maximum number of SRUs for job step execution

Similar commands: NOS/VE: CHANGE_JOB_LIMIT

Examples: SETJSL,250.

SETPR Do not use at DTRC.

SETTL Set the CPU time limit for each subsequent job step.

Syntax: SETTL,t.
 SETTL,* <-- set to unlimited

Parameters: t - maximum number of CPU seconds for job step
 execution
 (default: 64 decimal)

See also: ENQUIRE, LIMITS

Similar commands: NOS/VE: CHANGE_JOB_LIMIT

Examples: SETTL,5.

SHOW (IAF) Display a screen formatting panel for testing purposes.

Syntax: SHOW,I=panelname.

Parameters: I= - the name of a compiled panel in user library
PANELIB or in a global library set

Remarks: SHOW is an interactive procedure (? for help).

SKIP Unconditionally skip succeeding commands, ending with an ENDIF
with a matching label.

Syntax: SKIP,label.

Parameters: label - alphanumeric string (1-10 characters,
starting with a letter)

See also: ENDIF

Similar commands: VMS: GOTO

Examples: IF(R1<=1,DONE)

```
...  
    SKIP(DONE)  
IF(R1=2,DONE)  
...  
    SKIP(DONE)  
...  
ENDIF(DONE)
```

SKIPEI Position a file at end-of-information.

Syntax: SKIPEI,lfn.

Remarks: On magnetic tape with no EOI defined, stops at
EOF.

See also: SKIPF, SKIPFB, SKIPR

Similar commands: COS: SKIPD
VMS: OPEN with ACCESS=APPEND in a program

Examples: SKIPEI,myfile.

SKIPF Skip forward a specified number of files.

Syntax: SKIPF,lfn,n,m.

Parameters: n - decimal number of files to skip
(default: 1; max: 262143)

```

m - coded or binary
  m  meaning
  -  -----
  B  binary
  C  coded
    (default: B; C with SI tape is fatal)

```

Remarks: Will stop at EOI.

See also: SKIPEI, SKIPFB, SKIPR

Similar commands: COS: SKIPD; SKIPF; SKIPR; SKIPU
 NOS/VE: SKIP_TAPE_MARK

Examples: SKIPF,myfile,4,C.
 ^-- skip 4 coded files

SKIPFB Skip backward a specified number of files.

Syntax: SKIPFB,lfn,n,m.

Parameters: n - decimal number of files to skip
 (default: 1; max: 262143)

```

m - coded or binary
  m  meaning
  -  -----
  B  binary
  C  coded
    (default: B; C with SI tape is fatal)

```

Remarks: Will stop at BOI.

See also: SKIPEI, SKIPF, SKIPR

Similar commands: COS: SKIPD; SKIPF; SKIPR; SKIPU
 NOS/VE: SKIP_TAPE_MARK

Examples: SKIPFB,myfile,4,C.
 ^-- skip back 4 coded files

SKIPR Skip forward a specified number of record or file marks.

Syntax: SKIPR,lfn,n,level,m.

Parameters: n - decimal number of files to skip
 (default: 1; max: 262143)

```

level - level number (0-17)
       0-16 - EORs and EOFs counted
       17   - EOFs counted
       (default: 0)

```


m - coded or binary
m meaning
- -----
B binary
C coded
(default: B; C with SI tape is fatal)

Remarks: Consecutive EORs or EOFs are counted separately.

Will stop at EOI.

See also: SKIPEI, SKIPF, SKIPFB

Similar commands: COS: SKIPD; SKIPF; SKIPR; SKIPU

Examples: SKIPR,myfile,4,17.
 ^-- skip 4 binary files

SLOAD (Loader) Selective load modules from a file.

Syntax: SLOAD,lfn,name1,name2,...,namen.

Parameters: lfn - the file from which the listed modules
 are to be loaded

namei - the name of a module to be loaded

See also: LIBLOAD

Similar commands: VMS: LINK file/INCLUDE=

Examples: SLOAD,mybin,suba,subb,subg.

SORT This deals with sequenced files and is NOT the Sort/Merge program.

Remarks: SORT5 is the Sort program; MERGE is the Merge program.

SORT5 Sort files.

Syntax: FILE,lfnin1,...
 FILE,lfnin2,...
 ...
 FILE,lfnout,...
 SORT5.p1,p2,...,pn
 or
 SORT5.p1 p2 ... pn

Positional: SORT5.from,to,key,dir,1,,e,e1,
 dialog,end,,,ownf,ownfl,
 ownmrl,,own1,own2,own3,
 own4,own5,retain,seqa,
 seqn,seqr,seqs,status,sum,,
 verify,fastio.

Interactive: SORT5.DIALOG=YES

Directive file: SORT5.DIR=1fn
 SORT5.params,DIR=1fn
 SORT5.DIR=1fn,params
 SORT5.params,DIR=1fn,..
 .more_params

Parameters: FROM=1fn
 FROM=(1fn1,1fn2,...,1fnn)
 Up to 100 input files, read in the order
 specified, normally rewound before and
 after use.

TO=1fn
 The file to receive the sorted records,
 normally rewound before and after use.

KEY=(key_def,key_def,...)
 key_def - range -or-
 (range,type,ad) -or-
 (first,length,type,ad)
 range - first -or-
 first..last
 first - first byte/bit of key field
 last - last byte/bit of key field
 length - number of bytes/bits in key
 (default: 1)
 type - name of numeric data format
 or collating sequence
 (default: ASCII6)
 ad - order: A (ascending)
 D (descending)
 (default: A)

Up to 100 key-defs may be specified.

Keys are sorted first by the leftmost
 key_def.

If no keys are specified, KEY=1..mnr
 (minimum record length; smallest MNR or
 smallest FL or MRL on FILE statements) is
 used.

DIR=lfm

DIR=(lfm1,lfm2,...)

Read SORT5 parameters from one or more files.

(default: no directive file is read; the parameters of the SORT5 statement completely define the sort)

L=lfm

Output listing information.

(default: OUTPUT)

E=lfm

Error listing file.

(default: the L= file)

EL=el

Error level to be reported:

T - trivial + W, F, C

W - warning + F, C

F - fatal + C

C - catastrophic

(default: W)

DIALOG=YES or DIA=Y

Interactive dialog. May appear only in the SORT5 control statement. All information for the sort is entered in response to questions. All other parameters specified in the SORT5 statement, except STATUS, are ignored.

ENR=expr

ENR=expr..expr

The estimated number of records to be sorted (single decimal integer 0-10**9, a range of values, or one of the CCL variables: R1, R2, R3, R1G, EF, or EFG. (Use especially if ENR < 1500)

RETAIN=retain or RET=r

Specify the order for records with equal sort keys.

retain	meaning
YES or Y	records with equal keys retain their original order
NO or N	records with equal keys might not retain their original order

(default: NO)

STATUS=variable or ST=variable

Report the SORT5 status to one of the CCL variables: R1, R2, R3, R1G, EF, or EFG.

code	meaning
0	no errors
20	trivial
30	warning
40	fatal
50	catastrophic

Remarks: Each line of the SORT5 control statement or the directive file may be up to 100 characters, but characters beyond column 80 are ignored.

Batch: To continue on more than one line, end one line with two periods and start the next line with one period. CAUTION: because a line range is indicated by two periods, ranges must not be continued.

Interactive: Lines cannot be continued. If more than one line is needed, use a directive file or a procedure.

FILE statements are required for each file. The maximum record is specified with the FL parameter (if RT=Z or F) or MRL parameter (all others).

In the positional illustration above, reserved positions are indicated by adjacent commas.

See Sort/Merge Version 5 Reference Manual, 60484800, for other parameters.

See also: MERGE

Similar commands: COS, NOS/VE, VMS: SORT

Examples: FILE,infyl,BT=C,RT=Z,MRL=80.
 FILE,outfyl,BT=C,RT=Z,MRL=640.
 SORT5.infyl,outfyl,5..10
 ^-- sort columns 5-10 into ascending
 ASCII6 order
 SORT5.FROM=infyl,KEY=((5..10,,D)),TO=outfyl
 ^-- same, except descending
 = = = = =
 SORT5.KEY=6..25
 ^-- sort one 20-byte key starting in
 byte 6
 SORT5.KEY=((6,20))
 ^-- same as above
 SORT5.KEY=(6,20)
 ^-- sort two 1-byte keys (major key
 in byte 6, minor key in byte 20)

SORT5.KEY=6,20

^-- sort byte 6, read directives
from local file "20" (this is
the next positional parameter)

STIME Put the accumulated SRU value for the job into the dayfile.

Syntax: STIME.

See also: CTIME, ENQUIRE,S, RTIME

Similar commands: VMS: ^T

Examples: STIME.

SUBMIT Put a job into the input queue.

Syntax: SUBMIT,lfn,q,NR.c

Parameters: lfn - the file to be submitted - the first record
must be in 6-bit display code

q - output disposition
BC or B - central site
NO or N - discard output unless
specifically routed within the
batch job - no dayfile
RB=un - route output to a remote
batch terminal or interactive
user
TO - queued with wait disposition
(Default: TO)

NR - do not rewind the submit file or cREAD file
before or after processing
(Default: rewind)

c - prefix character for reformatting
directives in the file (assumes /JOB is the
first statement)
(Default: /)

Remarks: For both direct and indirect files.

See also: CSUBMIT, ROUTE

Similar commands: COS: SUBMIT
NOS/VE: SUBMIT_JOB
VMS: SUBMIT; CRAY SUBMIT; CSUBMIT (DTRC)

Examples: SUBMIT,myjob,BC.
 ^-- print at Central Site
 SUBMIT,myjob,RB=xxxx.
 ^-- use QGET to retrieve output from
 print queue
 SUBMIT,myjob,TO.
 ^-- use QGET to retrieve output from
 wait queue

SWITCH Set sense switches.

Syntax: SWITCH,switch_1,switch_2,...,switch_n,jsn.

Parameters: switch_i - switch to be set (1-6)
 0 - set all switches

jsn - may appear in any parameter position
 (Default: the current job)

Similar commands: COS: SWITCH
 NOS/VE: SET_SENSE_SWITCH

Examples: SWITCH,1,3,13
 ^-- turn on sense switches 1, 3, 5

TAPDMP (DTRC) Driver to execute TAPEDMP program.

Syntax: BEGIN,TAPDMP,,L,SLOT,VSN,D,CV,LB,
 NFILE,NREC,MODE,ISTART,PCODE,
 PRFILE,NUMPRT,PECODE,LENGTH,
 I,KEEP,VSNA.

Parameters: L - listable output file
 (default: OUTPUT)
 SLOT - Slot number
 VSN - Volume Serial Number
 D - Tape density
 9-track: GE or 6250; PE or 1600; HD (800)
 (default: PE)
 CV - Tape conversion mode (AS or EB)
 (defaults: omitted: no conversion (binary);
 keyword: AS)
 LB - Tape labelling
 lb meaning
 -- -----
 KL labelled
 KU unlabelled
 (defaults: omitted: KU; keyword: KL)
 I - input file containing your NAMELIST
 statement
 (if I=INPUT, the NAMELIST statement must
 be entered on one line)
 (default: TAPDMP will generate the
 NAMELIST statement)

KEEP - tape disposition
 K or KEEP - do not reposition the tape
 R or REWIND - rewind the tape
 other - unload the tape
 Use K or R if you wish to dump more of the
 tape without remounting it each time
 (default: the tape is unloaded)
 VSNA - alternate VSN
 (e.g., if external sticker is NAO123
 but internal VSN is NAO123, use both VSN
 and VSNA)

Remarks: See the description of TAPEDMP for the following
 parameters: NFILE, NREC, MODE, ISTART, PRCODE,
 PRFILE, NUMPRT, PECODE, LENGTH.

Not all features of TAPEDMP are supported by
 TAPDMP. Use TAPEDMP directly for them.

Use TDUMP to dump a 7-track tape.

See also: TAPEDMP, TDUMP

Similar commands: NOS/VE: DISPLAY_FILE
 VMS: DUMP

Examples: BEGIN,TAPDMP,,out1,99,mytap1,NFILES=2.
 ^-- analyze the first t2o files with
 output to file OUT1 -- since then
 tape is mounted unlabelled - the
 first file could be the label
 = = = = =
 BEGIN,TAPDMP,,out2,99,mytap1,CV=AS,MODE=0,
 PRCODE=2.
 ^-- alphanumeric dump of ASCII tape
 = = = = =
 BEGIN,TAPDMP,,out3,99,mytap1,CV=EB,MODE=2,
 PRCODE=4.
 ^-- hexadecimal dump of EBCDIC tape
 = = = = =
 BEGIN,TAPDMP,,out4,99,mytap1,NFILE=2,KEEP=R.
 ^-- analyze the tape; rewind after
 <use FSE to examine summary output in file OUT4>
 BEGIN,TAPDMP,,out5,99,mytap1,MODE=0,PRCODE=2,
 NFILE=2,PRFILE=2,NREC=10.
 ^-- alphanumeric dump of the first
 10 records of file 2

TAPEDMP (DTRC) Analyze a magnetic tape.

Syntax: TAPEDMP,infile,outfile.

Parameters: infile - input directives in Fortran NAMELIST
form (\$NPUT ... \$END)
(default: INPUT)

outfile - listable output
(default: OUTPUT)

Remarks: TAPEDMP was obtained from NASA.

TAPEDMP performs the following functions:

Analyze: determine the recording mode (ASCII, EBCDIC, or binary for 9-track; BCD or binary for 7-track) and summarize the analysis of a specified number of data blocks.

Dump: print the data in either alphanumeric, octal, hexadecimal, or external BCD.

List all magnetic tape labels.

If TAPEDMP is executed interactively, the output is formatted for 80-character output.

The tape must be mounted as local file TAPE1 and positioned properly.

Extreme caution should be exercised when using the detail print options to avoid voluminous output.

For 7-track tapes, if a mode is specified, an attempt to read at this mode will be made and, if there is an error, the mode will be switched.

In the printout, ASCII/BCD for coded tapes indicates ASCII for 9-track and BCD for 7-track.

When detailed printing (PRCODE) is requested, the unused bit count will be indicated for each record.

TAPEDMP attempts to process all errors and overrides system aborts which would normally occur. If TAPEDMP cannot continue, the following message is issued to the dayfile:

SYSTEM ABORT - ERROR FLAG=xx
where xx is the RPV (REPRIEVE) error code.

Directives: The directives are specified in Fortran NAMELIST form (\$NPUT ... \$END) using the following variables:

ISTART = record number to begin printing
(default: 1)

LENGTH = length in words of the longest block to be read - if > 514, RFL and REDUCE(-) are required to increase the execution field length to 23100 (octal) by the amount over 514
(default: 514)

LO = list label option (requires LB=KU and F=S or F=L in the LABEL statement)
0 - do not list labels
<>0 - list all tape labels (when selected, all other TAPEDMP parameters are ignored)
(Similar information is available from the LISTLB command)
(default: 0)

MODE = processing mode for magnetic tape
0 - ASCII for 9-track; BCD for 7-track
1 - binary
2 - EBCDIC for 9-track (requires CV=EB in the LABEL statement)
4 - analyze data/determine correct mode ||
(default: 4)

NFILE = number of files to analyze
(default: 1)

NREC = number of records to analyze
(default: all records)

NUMPRT = number of records per file to print
(default: all records)

PECODE = parity error print code
1 - no parity error print
0 - print parity error code according to PECOde (if PECOde=0, it is printed in octal)
(default: 1)

PRCODE = detail print code
0 - no detail print
1 - octal
2 - alphanumeric
3 - octal and alphanumeric
4 - hexadecimal (9-track only)
5 - external BCD (7-track only)
(default: 0)

PRFILE = file number in which to start detail
print - printing continues through NFILE
(default: 1)

Messages: END OF TAPEDMP

END-OF-TAPE ENCOUNTERED, TAPE DUMP TERMINATED
The end of this tape has been encountered.

EOF ENCOUNTERED
A physical tape mark was encountered.

INVALID INPUT DATA
NAMELIST data invalid or used incorrectly.

LENGTH IS NOT GT THAN ZERO
A system error has occurred in creating the
file buffer - increase the value of LENGTH.

LEVEL 17 EOF ENCOUNTERED
A software-created level 17 EOF was
encountered.

TAPE IS NOT LABELLED
LO<>0 was requested but no tape labels were
found - be sure LB=KU is in the LABEL
statement.

TAPE1 MUST BE A TAPE FILE

See also: LISTLB, TAPDMP, TDUMP

Similar commands: NOS/VE: DISPLAY_FILE
VMS: DUMP

Examples: Three ways of dumping (alphanumeric) 20 records
from file 1 of ASCII tape MYTAP1 in SLOT99.

Batch:
jobname,....
USER,....
CHARGE,....
LABEL,TAPE1,D=PE,F=L,PO=R,VSN=SLOT99=mytap1,
LB=KU.
ATTACH,UTILITY/UN=NSYS.
LDSET,LIB=UTILITY. <-- or LIBRARY,UTILITY.
TAPEDMP.
<eor>
\$NPUT NREC=20, PRCODE=2, MODE=2 \$END
<eoi>
= = = = =

Interactive:

```
NOTE,in./ $NPUT NREC=20, PRCODE=2, MODE=2 $SEND
LABEL,TAPE1,D=PE,F=L,PO=R,VSN=SLOT99=mytap1,LB=KU
ATTACH,UTILITY/UN=NSYS
LIBRARY,UTILITY
TAPEDMP,in
REWIND,TAPE1
```

```
= = = = =
```

Via procedure TAPDMP:

```
BEGIN,TAPDMP,,,99,mytap1,NREC=20,PRCODE=2,MODE=2.
```

```
-or-
```

```
BEGIN,TAPDMP,,,99,mytap1,,,,20,2,2.
```

```
= = = = =
```

```
$NPUT NFILE=4,
      PRCODE=2,
      NUMPRT=3,
```

```
$SEND
```

```
^-- short output which tells most of
    what you want to know about the
    tape
```

TCOPY Copy X (binary), E, B, or SI (coded) tapes to disk, I, or SI (binary) tape.

Syntax: TCOPY, lfn_in, lfn_out, format, tc, copycnt, charcnt, erlimit, plp2, lfnlst, ns.

```
TCOPY, I=lfn_in, O=lfn_out, F=format, TC=tc,
      N=copycnt, CC=charcnt, EL=erlimit, PO=plp2,
      L=lfnlst, NS=ns.
```

Parameters: I= - input file to be copied
(default: INPUT)

O= - the output copied file
(default: OUTPUT)

CC= - used for obsolete E or B tapes

EL= - number of non-fatal errors before abort;
EL=U for unlimited error processing
(default: 0; ignored for E/B output or
terminal input)

F= - type of conversion

format	meaning
E	obsolete
B	obsolete
X	obsolete
SI	SI coded tape to disk (SI tape is labelled or unlabelled and assigned as S)

(default: X)

L= - alternate file for parity error messages
for EL<>0; cannot be same as I=, 0=
(default: OUTPUT)

N= - copy count (meaning determined by TC=)
(default: N=1)

NS= - noise size for E to B conversion
(maximum: 41; NS=0 uses default of 18)

PO= - processing options:

pi	meaning
E	copy input blocks with parity or block-too-long errors (default: error blocks skipped)
T	truncate long blocks for E/B output (default: split into multiple blocks)

TC= - termination condition with N=

tc	meaning
F or EOF	N is number of files
I or EOI	N is ignored (copy to end-of-information)
D or EOD	N is number of double EOFs to copy to (default: TC=D)

See also: COPY (S, L, F tapes), SCOPY (display EOR/EOF)

Similar commands: NOS/VE: DISPLAY_FILE

Examples: TCOPY,tape,disk,SI.
 ^-- copy a complete coded NOS/BE
 tape

TDU (IAF) Compile a terminal definition file and store it in a
user library which can later be accessed by a SCREEN or LINE
command.

Syntax: TDU,I=definition,L=listing,LIB=library.

Parameters: I= - the terminal definition file in 6/12-bit
display code

L= - the output listing file
(default: OUTPUT)

LIB= - the library to receive the load capsule
(default: TERMLIB)

Remarks: TDU may appear in a procedure.

Similar commands: NOS/VE: DEFINE_FILE

TDUMP Octal or alphanumeric dump of all or part of a file.

Syntax: TDUMP,p1,p2,...,pn.

Parameters: I=lfm - local file to be dumped
(default: TAPE1)

L=lfm - listable output (never rewind)
(default: OUTPUT)

O - octal dump only
A - alphanumeric dump only
(if both specified, last is used)
(default: octal and alphanumeric)

R=rcount - decimal maximum number of records to
dump (restarts for each file)
(default: omitted or R=0: dump to EOI)

F=fcount - decimal maximum number of files to dump
(default: dump to EOI; F=0 => dump
until double EOF or EOI)

N=ncount - decimal maximum number of output lines
with each output line containing 40
input characters
(default: omitted or N=0: dump to EOI)

NR - do not rewind input file

See also: TAPEDMP

Similar commands: NOS/VE: DISPLAY_FILE

Examples: LABEL,TAPE1,....
TDUMP.
= = = = =
TDUMP,R=5,N=100.
^-- dump 5 tape records but no more
than 100 output lines

TPAUDIT (DTRC) Audit the magnetic tapes assigned to you.

Syntax: BEGIN,TPAUDIT.

See also: TPGET, TPRLS

Examples: BEGIN,TPAUDIT.

TPGET (DTRC) Get up to 3 magnetic tapes for use on the CDC CYBER 860
or DEC VAXcluster.

Syntax: BEGIN,TPGET.

Remarks: You will be asked how many tapes you want and then, after they have been selected, to verify that you want them.

See also: TPAUDIT, TPRLS

Examples: BEGIN,TPGET.

TPRLS (DTRC) Release magnetic tapes assigned to you by TPGET.

Syntax: BEGIN,TPRLS.

Remarks: You will be asked the numbers of the tape(s) to be released, and to verify that they are to be released.

See also: TPGET, TPRLS

Examples: BEGIN,TPRLS.

TRMDEF (IAF) Change terminal characteristics. Use in your prologue to set terminal characteristics if you normally use a terminal other than the default kind.

Syntax: TRMDEF,L=lfm,tcl=v1,...,tcn=vn.

Parameters: L=lfm - listable output

tci=vi - new terminal characteristic(s)

tci	meaning
EP	echoplex mode YES - full duplex NO - half duplex
PW	page width (20-255) 0 - infinite
FL	fold long lines ON - fold OFF - do not fold

vi	meaning
v	any alphanumeric character (e.g., \$*\$; display code 0-44B)
\$v\$	any character (dollar-delimited) (for "\$", user \$\$\$\$)
vvvB	octal value ASCII character (e.g., 52B (same as \$*\$))
vvD	decimal value of ASCII character (e.g., 42D (same as \$*\$))
Hvv	hexadecimal value of ASCII character (e.g., X2A (same as \$*\$))

UNIROUT (DTRC) Shift a UNICOS ASCII8 file and print it.

Syntax: BEGIN,UNIROUT,,FROM=from,UJN=ujn,SITE=site,
REP=rep.

Parameters: FROM - the ASCII8 file to be printed

UJN - the jobname (cover sheet banner)
(4-7 alphanumeric characters starting with
User Initials)

SITE - the print site
site meaning

C print at Central Site (Carderock)
(default: C)

REP - the number of extra copies to print (0-31)
(default: 0 (print original and no extra
copies))

Remarks: This is useful for printing batch job output from
UNICOS on the Cray.

See also: ROUTE

Examples: BEGIN,UNIROUT,,crayout,abcduni,A.

UNLOAD Release files assigned to your job and perhaps their file
space.

Syntax: UNLOAD,lfnl,lfn2,...,lfnn. <-- specified files
UNLOAD,*,lfnl,lfn2,...,lfnn. <-- all but
specified files

Similar commands: NOS/VE: RELEASE_RESOURCE

Examples: UNLOAD,myfile.

UNLOCK Rescind the LOCK command and clear the write interlock for
specified local disk files.

Syntax: UNLOCK,lfnl,lfn2,...,lfnn.

Remarks: Library files cannot be unlocked.

See also: LOCK

Examples: See LOCK.

UPDATE Create, edit or copy a program library.

Syntax: UPDATE,p1,p2,...,pn.

Parameters: Note: file parameters (C, G, I, K, N, O, S, T)
may be followed by 6 (6-bit display code)
or 8 (7-bit ASCII -- also requires I8)

pi	meaning
A	copy old sequential PL to new random access PL
B	copy old random access PL to new sequential PL
C=lfm	write compile file on lfm
C=PUNCH	implies D and 8 parameters
C	same as C=COMPILE
omitted	same as C=COMPILE
C=0	no compile output (Note: C is ignored if K is used)
D	compile file is 80 characters
omitted	compile file is 72 characters
E	edit the old PL (to completely edit, use E on two UPDATE commands -- the first will rearrange the directory and remove

purged identifiers -- the second will remove identifiers appearing only in the file's directory

F full UPDATE mode

G=lfm output file for PULLMOD directives
omitted appended to the S file

H=n one of:
3 - use 63-character set
4 - use 64-character set
omitted - use old PL char set

I=lfm primary input directive file
I=0 no input directives
omitted same as I=INPUT
(Note: 6/12_bit ASCII if INPUT is connected, else 6-bit display code)

K=lfm compile file with decks in order of
COMPILE directives
K same as K=COMPILE
omitted no input *COMPILE directives

L=clc2...cn list options -- one or more of:
A - list deck names, correction set identifiers, COMDECK directives, definitions, compile file decks
F - same as L=A123456789 (not 0)
0 - no listing
1 - error lines
2 - active UPDATE directives
3 - notes on each line with changed status
4 - text lines
5 - active compile file directives
6 - active and inactive lines
7 - all active lines
8 - all inactive lines
9 - correction history of options 5, 7, 8
Defaults: L=A1234 (correction run)
L=A1 (copy)

M=lfm merge with old PL
M same as M=MERGE
omitted no merging
(Note: both libraries must have the same character set)

N=1fn	new program library
N	same as N=NEWPL
N=0	no new program library
omitted	same as N=0
	(Note: default character set is that of OLDPL (except if OLDPL is 6-bit display code and I file is 7-bit ASCII, then NEWPL is 7-bit ASCII))
O=1fn	listable output file
O=0	no listable output
omitted	same as O=OUTPUT
P=1fn/s1/s2/.../s7	the old program library
P	same as P=OLDPL
P=0	no old program library
omitted	same as P=OLDPL (si are secondary old PLs)
Q	quick mode (process only decks on COMPILE directives)
R=clc2..c4	files to rewind before and after C - compile file N - new PL P - old and merge PLs S - source and PULLMOD files
R	no rewind
omitted	same as R=CNPS (not merge PL)
S=1fn	output source file
S	same as S=SOURCE
S=0	no output source file unless T=1fn is specified
omitted	same as S=0
T=1fn	same as S=1fn, except that common decks are excluded (Note: takes precedence over S)
U	do not abort for fatal errors
W	the new PL is sequential
omitted	the new PL is random (except sequential on magnetic tape)
X	compile file is compressed
omitted	compile file is not compressed
8	compile file is 80-character lines
omitted	compile file is 90-character lines

*=char	master control character (any 6-bit octal value 01-50, 53-54)
omitted	master control character is *
/=char	comment control character (A-Z, 0-9, +-*/\$=)
omitted	master control character is / (Note: do not use a command abbreviation for <char> unless NOABBREV is in effect)

See also: Section 5-4

Similar commands: COS: UPDATE
 NOS/VE: EXTRACT_SOURCE_LIBRARY; SOURCE_CODE_UTILITY (SCU)
 VMS: CMS; LIBRARIAN

Examples: NOTE,uin./*COMPILE A,B,C
 UPDATE,I=uin.
 FTN5,I.
 RETURN,uin.
 EXIT.
 RETURN,uin.
 = = = = =
 /UPDATE. <-- interactive
 ? *c a,b,c <-- enter UPDATE directive(s)
 ? <CR> <-- end-of-file

UPROC Specify a user prologue to be executed each time you start a batch or interactive job.

Syntax: UPROC,FN=pfile.
 UPROC,pfile.

Parameters: pfile - an indirect permanent file containing the prologue
 0 - no longer execute a user prologue
 (Default: 0)

Remarks: LOGINPR is the preferred name.

UPROC is normally executed only once to let NOS know that <pfile> is to be executed at the start of each login.

If you purge your prologue file, be sure to execute "UPROC." or "UPROC,0." to prevent future batch jobs from aborting.

See also: page 1-4-1

sIMILAR commands: NOS/VE: existence of your file PROLOG
 VMS: existence of your file LOGIN.COM

Examples: /FSE,loginpr.
 < create your procedure >
 ?? QR <-- save the file
 /UPROC,loginpr.

USER Identify you and provide validation information for each batch job.

Syntax: USER,username,password

Parameters: username - your 4-character User Initials
 password - your 4- to 7-character password

Remarks: This must immediately follow your job statement.

Similar commands: COS: ACCOUNT

Examples: USER,xxxx,mypw.

VERIFY Compare files in binary mode.

Syntax: VERIFY,lfn1,lfn2,p1,p2,...,pn.

Parameters: lfn1 - first file to be compared
 (default: TAPE1)

 lfn2 - second file to be compared
 (default: TAPE2)

 A - abort after completion if errors

 BS=bs - maximum block size for S, L tapes
 (defaults: 1000B words (S), 2000B (L))

 C - both files coded (S, L tapes only)

 C1 - first file coded (S, L tapes only)

 C2 - second file coded (S, L tapes only)

 E=y - maximum number of errors to list
 (default: 100)

 E - same as E=0

 L=lfn - error output file
 (default: OUTPUT)

 N=x - number of files of multi-file file
 (default: 1)

 N - compare until EOF on both files

 N=0 - compare until empty file in either file

 R - rewind both files before and after

Remarks: On mismatch, the record number, word within the record, and the words from both files which do not match are listed.

Your terminal should be in NORMAL mode (not ASCII) before listing VERIFY output at your terminal.

See also: VFYLIB

Similar commands: COS: COMPARE
NOS/VE: COMPARE_FILES

Examples: VERIFY,fyl1,fyl2,N,R.

VFYLIB List differences in name, type, length, and checksum for the records of two library files.

Syntax: VFYLIB,lfn_1,lfn_2,lfn_3,NR.

Parameters: lfn_1 - first file
lfn_2 - second file
lfn_3 - the listable output file
NR - do not rewind lfn_1 and lfn_2 after processing

Defaults: VFYLIB,OLD,NEW,OUTPUT.

Remarks: lfn_1 and lfn_2 are rewound before comparing.

Your terminal should be in NORMAL mode (not ASCII) before listing VFYLIB output at your terminal.

See also: VERIFY (binary comparison)

Similar commands: COS: ITEMIZE
NOS/VE: COMPARE_OBJECT_LIBRARY
VMS: DIFFERENCES

Examples: VFYLIB,fyl1,fyl2,out.

VSN Associate a local file name with one or more volume serial numbers.

Syntax: VSN,lfn_1=vsn_1,lfn_2=vsn_2,...,lfn_n=vsn_n.

Parameters: lfn_i - local file name

`vsni` - 1- to 6-character vsn's to be associated
with `lfni`
(\$-delimited if any non-alphanumerics)

<code>vsni</code>	meaning
omitted	any available scratch tape is assigned automatically
0	same as omitted
SCRATCH	same as omitted
<code>vsna=vsnb=...=vsnn</code>	duplicate volumes (any may be used)
<code>vsna/vsnb/.../vsnn</code>	successive volumes (must be used in listed order)

See also: LABEL

Examples: VSN,tape=NA9876.

WHATJSN (IAF) Get the job sequence number for the specified user name.

Syntax: WHATJSN,username

Parameters: username - the username whose jsn is desired

Remarks: You must be in the ACCESS subsystem.

See also: WHO

Similar commands: VMS: SHOW USERS

Examples: WHO

WHILE Start of a command loop.

Syntax: WHILE,exp,label.

Parameters: exp - any valid expression evaluating to true
or false

label - alphanumeric string (1-10 characters,
starting with a letter)

Remarks: The loop ends with an ENDW statement.

See also: ENDW

Similar commands: COS: LOOP
NOS/VE: WHILE

Examples: WHILE,R1<5,DOIT.

```
...
SET,R1=R1+1.
ENDW,DOIT.
```

WHO (IAF, DTRC) List the users currently logged in.

Syntax: WHO,lfm.

Parameters: lfm - output file
(default: OUTPUT)

Remarks: The display shows the total number of users who have logged in since the last system deadstart, the number of active (logged in) users (the "D" suffix indicates a decimal number), and a table showing the IAF connection number, User Initials, Job Sequence Number, and port number for each logged-in user. An asterisk in the W column indicates the user has been sent a Warning message by the operator but has not yet received it (messages from the operator while you are in FSE are not received until you exit from FSE).

Similar commands: NOS/VE: WHO (DTRC)
VMS: SHOW USERS

Examples: WHO,whoout. <-- write the display in local file
WHOOUT

= = = = =

WHO <-- display at the terminal
TOTAL USERS = 36D ACTIVE USERS = 5D

CONN	USER	JSN	W	TERM
3	CARA	AADD		T1200
4	TLIB	AAFH		T1210
5	CASG	AAFW		T1230
6	CTSC	AAFX		T1240
7	AMDS	AAFZ		T1610

WHO COMPLETE.

WHOAMI (DTRC) Display your user ID and account number.

Syntax: BEGIN,WHOAMI.

Similar commands: NOS/VE: DISPLAY_JOB_ATTRIBUTE
VMS: MYACCOUNT

Examples: BEGIN,WHOAMI.

WRITEF Write a specified number of file marks on a file.

Syntax: WRITEF,lfn,x.

Parameters: lfn - the file to receive the file marks

x - decimal number of file marks to write
(default: 1; max: 262143)

Remarks: If previous write was not an EOR, an EOR is added before the EOF.

Use WRITEF after FCOPY when creating a multi-file stranger tape.

See also: WRITER

Examples: WRITEF,myfile,2.

WRITER Write a specified number of empty records on a file.

Syntax: WRITER,lfn,x.

Parameters: lfn - the file to receive the empty records

x - decimal number of empty records to write
(default: 1; max: 262143)

See also: WRITEF

Examples: WRITER,myfile,3.

X (IAF) Execute a batch command.

Syntax: X,ccc

Parameters: ccc - any valid batch command (up to 80 chars)

Examples: X,BASIC <-- compile a BASIC program without
changing to the BASIC subsystem

X,BASIC Compile a BASIC program without changing to the BASIC subsystem.

Syntax: X,BASIC,I=lfn,B=lfn.

Parameters: I= - the BASIC source program
omitted - same as I=INPUT

B= - the output object module
omitted - execute without creating object module

```

Examples:  X,BASIC,I=mybas.
              ^-- compile and execute (do not
              create an object module)
          = = = = =
          X,BASIC,I=mybasi,B=mybaso.      <-- compile
          mybaso.                          <-- execute

```

```
Parameters: lfn      - the local file to be sent

              copies - number of copies
                      (default: 1)

              job     - Xerox 8700 job name
                      (default: STDLND)

              paper   - 3-HOLE or PLAIN
                      (default: 3-HOLE)

              duplex  - YES (2-sided) or NO (1-sided)
                      (default: YES)

              forms   - forms overlay name
                      (default: NONE)

              ascii   - YES (6/12-bit display code)
                      NO  (6/6-bit  display code)
                      8   (8/12-bit ASCII)
                      (default: NO)

              shift   - 0 - carriage control is in column 1
                      1 - single-spaced listing
                      (default: 0)

              ujn     - change the "user job name" field on the
                      banner page (1-7 characters; xxxxxxxx)
                      (default: your user initials)
```

Similar commands: NOS/VE: XEROX (DTRC)
VMS: XEROX, XEROXD, XEROXF, XEROXFD
(all DTRC)

Examples: `HELPME.XMODEM.`

*** Interactive Cray Facility Commands ***

The Interactive Cray Facility (ICF) provides the CDC NOS user with access to the CRAY X-MP.

The following are the ICF commands:

/ABORT Abort an interactive Cray job.

Syntax: /ABORT (/AB)

Remarks: May also use USER-BREAK-2 key.

Similar commands: VMS CRAY context: ABORT, DROP, KILL

Examples: /AB

/ATTENTION Send an attention interrupt to the interactive Cray job.

Syntax: /ATTENTION (/AT)

Remarks: May also use USER-BREAK-1 key.

Similar commands: VMS CRAY context: ATTENTION

Examples: /AT

/BYE Terminate the interactive Cray session.

Syntax: /BYE HOLD AP=NAM_application (/B)
/BYE QUIT AP=NAM_application

Parameters: same as /LOGOFF

Remarks: Equivalent to LOGOFF.

Similar commands: VMS CRAY context: BYE, QUIT

Examples: /BYE

/CONNECT Logically connect to another terminal (such as a plotter).

Syntax: /CONNECT terminal_name (/C)

Parameters: terminal_name

See also: /ENDCONNECT, /ICFSTATUS

Similar commands: VMS CRAY context: ATTACH

/DISCARD Discard all output sent to the terminal.

Syntax: /DISCARD (/D)

Similar commands: VMS CRAY context: DISCARD

Examples: /D

/ENDCONNECT Terminate the logical connection between a master and slave terminal.

Syntax: /ENDCONNECT (/ENDC)

See also: /CONNECT

Examples: /ENDC

/ENDPLAY Terminate reading from a PLAY file.

Syntax: /ENDPLAY (/ENDP)

See also: /PLAY

Examples: /ENDP

/EOF Send an end-of-file to COS.

Syntax: /EOF (/EO)

Similar commands: VMS CRAY context: EOF

Examples: /EOF

/HELP Display a brief description of an interactive Cray command.

Syntax: /HELP command_name (/H)

Parameters: command_name - command for which help is
 requested
 (default: a list of all inter-
 active commands and
 their parameters)

Similar commands: VMS CRAY context: HELP

Examples: /H EOF or

/ICFSTATUS Display the status of stations and terminals connected to ICF.

Syntax: /ICFSTATUS (/I)

Similar commands: VMS CRAY context: ISTATUS

Examples: /I

/LOGOFF Terminate an interactive Cray session.

Syntax: /LOGOFF HOLD AP=NAM_application (/LOGOF)
/LOGOFF QUIT AP=NAM_application

Parameters: HOLD - suspend the interactive session

QUIT - quit the interactive session
(default: QUIT)

AP= - the next application

Remarks: same as /BYE

See also: /LOGON

Similar commands: VMS CRAY context: BYE; QUIT

Examples: /LOGOFF
/LOGOFF HOLD AP=ICF

/LOGON Start an interactive Cray session (or reconnect to an existing session).

Syntax: /LOGON MF=mf

Parameters: MF= - the Cray mainframe
mf meaning

MCR CRAY X-MP at DTRC

Remarks: LOGON is not allowed if you are already logged on.

At the exclamation prompt (!), enter an ACCOUNT statement.

See also: /LOGOFF

Similar commands: VMS CRAY context: INTERACTIVE
VMS: CINT; CRAY INTERACTIVE

Examples: /LOGON

/PERIOD Control automatic addition of a period terminator to Cray commands.

Syntax: /PERIOD ON (/PE)
 /PERIOD OFF

Parameters: ON - ICF supplies the terminating period on
 Cray commands
 OFF - you must supply the terminating period
 (default: OFF)

Examples: /PE ON

/PLAY Read commands and data from a NOS file.

Syntax: /PLAY filename NOECHO (/PL)

Parameters: filename - a NOS 8/12-bit ASCII file
 NOECHO - N - do not echo the lines as they are
 read
 (default: echo the lines)

Remarks: /ENDPLAY

Similar commands: VMS CRAY context: @, PLAY

Examples: /PLAY myascii

/PREFIX Change the ICF prefix character.

Syntax: /PREFIX prefix_character (/PR)

Parameters: prefix_character - the new ICF command prefix
 character

Remarks: The default prefix character is slash (/).

To restore to the default, use "pPREFIX=/",
where "p" is the current prefix character.

Similar commands: VMS CRAY context: none

Examples: /PR=~ <-- change to tilde
 ~ST <-- display status
 ~PR=/ <-- restore to slash
 /ST <-- display status

/QUIT Immediately terminate the interactive Cray session.

Syntax: /QUIT HOLD AP=NAM_application (/Q)
/QUIT QUIT AP=NAM_application

Parameters: same as /LOGOFF

Remarks: Can also use LOGOFF or BYE.

Similar commands: VMS CRAY context: BYE; QUIT

Examples: /QUIT

/RESUME Resume a suspended interactive Cray session.

Syntax: /RESUME (/R)

See also: /SUSPEND

Examples: /R

/STATUS Display Cray job status.

Syntax: /STATUS (/ST)

Similar commands: VMS CRAY context: ISTATUS; JSTAT; STATUS

Examples: /ST

/SUSPEND Suspend an interactive Cray session.

Syntax: /SUSPEND (/SU)

See also: /RESUME

Examples: /SU

/* An ICF comment line.

Syntax: /* comment

Parameters: comment - optional text

Similar commands: VMS CRAY context: COMMENT, MESSAGE

Examples: /* This is a comment

***** Glossary *****

Alphabetic (CDC - NOS)

The letters A-Z.

Alphabetic (CDC - NOS/VE)

The letters A-Z, a-z.

Alphabetic (Cray - COS)

\$. %, @, and the letters A-Z, a-z.

Alphabetic (DEC)

\$. _ (underscore), and the letters A-Z, a-z (upper and lower case are the same).

Alphanumeric

Alphabetic and the digits 0-9.

Dual-state

The state in which two operating systems operate simultaneously in the same mainframe. At DTRC, the CDC CYBER 860 operates in dual-state with NOS and NOS/VE.

User initials (userid or username)

The 4-character ID assigned to each user by Code 3502. This is used to identify jobs, for charge authorization, to identify permanent and MSS files, magnetic tapes, etc.

***** Index *****

Note - NOS/BE system control statements are flagged with ".
Intercom commands are flagged with @.
UPDATE directives begin with *.
Compiler options are flagged with \$.

Primary references are flagged with an asterisk after the page number, for example, 1-1*.

* (comment)	9-2-2, 9-3-4*, I-4*, I-138*
%A	8-2-5
ABBREV	9-4-3*
Abbreviation	H-1
Abort	I-3, I-17, I-134
ABORT (ICF)	I-134*
Absolute	9-5-1, 9-6-1
ACCEPT_LINE	H-108*
Access	8-1-2, 9-1-2, H-120, H-124
ACCESS	9-2-4, I-5*, I-35, I-129
Access control entry	H-52, H-54
Access control list	H-57, H-59
Access mode	I-2*
ACCL	H-108*
Account number	8-1-2, I-19, I-130
Account number, default	I-19
Accounting	I-102
ADD	9-5-2*
ADDFILE	9-4-2*
ADD_LIBRARY	8-6-1
ADD_MODULE	8-5-2, 8-7-1*
Alphabetic	H-2, G1-1
Alphanumeric	H-2, I-119, G1-1
ANALYZE_OBJECT_LIBRARY	8-2-11, H-9*
ANALYZE_PROGRAM_DYNAMICS	8-2-11, H-10*
ANAOL	H-9*
ANAPD	H-10*
ANYM	H-10*
ANY_FAULT	H-3*
ANY_MAIL	8-2-11, 8-5-3, H-10*

APPEND	9-2-8, I-6*
Application	I-15, I-60, I-75
Application, NAM	I-6
APPLLIB	9-7-1
APPSW	9-2-5, I-6*
Arrow	9-1-5
ASCII	9-2-4, H-63, I-7*, I-98
ASCII8	I-29, I-122
Assign	I-8
ASSIGN	9-2-6, 9-2-8, 9-2-9, I-7*
Asynchronous	H-163, H-165
Attach	H-11
ATTACH	9-2-8, 9-7-1, I-8*, I-56
ATTACH_FILE	H-11*
ATTACH_JOB	H-11*
ATTENTION (ICF)	I-134*
ATTF	H-11*
ATTJ	H-11*
Attribute	H-17, H-18, H-20, H-22, H-23, H-25, H-26, H-28, H-63, H-66, H-68, H-69, H-71, H-75, H-77, H-78, H-79, H-80, H-122, H-147, H-149, I-103
Attribute, file	H-122, I-45
Attribute, function	8-5-1
Attribute, procedure	8-5-1, H-135
Attrubute	H-64
Audit	H-59, H-120, I-16, I-119
Auto-drop	I-103
AUX	9-2-6, 9-3-5, I-8*
Auxiliary printer	H-12, I-8, I-9
AUXP	H-12*
AUXPRINT	8-2-14, 8-5-3, H-12*
AUXPRNT	9-2-6, 9-3-5, I-9*
Backspace	I-12
Backup	H-12, H-15, H-57, I-91
BACKUP_PERMANENT_FILES	8-2-8, H-12*
Backward	H-153
BACPF	H-12*
Banner	I-14
BASIC	8-2-13, 9-2-4, 9-2-10, H-12*, I-10*, I-131
Batch	8-1-5*, 9-1-7*, H-3, H-5, H-7, H-45, H-112, H-116, H-117, H-162, I-64,

Batch	I-131
BATCH	9-2-4, I-10*
Batch editor	I-51, I-123
Batch job classes	8-1-6*, 9-1-8*
Batch jobs, killing	8-1-6*, 9-1-8*
BEFORE	9-4-2*, 9-5-2*
BEGIN	9-2-1, 9-2-10, I-10*
Beginning-of-information	I-97
BELOAD	I-11*
BETONOS	9-7-1
Binary	H-4, H-33, I-24, I-36, I-63
Binary mode	I-127
BINARY_OBJECT	H-4*
BKSP	9-2-6, I-12*
BLANK	9-2-9, I-13*
Blank common	9-6-4
Block	9-1-6*, H-3, H-55, H-81
BLOCK	8-2-1, 8-6-3, 9-2-2, H-13*, I-14*
Block data	9-6-5
Block letters	I-14
BLOCKEND	8-6-3, H-13*
BOI	I-97
Boolean	H-2
Break	H-163
BUILD	9-5-2*
BYE	9-2-5, I-15*
BYE (ICF)	I-134*
Cache memory	8-1-1, 9-1-1
CALCFN	9-7-2
Calcomp	I-83
CALC936	9-7-2
CALL	9-4-2*, 9-4-5
Call by name	I-79
Cancel	I-4
CANCEL	8-2-1, H-3, H-14*
Capsule	9-6-2
Carderock	I-122
Case	H-25
Catalog	8-8-1, H-12, H-15, H-16, H-39, H-52, H-57, H-59, H-83, H-125
CATALOG	9-2-10, I-15*
Catalog, working	H-28, H-82
CATASTROPHIC	H-4, H-7

Category Type	I-2*
CATLIST	9-2-8, I-16*
CAUSE	8-2-1, H-14*
Caution	9-6-6
CC (Proc)	9-3-1*
CCL	9-2-1
CDC	9-1-1*, I-1, G1-1
CDC CYBER 180	8-1-1*, 9-1-1
CDCnet	9-1-1
cdc860	8-1-2, 9-1-2
CDEFINE	I-17*
CDROP	I-17*
Central processing unit	8-1-1, 9-1-1
CHABLT	H-15*
CHACA	H-17*
CHACC	H-15*
CHACE	H-16*
CHACSM	H-16*
CHAFa	H-17*
CHAIa	H-18*
CHAI5	H-19*
CHAJa	H-20*
CHAJL	H-21*
CHALA	H-22*
CHALP	H-22*
CHAML	H-23*
change	I-137
Change	I-18, I-31, I-86, I-120
CHANGE	9-2-8, 9-4-2*, I-18*
CHANGE_BACKUP_LABEL_TYPE	8-2-10, H-15*
CHANGE_CATALOG_CONTENTS	8-2-8, H-15*
CHANGE_CATALOG_ENTRY	8-2-8, H-16*
CHANGE_COMMAND_DESCRIPTION	8-7-2
CHANGE_COMMAND_SEARCH_MODE	8-2-2, 8-8-2, H-16*
CHANGE_CONNECTION_ATTRIBUTES	8-2-4, H-17*
CHANGE_DECK	8-6-1
CHANGE_DECK_NAMES	8-6-1
CHANGE_DECK_REFERENCES	8-6-1
CHANGE_FILE_ATTRIBUTES	8-2-8, H-17*
CHANGE_FUNCTION_DESCRIPTION	8-7-2
CHANGE_INPUT_ATTRIBUTE	8-2-2, H-18*
CHANGE_INTERACTION_STYLE	8-2-4, H-19*
CHANGE_JOB_ATTRIBUTE	8-2-2, H-20*
CHANGE_JOB_LIMIT	8-2-2, H-21*
CHANGE_LIBRARY	8-6-1

CHANGE_LINK_ATTRIBUTES	8-2-4, H-22*
CHANGE_LOGIN_PASSWORD	8-2-2, H-22*
CHANGE_MESSAGE_LEVEL	8-2-2, H-23*
CHANGE_MODIFICATION	8-6-1
CHANGE_MODULE_ATTRIBUTE	8-7-2*
CHANGE_OUTPUT_ATTRIBUTE	8-2-2, H-23*
CHANGE_PROGRAM_DESCRIPTION	8-7-4*
CHANGE_SCL_OPTION	8-2-2, H-24*
CHANGE_SCL_OPTIONS	H-78
CHANGE_TAPE_LABEL_ATTRIBUTES	H-25*
CHANGE_TERMINAL_ATTRIBUTES	8-2-4, H-26*
CHANGE_TERM_CONN_ATTRIBUTES	H-17*
CHANGE_TERM_CONN_DEFAULTS	8-2-4, H-27
CHANGE_UNSEEN_MAIL_ACTION	H-27*
CHANGE_UTILITY_ATTRIBUTES	8-2-2, 8-2-11, H-28*
CHANGE_WORKING_CATALOG	8-2-7, H-28*
CHANGE_170_REQUEST	8-2-10, H-14*
CHAOA	H-23*
Character, inhibit	9-3-2
Character, master	9-4-1
Character, prefix	I-137
Character set	I-31, I-42
Characteristic	I-18, I-120
Charge	8-1-4, 9-1-6, I-80
CHARGE	9-1-2, 9-2-2, I-19*
Charge number	9-1-2, I-17
Charges	H-139
CHASCLO	H-24*
CHASO	H-24*, H-78
CHATA	H-26*
CHATCA	H-17*
CHATLA	H-25*
CHAUA	H-28*
CHAUMA	H-27*
CHAWC	H-28*
CHA1R	H-14*
Checkpoint	9-2-9, I-20, I-94, I-95
CHECK_FOR_MAIL	8-2-11, H-29*
CHEFM	H-29*
Christensen protocol	I-133
CHVAL,CN	9-1-2, 9-2-2, I-19*
CJOB	I-20*
CKILL	I-20*
CKP	9-2-9, I-20*
CLEAR	9-2-6, I-20*

Clear screen	9-3-2
Clock	I-99
CM	I-2*
Cobol	9-6-8, I-21
COBOL	8-2-13, H-29*
COBOL5	9-2-10, I-21*
Code, condition	8-4-4
Code, disposition	I-17
Code, forms	I-98
Coded	I-25, I-100
COLLECT_TEXT	8-2-7, 8-5-1, 8-8-2, H-32*
COLT	H-32*
COM	H-97
Combine	I-86
COMBINE_LIBRARY	8-6-1
COMBINE_MODULE	8-5-2, 8-7-5*
COMDECK	9-4-1*, 9-4-5
COMF	H-33*
Command	8-8-1, H-1, H-60, H-88, H-92, I-1*, I-4, I-40, I-131, I-134
COMMAND	8-2-2, H-32*
Command environment	H-105
Command file	I-137
Command help	8-4-1
Command list	8-8-1, 8-8-2, H-16, H-41, H-53, H-61, H-62, H-138
Command list entry	8-8-1, 8-8-2
Command stream	H-110, H-111
Command table	H-105, H-163
Command table, utility	H-32
Command utility	H-166
COMMAND_FAULT	H-3*
COMMAND_LIST	H-131, H-137
COMMD	H-33*
Comment	9-3-4, 9-4-4, I-4, I-82, I-138
COMMENT	9-2-2, 9-5-2*, I-22*
Common	9-6-5
Common, blank	9-6-4
Common block	9-6-6
COMOL	H-33*
Compare	H-33, H-105, I-127
COMPARE_FILE	8-2-7, H-33*
COMPARE_OBJECT_LIBRARY	8-2-12, H-33*
Compile	H-12, H-29, H-100, H-129, I-21, I-48, I-52, I-131

COMPILE	9-4-2*, 9-4-6
COMPRESS_MAIL_DATABASE	8-2-11, H-33*
Computer rates	I-90
Concatenate	9-3-1
Condition	H-14, H-168
Condition code	8-4-4
Condition, damage	H-15
Condition handler	H-3
Conditional	9-3-2, I-62
Conditional execution	H-110, H-169
Conect	H-43
Connect	H-54, H-63, H-65, H-80
CONNECT (ICF)	I-134*
Connection, interstate	H-45, H-55
CONTENT	9-3-5
Continuation	9-6-6
CONTINUE	8-2-1, H-3*, H-34*
Control Data Corporation	8-1-1, 9-1-1*
Control entry, access	H-52, H-54
Control list, access	H-57, H-59
Control sequence	8-4-7
Control statement	8-5-1, 8-8-2, 9-3-1
Control, transfer	H-92
CONUTS	H-34*
Convert	H-34, I-22, I-42, I-56, I-89
CONVERT_UPDATE_TO_SCU	8-2-11, 8-2-12, 8-6-3, H-34*
COPF	H-35*
Copies	I-98
Copy	H-32, H-35, H-106, H-123, H-140, I-22, I-24, I-25, I-26, I-28, I-29, I-48, I-100, I-117
COPY	8-6-3, 9-2-6, 9-4-2*, 9-4-3*, 9-5-2*, I-22*
COPYBF	9-2-6, I-24*
COPYBFR	9-2-6, 9-3-5, I-24*
COPYBR	9-2-6, I-24*
COPYC	8-6-3
COPYCF	9-2-6, I-25*
COPYCR	9-2-6, I-25*
COPYEI	9-2-6, I-26*
COPYL	9-2-10, I-26*
COPYLM	9-2-10, I-27*
COPYSBF	9-2-6, I-28*
COPYSF8	9-2-6, I-29*
COPYX	9-2-6, I-29*

COPY_FILE	8-2-7, H-35*
CORE	9-6-8
Core image	9-6-1
CORRECT (Proc)	9-3-1*
COS	G1-1
Cost	H-139, I-65
CPU	8-1-1, 9-1-1
Cray	H-160, H-161, I-17, I-20, I-30, I-31, I-32, I-134, G1-1
Cray station	I-17
Create	9-3-1, 9-4-5, 9-5-3, I-34, I-71, I-121, I-123
Create a file	I-82
CREATE_BRIEF_HELP_MODULE	8-4-2*
CREATE_CATALOG	8-2-8, H-39*
CREATE_CATALOG_PERMIT	8-2-8, H-39*
CREATE_COMMAND_DESCRIPTION	8-7-6
CREATE_COMMAND_LIST_ENTRY	8-2-2, 8-8-2, H-41*
CREATE_DECK	8-6-1
CREATE_DEFAULT_VARIABLE	8-2-13, H-42*
CREATE_FILE	8-2-8, H-42*
CREATE_FILE_CONNECTION	8-2-7, H-43*
CREATE_FILE_PERMIT	8-2-8
CREATE_FILE_VARIABLE	8-2-13, H-44*
CREATE_FORM_MODULE	8-7-6*
CREATE_FULL_HELP_MESSAGE	8-4-2*
CREATE_FUNCTION_DESCRIPTION	8-7-7
CREATE_INTERSTATE_CONNECTION	8-2-4, H-45*
CREATE_LIBRARY	8-6-1
CREATE_LINKED_MODULE	8-7-7*
CREATE_MANUAL	8-2-11, H-46*
CREATE_MESSAGE_MODULE	8-4-1*
CREATE_MODIFICATION	8-6-1
CREATE_MODULE	8-7-7*
CREATE_OBJECT_LIBRARY	8-2-12, 8-4-1, 8-5-2*, 8-7-1*, H-25, H-46*
CREATE_PARAMETER_ASSIST_MESSAGE	8-4-3*
CREATE_PARAMETER_HELP_MESSAGE	8-4-3*
CREATE_PARAMETER_PROMPT_MESSAGE	8-4-4*
CREATE_PROGRAM_DESCRIPTION	8-7-8*
CREATE_PROGRAM_PROFILE	8-2-12, H-47*
CREATE_STATUS_MESSAGE	8-4-4*
CREATE_TOPICS_FILE	8-2-11, H-48*
CREATE_VAX_REQUEST	8-2-10, H-48*
CREATE_170_REQUEST	8-2-10, H-36*

CREC	H-39*
CRECLE	H-41*
CRECP	H-39*
CREDV	H-42*
CREF	H-42*
CREFC	H-43*
CREFV	H-44*
CREIC	H-45*
CREM	H-46*
CREMM	8-4-1*
CREOL	8-5-2*, 8-7-1*, H-25, H-46*
CREPP	H-47*
CRERUN	I-30*
CRETf	H-48*
CREVR	H-48*
CRElR	H-36*
CSET	9-2-4, I-31*
CSTATUS	I-31*
CSUBMIT	9-2-2, I-32*
CT	I-2*
CTASK	I-32*
ctD	I-5*
ctE	I-5*
CTIME	9-2-2, I-33*
ctS	I-5*
ctl	I-4*
ct2	I-4*
Cursor	9-1-5
CWEOF	9-4-2*
CYBER control language	9-2-1
CYBER 860	9-1-1, G1-1
Cycle	H-15, H-52, H-53, H-137
CYCLE	8-2-1, H-49*
%D	8-2-5, 9-2-4, I-5*
DA	9-6-8
Damage condition	H-15
Data	H-166
DATA (Proc)	9-3-1*, 9-3-2
Date	H-2, I-33
DATE	9-2-12, 9-5-2*, I-33*
Dayfile	9-3-2, I-22, I-33, I-36, I-99, I-111
DAYFILE	9-2-2, I-33*
Debug	H-4, H-146, H-147

Debug library list	H-146
DEBUG_MODE	H-4*
DEC	G1-1
DEC VT-100	9-1-3, 9-3-2
DECF	H-50*
Decimal	H-2
Deck	9-4-1, H-84, H-93, H-96, H-105
DECK	8-6-3, 9-4-1*, 9-4-5
Decrypt	H-50
DECRYPT_FILE	8-2-11, H-50*
DEC_VT100_GOLD	8-3-1
Default	H-15, H-42, H-69, H-147
Default account number	I-19
DEFINE	9-2-8, I-34*, I-93
DEFINE_PRIMARY_TASK	8-2-2, H-51*
DEFINE_TERMINAL	8-2-4, H-51*
Definition file, terminal	H-51
Definition, panel	I-87*
Definition, terminal	I-118, I-120
DEFT	H-51*
DELAC	H-52*
DELC	H-52*
DELCLE	H-53*
DELCP	H-52*
Delete	H-52, H-125, H-137
DELETE	9-4-2*, 9-4-6, 9-5-2*
DELETE_ALL_CYCLES	8-2-8, 8-5-3, H-52*
DELETE_CATALOG	8-2-8, H-52*
DELETE_CATALOG_PERMIT	8-2-9, H-52*
DELETE_COMMAND_LIST_ENTRY	8-2-2, 8-8-2, H-53*
DELETE_DECK	8-6-1
DELETE_FEATURE	8-6-1
DELETE_FILE	8-2-9, H-53*
DELETE_FILE_CONNECTION	8-2-7, H-54*
DELETE_FILE_PERMIT	8-2-9, H-54*
DELETE_GROUP	8-6-1
DELETE_INTERSTATE_CONNECTION	8-2-4, H-55*
DELETE_MODIFICATION	8-6-1
DELETE_MODULE	8-5-2, 8-7-9*
DELETE_VARIABLE	8-2-13, 8-5-1, 8-8-2, H-55*
DELF	H-53*
DELFC	H-54*
DELFP	H-54*
DELIC	H-55*
Delimit	H-7

DELV	H-55*
Density	I-66
Descriptor, value	H-2
Design	H-56
DESIGN_SCREEN	8-2-4, H-56*
DESS	H-56*
Destroy	I-85
Detach	H-56, H-57, I-5
DETACH_FILE	8-2-9, H-56*
DETACH_JOB	8-2-2, H-57*
DETF	H-56*
DETJ	H-57*
Diagram, tree	9-6-5*
DIAL	9-2-5, I-35*
Differences	I-128
Digit	G1-1
Direct file	9-1-6*, I-8, I-34, I-57
Directive	9-6-6
Directive, LIBEDIT	9-5-1*
Directive, reformatting	9-1-7*
Directive, UPDATE	9-4-1*
DISAT	H-57*
DISBLT	H-57*
DISC	H-57*
DISCA	H-63*
DISCARD (ICF)	I-135*
DISCE	H-59*
DISCI	H-60*
DISCL	H-61*
DISCLE	H-62*
DISCLO	H-78*
Disconnect	H-54, H-57
DISCP	H-60*
DISF	H-63*
DISFA	H-64*
DISFC	H-65*
DISFI	H-66*
DISIA	H-66*
DISIS	H-70*
DISJA	H-68*
DISJAD	H-69*
DISJL	H-70*
DISJS	H-70*
DISL	H-72*
DISLA	H-71*

DISM	H-73*
DISOA	H-75*
DISOH	H-76*
DISOL	H-73*
DISOS	H-77*
DISOT	H-74*
DISPA	H-77*
Display	8-1-3, 9-1-3, H-152, H-153
DISPLAY	9-2-1, I-36*
Display code	I-37
DISPLAY_ACTIVE_TASKS	8-2-5, H-57*
DISPLAY_BACKUP_LABEL_TAPE	8-2-10, H-57*
DISPLAY_CATALOG	8-2-9, H-57*
DISPLAY_CATALOG_ENTRY	8-2-9, H-59*
DISPLAY_COMMAND_INFORMATION	8-2-5, 8-8-2, H-60*
DISPLAY_COMMAND_LIST	8-2-5, 8-8-2, H-61*
DISPLAY_COMMAND_LIST_ENTRY	8-2-5, 8-8-2, H-62*
DISPLAY_COMMAND_PARAMETERS	H-60*
DISPLAY_CONNECTION_ATTRIBUTES	8-2-4, H-63*
DISPLAY_DECK	8-6-1
DISPLAY_DECK_LIST	8-6-1
DISPLAY_DECK_REFERENCES	8-6-2
DISPLAY_FEATURES	8-6-2
DISPLAY_FEATURE_LIST	8-6-2
DISPLAY_FILE	8-2-7, H-63*
DISPLAY_FILE_ATTRIBUTES	8-2-9, H-64*
DISPLAY_FILE_CONNECTIONS	8-2-7, H-65*
DISPLAY_FUNCTION_INFORMATION	8-2-11, H-66*
DISPLAY_GROUP	8-6-2
DISPLAY_GROUP_LIST	8-6-2
DISPLAY_INPUT_ATTRIBUTE	8-2-7, H-66*
DISPLAY_INPUT_STATUS	H-70*
DISPLAY_JOB_ATTRIBUTE	8-2-2, H-68*
DISPLAY_JOB_ATTRIBUTE_DEFAULT	8-2-5
DISPLAY_JOB_ATTRIBUTE_DEFAULTS	H-69*
DISPLAY_JOB_LIMIT	8-2-5
DISPLAY_JOB_LIMITS	H-70*
DISPLAY_JOB_STATUS	8-1-6, 8-2-5, H-70*
DISPLAY_LIBRARY	8-6-2
DISPLAY_LINK_ATTRIBUTES	8-2-4, H-71*
DISPLAY_LOG	8-2-2, H-72*
DISPLAY_MESSAGE	8-2-2, H-73*
DISPLAY_MODIFICATION	8-6-2
DISPLAY_MODIFICATION_LIST	8-6-2
DISPLAY_NEW_LIBRARY	8-5-2, 8-7-9*

DISPLAY_OBJECT_LIBRARY	8-2-12, H-73*
DISPLAY_OBJECT_TEXT	8-2-12, H-74*
DISPLAY_OUTPUT_ATTRIBUTE	8-2-2, H-75*
DISPLAY_OUTPUT_HISTORY	8-2-2, H-76*
DISPLAY_OUTPUT_STATUS	8-2-3, H-77*
DISPLAY_PRINT_STATUS	H-77*
DISPLAY_PROGRAM_ATTRIBUTES	8-2-3, H-77*
DISPLAY_SCL_OPTIONS	8-2-3, H-78*
DISPLAY_TAPE_LABEL_ATTRIBUTES	8-2-10, H-78*
DISPLAY_TASK_STATUS	8-2-3, H-79*
DISPLAY_TERMINAL_ATTRIBUTES	8-2-4, H-79*
DISPLAY_TERM_CONN_ATTRIBUTES	H-63*
DISPLAY_TERM_CONN_DEFAULTS	8-2-4, H-80*
DISPLAY_TOPICS_FILE	8-2-5, H-80*
DISPLAY_VALUE	8-2-14, 8-5-1, 8-8-2, H-80*
DISPLAY_VARIABLE_LIST	8-2-14, 8-5-1, 8-8-2, H-81*
DISPLAY_WORKING_CATALOG	8-2-9, H-82*
Dispose	I-98
Disposition code	I-17
DISPS	H-77*
DISSO	H-78*
DISSPLA	9-7-2
DISTA	H-63*, H-79*
DISTCD	H-80*
DISTF	H-80*
DISTLA	H-78*
DISTS	H-79*
DISV	H-80*
DISVL	H-81*
DISWC	H-82*
DMB	9-2-8, I-36*
DMD	9-2-8, I-37*
DMP	9-2-8, I-37*
Dollar sign	8-5-1
DROP	9-1-8, 9-2-2, I-37*
DTLIB	8-7-14*, 9-5-5, 9-7-1
DTNET	9-1-9
DTRC	8-5-3, 8-7-14, 9-3-5, 9-5-5, H-9
DT100	9-1-5, I-101
Dual-state	8-1-1, 9-1-1, H-22, H-45, G1-1*
Dump	9-2-8, H-63, I-20, I-36, I-37, I-112, I-114, I-119
Dump, Post Mortem (PMD)	9-6-8
DUMPF	I-11
Duplex	I-120

Dynamic load

9-6-2

%E	9-2-4, I-5*
Echo	H-26, I-120
ECHO	8-2-5, 8-5-3, H-82*
EDIC	H-83*
EDID	H-84*
EDIF	H-84*
EDIL	H-84*
Edit	H-83, H-84, H-105, H-152
Editor (batch)	I-51, I-123
Editor (FSE)	9-2-10, I-51
EDIT_CATALOG	8-2-9, H-83*
EDIT_DECK	8-2-12, 8-6-2, H-84*
EDIT_FILE	8-2-11, 8-3-1*, H-84*
EDIT_FILE (editor)	8-1-3
EDIT_LIBRARY	H-84*
ELSE	8-6-3, 9-2-1, H-110*, I-38*, I-39
ELSE (Proc)	9-3-1*, 9-3-1
ELSEIF	8-6-3, H-110*
EMA	H-85*
EMAIL	8-2-11, H-85*
Empty	I-131
ENCF	H-86*
Encrypt	H-86
ENCRYPT_FILE	8-2-14, H-86*
END	9-6-4
ENDCONNECT (ICF)	I-135*
ENDHELP (Proc)	9-3-1*
ENDIF	9-2-1, I-38, I-39*, I-105
ENDIF (Proc)	9-3-1, 9-3-1*
ENDPLAY (ICF)	I-135*
ENDTEXT	9-4-3*
ENDW	9-2-1, I-39*
End-of-file	9-3-2, I-135
End-of-information	I-26, I-105
End-of-record	9-3-2
END_LIBRARY	8-6-2
END_MESSAGE_MODULE	8-4-6*
ENQUIRE	9-2-2, I-39*, I-69
ENTER	9-2-2, I-40*
ENTER (Proc)	9-3-2*
ENTER_FILE_MANAGER	8-2-7, H-87*
ENTER_PROGRAMMING_ENVIRONMENT	8-2-14, H-87*

ENTFM	H-87*
ENTPE	H-87*
Entry	H-16, H-41, H-53, H-62, H-138
Entry, access control	H-52, H-54
Entry point	9-6-3, I-41
Environment	H-42, H-44, H-131, H-137
Environment, command	H-105
Environment, file migration	H-128
Environment, programming	H-87
EOF	I-86, I-100, I-105
EOF (ICF)	I-135*
EOF (Proc)	9-3-2*
EOI	I-26, I-105
EOR	I-86, I-100
EOR (Proc)	9-3-2*
EP	I-120
Erase	I-85
ERRMSG	9-2-2, I-41*
Error	H-3, H-4, H-7, I-81
ERROR	H-4*
Error processing	8-4-1
ERROR_LEVEL	H-4*
Evaluate	I-36
Event	H-168
EVICT	I-41*
EX (Proc)	9-3-2*
Exchange	I-37
Exchange package	I-36, I-37
EXCLUSIVE (search)	8-8-2
EXEC	H-88*
Execute	H-10, H-88, H-89, I-37, I-69, I-91, I-131
EXECUTE	9-2-4, 9-2-11, 9-6-3, I-41*
Execute-only	9-7-1
EXECUTE_COMMAND	8-2-3, 8-8-1, H-88*
EXECUTE_INTERSTATE_COMMAND	8-2-3, H-88*
EXECUTE_TASK	8-2-13, 8-8-1, H-89*
Execution	8-8-1*, 9-3-2
Execution, conditional	H-110, H-169
Execution, object program	9-6-8*
Execution, repetitive	H-99, H-117, H-140, H-169
EXEIC	H-88*
EXET	H-89*
EXIT	8-2-1, 9-2-1, H-3*, H-92*, I-41*
EXP	H-94*

EXPAND (Proc)	9-3-2*
EXPAND_DECK	8-6-2
EXPAND_FILE	8-6-2
EXPAND_SOURCE_FILE	8-2-12, 8-6-3, H-93*
EXPLAIN	8-2-5, 9-2-5, H-46, H-94*, I-42*
EXPLAIN_MESSAGE	8-2-5, H-95*
EXPM	H-95*
Expression	H-80
EXPSF	H-93*
External	I-72
Externals, unsatisfied	I-99
Extract	I-59
EXTRACT_DECK	8-6-2
EXTRACT_MODIFICATION	8-6-2
EXTRACT_SOURCE_LIBRARY	8-2-12, 8-6-3, H-96*
EXTSL	H-96*
FAMILY	H-4*
FATAL	H-4, H-7
Fault, any	H-3
Fault, command	H-3
Fault, limit	H-3
Fault, program	H-3
FCOPY	9-2-6, I-42*
Fetch	H-123
FETCH	I-56
Fiche	I-83
FICHE	8-2-14, 8-5-3, 9-2-12, 9-3-5, H-97*, I-44*
Field length	I-79, I-97, I-103
File	9-2-6, 9-4-3, H-2, H-11, H-15, H-17, H-35, H-42, H-43, H-48, H-52, H-53, H-54, H-56, H-63, H-64, H-84, H-105, H-106, H-108, H-111, H-122, H-123, H-125, H-131, H-137, H-138, H-140, H-143, H-144, H-147, I-7, I-12, I-15, I-22, I-24, I-25, I-26, I-28, I-29, I-39, I-41, I-76, I-85, I-86, I-97, I-105, I-106, I-111, I-119, I-122, I-127
FILE	9-2-6, 9-5-3*, H-4*, H-44, I-45*
File attribute	H-122, I-45
File, command	I-137
File, direct	9-1-6*, I-8, I-34, I-57

File, indirect	9-1-6*, I-6, I-56, I-57, I-93, I-98, I-100
File, local	9-3-1, I-69, I-74, I-93, I-100, I-122
File management	8-2-7
File Manager	H-87
File mark	I-131
File migration environment	H-128
File name, local	I-2*
File, object	H-33, H-74
File, permanent	8-1-1, 8-1-4*, 8-2-8, 9-1-1, 9-1-6*, 9-2-8, H-12, H-144, I-3, I-8, I-16, I-18, I-34, I-56
File, procedure	H-73
File, random	I-24
File, terminal definition	H-51
File transfer	I-32
File, transfer	9-1-9
File transfer	H-116, H-173, H-175, I-65, I-133
FILE_CONNECTIONS	H-131, H-137
FILE_MANAGEMENT_UTILITY	8-2-7, H-98*
FILMU	H-98*
Flag	I-92, I-101
Flow control	8-2-1, 9-2-1
FLR	8-2-13, 9-2-11, 9-3-5, I-48*
FMU	H-98*
Fn (Proc)	9-3-2*
Fold	H-25
FOR	8-2-1, H-99*
FOREND	H-99
Form	H-117
FORM	I-48*
Format	H-23
Formatting, screen	I-87*, I-105
FORMAT_SCL_PROC	H-99*
FORMAT_SCL_PROCEDURE	8-2-13, 8-5-1, H-99*
Forms code	I-98
FORSCLP	H-99*
FORSP	H-99*
Fortran	9-6-8, I-52
FORTRAN	8-2-13, 9-2-4, H-100*, I-51*
Forward	H-153
Front-end	I-32
FSE	9-1-5, I-51*
FSE (editor)	9-2-10

FTN	8-2-13, H-100*
FTN5	9-2-10, I-52*
FTP	9-1-9, 9-1-10*
FTP prolog file	9-1-9*
FTPPRLG	9-1-9*
Full-screen	8-1-3, 9-1-3, 9-1-5, I-51
FUNCEND	8-5-1, H-103*
Function	8-5-1*, 8-8-1, H-1, H-66, H-92
FUNCTION	8-2-12, 8-5-1, H-103*, H-104*
Function attribute	8-5-1
Function keys	8-1-3, 9-1-3, 9-3-2
Function library	8-5-2, 8-5-3
Function module	8-5-2
Function name	8-5-1
Function parameter	8-5-1
Function table	H-105
GENCT	H-105*
GENERATE_COMMAND_TABLE	8-2-5, H-105*
GENERATE_LIBRARY	8-5-2, 8-7-10*
GENERATE_SCU_EDIT_COMMANDS	8-2-13, 8-6-3, H-105*
GENSEC	H-105*
GET	9-2-8, 9-7-1, I-8, I-56*
GETASC	9-2-8, 9-3-5, I-56*
GETATT	9-2-8, I-57*
GETF	H-106*
GETL	H-108*
GET_FILE	8-2-7, H-106*
GET_LINE	8-2-14, 8-5-1, 8-8-2, H-108*
Global	9-6-6, I-72
GLOBAL	9-6-4
GLOBAL (search)	8-8-2
GO	9-2-2, I-57*
GOODBYE	9-2-5, I-57*
GPSS	9-7-1
GRIPE	9-2-12, 9-3-5, I-58*
Group	H-13
Group identifier	9-5-1
GTR	9-2-10, I-59*
H	H-109*
Handler, condition	H-3
Header, procedure	9-3-3

HELLO	9-2-5, I-60*
Help	9-3-1, 9-3-2, H-60, H-94, H-95, I-42, I-61, I-62, I-135
HELP	8-2-5
%HELP	9-2-4
HELP	9-2-5, 9-3-5, H-46, H-109*, I-61*
Help, command	8-4-1
HELP (ICF)	I-135*
Help library	8-4-1*
HELP (Proc)	9-3-2*
HELPBE	9-2-5, I-62*
HELPME	9-2-5, I-62*
Hexadecimal	H-63
History	H-76
Host	I-74
HOTSPOT	9-7-1
IAF	9-1-1, 9-1-2
IC (Proc)	9-3-2*
ICF	I-134*
ICFSTATUS (ICF)	I-136*
IDENT	9-4-2*, 9-4-6
Identifier, group	9-5-1
Identifier, record	9-5-1
IF	8-2-1, 8-6-3, 9-2-1, H-110*, I-38, I-39, I-62*
IF (Proc)	9-3-1, 9-3-2*
IFEND	8-6-3, H-110*
IGNORE	9-5-3*
IMSLM	9-7-2
IMSLSS	9-7-2
INCC	H-110*
INCF	H-111*
INCL	H-111*
INCLUDE	9-6-4, I-123
INCLUDE_COMMAND	8-2-1, 8-5-1, 8-8-2, H-110*
INCLUDE_FILE	8-1-5, 8-2-1, 8-5-1, 8-8-2, H-111*
INCLUDE_LINE	8-2-1, 8-5-1, 8-8-2, H-111*
Indirect file	9-1-6*, I-6, I-56, I-57, I-93, I-98, I-100
INFORMATIONAL	H-4, H-7
Inhibit character	9-3-2
INIT	H-112*
INITIALIZE_TERMINAL	8-2-4, H-112*

Input	9-3-4, H-18, H-66, H-164, I-20, I-30
INPUT	H-5*
Input queue	I-111
Insert	H-110, H-111
INSERT	9-4-2*, 9-4-6, 9-5-3*
Instruction stack	8-1-1, 9-1-1
Interaction style	8-3-1
Interaction stype	H-19
INTERACTION_STYLE	H-131, H-137
Interactive	8-2-4, 9-2-4, H-3, H-117, H-143,
	H-161, I-101, I-138
Interactive Cray Facility	I-134*
Interactive Facility	8-1-1*, 9-1-1
Interactive procedure	I-10
Interactive status	8-2-5
Interpreter	H-24
Interrupt	H-163, I-4, I-134
INTERRUPT	H-3*
Interstate	H-88
Interstate connection	H-45, H-55
ITEMIZE	9-2-10, I-63*
Iterate	H-49
I/O	8-1-1
%J	8-2-5
JCL	I-1
job	9-2-2
Job	H-3, H-7, H-11, H-18, H-20, H-21,
	H-42, H-51, H-56, H-57, H-68, H-69,
	H-70, H-72, H-76, H-117, H-118,
	H-145, H-160, H-161, H-162, H-164,
	I-20, I-32, I-39, I-64*, I-65, I-91,
	I-103
JOB	8-1-5, 8-2-3, 8-8-2, H-112*
Job, batch	8-1-5*, 9-1-7*
Job classes, batch	8-1-6*, 9-1-8*
Job control	8-2-1, 9-2-2
Job control language	8-2-1, 9-2-1, I-1
Job control statement	8-5-1, 9-3-1
Job flow	8-8-2
Job library list	H-77
Job name	H-5, H-7, I-3
Job order number	8-1-2, 9-1-2, H-147, H-171
Job processing	8-2-5, 9-2-5

Job Sequence Name	I-2
Job sequence number	I-129
Job step	I-104
JOB COST	9-2-2, 9-3-5, I-65*
JO BEND	8-8-2
Jobs, killing batch	9-1-8*
JOB_NAME	H-5*, H-7*
JSN	I-2*
KFRMIT	8-2-5, H-116*, I-65*
Keypad	8-1-3, 9-1-5
Keys, function	8-1-3, 9-1-3, 9-3-2
Keys, terminal	8-1-3, 9-1-3
Keyword	H-2
Keyword parameter	I-1*
Killing batch jobs	8-1-6*, 9-1-8*
Label	9-3-1, H-15, H-25, H-57, H-78, I-13, I-38, I-39, I-74
LABEL	9-2-9, I-65*
Language	8-2-13, 9-2-10
LDSET	9-2-11, 9-6-3, 9-6-4, I-68*
LENGTH	9-2-2, I-69*
Length, line	H-27
Length, page	H-27
Letter	G1-1
LFN	I-2*
LGO	9-2-11, 9-6-3, I-69*
LIBEDIT	9-2-10, 9-5-1*, I-70*
LIBEDIT directive	9-5-1*
LIBGEN	9-2-10, 9-5-1, 9-5-3*, I-71*
LIBLOAD	9-2-11, 9-6-3, I-72*
Library	9-2-10, I-48, I-59, I-68, I-70, I-72, I-128
LIBRARY	9-2-10, 9-2-11, 9-6-3, 9-7-1, I-72*
Library, function	8-5-2, 8-5-3
Library, help	8-4-1*
Library list, debug	H-146
Library list, job	H-77
Library maintenance	8-2-12
Library, message	8-4-1*
Library, object	8-7-1*, 8-7-14*, 8-8-1, 9-5-1*, H-9, H-33, H-46, H-73, H-74

Library, procedure	8-5-2*, 8-5-3, 9-3-5
Library, program	8-7-14, 9-4-1, I-123
Library, source	8-6-1, H-34, H-96, H-159
Library, subprogram	8-7-14
Library, user	I-71, I-121
Limit	H-21, H-70
LIMIT	9-4-4*
Limit, output line	9-6-8, I-54
Limit, print	I-54
Limit, SRU	I-102, I-104
Limit, time	I-104
LIMITS	9-2-2, I-73*
LIMIT_FAULT	H-3*
Line	8-3-1, H-111
LINE	9-2-4, I-73*
Line length	H-27
Line limit, output	9-6-8, I-54
Line mode	H-19
Link	H-22, H-71
LINPACK	9-7-2
List	H-2, I-15, I-16
LIST	9-2-5, 9-4-3*, 9-5-3*, H-5*, I-74*
List, access control	H-57, H-59
List, command	H-41, H-53, H-61, H-62, H-138
List, debug library	H-146
List, job library	H-77
List of users	I-130
LISTLB	9-2-9, I-74*
LISTLID	9-2-2, I-74*
Load	9-5-1, I-11, I-68, I-69, I-72, I-77, I-79, I-81, I-107
LOAD	9-2-11, 9-6-1, 9-6-3, 9-6-4, I-75*
Load map	9-6-6, I-41, I-68, I-77, I-81
Load module	8-7-1, H-10
Loader	8-2-13, 9-2-11, 9-6-1*, H-9, I-92
Load/dump memory	8-2-10
Local file	9-3-1, I-69, I-74, I-93, I-100, I-122
Local file name	I-2*
LOCAL_FILE_NAME	H-5*
LOCK	9-2-6, I-75*, I-122
Log	H-72, H-73, H-76, I-33
Login	H-147, I-19, I-60
LOGIN	8-2-6, 8-8-2, 9-2-5, H-116*, I-75*
LOGINPR	I-126*

LOGOFF (ICF)	I-136*
LOGON (ICF)	I-136*
LOGOUT	8-2-6, 8-8-2, 9-2-5, H-117*, I-76*
Loop	I-39, I-129
LOOP	8-2-1, H-117*
LOOPEND	H-117*
Lower case	9-1-9
LO72	9-2-6, I-76*
Mail	H-10, H-27, H-29, H-33, H-85
MANAGE_FORM	8-2-12, H-117*
MANAGE_JOB	8-2-6, H-118*
MANAGE_OUTPUT	8-2-6, H-118*
MANF	H-117*
MANJ	H-118*
MANO	H-118*
Manual, on-line	8-9-1*, H-46, H-48, H-80, H-94, H-109, I-42
Manual, topics	H-48
MAP	9-2-11, 9-6-1, 9-6-3, I-77*
Map, load	9-6-6, I-41, I-68, I-77, I-81
Mark, substitution	H-7
Mass Storage System	8-1-1, 8-2-9, 9-1-1, H-120, H-122, H-123, H-124, H-125
Master	I-134, I-135
Master character	9-4-1
MEAPE	H-118*
Measure	H-10
Measurement, program	H-118
MEASURE_PROGRAM_EXECUTION	8-2-12, H-118*
Medium-speed	8-1-1, 9-1-1
Memory	8-1-1, 9-1-1, I-2, I-37
Memory image	9-6-1
Menu	9-3-2, H-145
MENU	H-145*
MERGE	8-2-7, H-119*, I-78*
Message	9-3-2, H-4, H-23, H-73, H-95, H-145, I-35, I-41
Message library	8-4-1*
Message module	8-4-1, 8-5-2
Message, status	8-4-1, 8-4-8
Message text	8-4-7*
MESSAGE_LEVEL	H-131, H-138
MFL	9-2-2, I-79*

Microcomputer	I-65
Microfiche	H-97, I-44*, I-83
Migration environment, file	H-128
Mode	I-7
MODE	I-79*
Mode, line	H-19
Mode, screen	H-19
Mode, search	8-8-2, H-16
Modify	9-4-2
MODIFY	I-79*
Module, function	8-5-2
Module, load	8-7-1, H-10
Module, message	8-4-1, 8-5-2
Module, object	9-5-1, 9-6-1, I-26, I-27, I-75
Module, procedure	8-5-2
Module, text	9-4-1
Mount	I-65
MOVE	9-4-2*
MSACCES	8-2-9, 8-5-3, H-120*
MSAUDIT	8-2-9, 8-5-3, 9-3-5, H-120*
MSCATLIST	8-2-9, 8-5-3, H-122*
MSCHANG	8-2-9, 8-5-3, H-122*
MSFETCH	8-2-9, 8-5-3, H-123*
MSFILES	8-2-9
MSGs	9-6-8
MSPASSW	8-2-9, 8-5-3, H-124*
MSPURGE	8-2-10, 8-5-3, H-125*
MSS	H-120, H-122, H-123, H-124, H-125
MSSTORE	8-2-10, 8-5-3, H-125*
Multi-file	I-44, I-67, I-74
NA	I-3*
NAM application	I-6
name	8-2-13, 9-2-1, 9-2-11, 9-6-3, I-79*
Name	9-3-3, H-2
Name, function	8-5-1
Name, job	H-5, H-7
Name, job sequence	I-2
Name, local file	I-2*
Name, procedure	8-5-1, H-135
NATURAL_LANGUAGE	H-131, H-138
Network	9-1-9*, I-74
NEW	9-5-3*
New routines	I-80

NEWCHRG	9-2-2, 9-3-5, I-80*
NEW	H-127*
NEWRTNS	9-2-12, 9-3-5, I-80*
News	H-127, H-128, H-139, I-80, I-84, I-90
NEWS	8-2-14, 8-5-3, 9-2-12, 9-3-5, H-128*, I-80*
NEW_ROUTINES	8-2-14, 8-5-3, H-127*
NOABBREV	9-4-3*
NOCLR (Proc)	9-3-2*
NOEXIT	9-2-1, I-81*, I-84
NOGO	9-2-11, 9-6-3, I-81*
NOINS	9-5-3*
NOLIST	9-4-4*
NOREP	9-5-3*
NORERUN	9-2-2, I-81*
NOREW	9-5-4*
NORMAL	9-2-4, I-82*
NOS	8-1-1, 9-1-1, H-14, H-22, H-36, H-45, H-71, H-88, H-106, H-120, H-122, H-123, H-124, H-125, H-140, H-143, I-1, G1-1 I-11, I-62 8-1-1*, G1-1 9-2-2, I-22, I-82*
NOS/BE	
NOS/VE	
NOTE	
NOTE (Proc)	9-3-2*
NSYS	8-10-1*, 9-5-5, 9-7-2
NULL	9-2-4
\$NULL	H-4, H-5, H-30, H-101, H-102, H-129, H-131, H-150, H-174, H-176
NULL	I-83*
Number, job order	H-147, H-171
Number, project	H-147
OASYS	9-1-9
Object	H-131, H-137
Object file	H-33, H-74
Object library	8-7-1*, 8-7-14*, 8-8-1, 9-5-1*, H-9, H-33, H-46, H-73, H-74
Object module	9-5-1, 9-6-1, I-26, I-27, I-75
Object program execution	9-6-8*
Object record	H-74
Octal	H-2, I-2, I-37, I-119
OFFSW	9-2-2, I-83*

Off-line request	I-83*
OFLREQ	9-2-12, 9-3-5, I-83*
OLD	9-5-4*
OLDN	H-128*
OLDNEWS	9-2-12, 9-3-5, I-84*
OLD_NEWS	8-2-14, 8-5-3, H-128*
Omit	9-3-1
ONEXIT	9-2-1, I-84*
ONSW	9-2-2, I-84*
On-line	I-61, I-62
On-line manual	8-9-1*, H-46, H-48, H-80, H-94, H-109, I-42
OP	9-6-8
OPEFMA	H-128*
OPEN_FILE_MIGRATION_AID	8-2-7, H-128*
Operating system	8-1-1, 9-1-1
Operator	H-145
Option	H-24, H-78
OUT	9-2-6, I-85*
Output	H-5*, H-23, H-75, H-76, H-77, H-118, H-145, H-165, I-20, I-135
OUTPUT	H-5*
Output line limit	9-6-8, I-54
Overlay	9-6-2
Overwrite	I-85
OVWRITE	9-2-6, I-85*
PACK	9-2-6, I-86*
Page length	H-27
PAGE (Proc)	9-3-2*
Page width	H-27
Panel	I-105
Panel definition	I-87*
Parameter	9-3-3, 9-3-4, H-4, H-9, H-60, I-1*, I-2, I-10
Parameter, function	8-5-1
Parameter, keyword	I-1*
Parameter, positional	I-1*
Parameter, procedure	8-5-1, H-135
Parameter Substitution	8-4-7
Parity	H-27
PASCAL	8-2-13, H-129*
PASSWOR	9-2-3, I-86*
Password	8-1-2, 9-1-2, H-5*, H-22, H-124, I-3,

Password	I-17, I-86, I-127
PASSWORD	H-5*
pathname	8-2-13
Pattern	H-25
Pause	H-163, I-57, I-86
PAUSE	9-2-3, H-3*, I-86*
PC	I-65
PCA	H-118
PDU	I-87*
PERIOD (ICF)	I-137*
Peripheral processor	8-1-1, 9-1-1
Permanent file	8-1-1, 8-1-4*, 8-2-8, 9-1-1, 9-1-6*, 9-2-8, H-12, H-144, I-3, I-8, I-16, I-18, I-34, I-56
Permission	8-2-8, 8-2-9, 9-1-9, 9-1-10, 9-2-8, H-39, H-52, H-54, I-2*, I-88
PERMIT	9-2-8, I-88*
PERT78	9-7-1
PFN	I-3*
PL	9-4-1*, 9-4-5
PL	9-6-8
PL (program library)	I-123
PLAY	I-135
PLAY (ICF)	I-137*
Plural	H-1
PMD	9-6-8
POP	8-2-14, H-131*
Pop up	H-152, H-153
Position	H-144, H-153, I-97, I-105, I-106
Positional parameter	I-1*
Post Mortem Dump (PMD)	9-6-8
PP	8-1-1, 9-1-1
Prefix character	I-137
PREFIX (ICF)	I-137*
Preset	I-68, I-103
PRIF	H-131*
Print	H-12, H-131, H-165, H-171, I-28, I-29, I-122
Print limit	I-54
Printer	9-1-1
Printer, auxiliary	H-12, I-8, I-9
PRINT FILE	8-2-7, H-131*
Priority (batch)	I-64
Priority (batch job)	8-1-6, 9-1-8
Private	I-88

PROC (Proc)	9-3-3*
Procedure	8-2-11, 8-5-1*, 8-8-1, 9-2-10, 9-3-1*, 9-3-1, 9-3-5, H-99, H-135, I-10, I-41, I-73, I-96
PROCEDURE	8-2-11, 8-5-1, H-135*
Procedure attribute	8-5-1, H-135
Procedure file	H-73
Procedure header	9-3-3
Procedure library	8-5-2*, 8-5-3, 9-3-5
Procedure module	8-5-2
Procedure name	8-5-1, H-135
Procedure parameter	8-5-1, H-135
PROCEND	8-5-1, H-135*
PROCFIL	9-3-5*, 9-7-1, I-10
PROCLIB	8-5-3
PROCOMM	I-65
Profile, program	H-47
Program	8-8-1, H-77, H-89, H-149, H-163, I-72
Program library	8-7-14, 9-4-1, I-123
Program measurement	H-118
Program profile	H-47
Programming environment	H-87
Programming language	8-2-13
PROGRAM_ATTRIBUTES	H-131, H-138
PROGRAM_FAULT	H-3*
Project number	8-1-2, H-147
Prolog	9-1-9, H-5*, I-126
PROLOG	H-5*
Prompt	9-3-1, 9-3-2, 9-3-3, 9-3-4, H-24
PROMPT (Proc)	9-3-4*
PRU	8-1-4*, 9-1-6*
PURDECK	9-4-3*
PURGALL	9-2-8, I-88*
Purge	H-125, I-88, I-89
PURGE	8-2-9, 8-5-3, 9-2-8, 9-4-3*, H-137*, I-89*
PUSH	8-2-14, H-137*
PUSH_COMMANDS	8-2-6, H-138*
PUTASC	9-2-8, 9-3-5, I-89*
PUTL	H-138*
PUT_LINE	8-2-7, 8-5-1, 8-8-2, H-138*
PW	I-3*

QGET	9-2-3, I-90*
Queue	H-77, H-164, I-31, I-37, I-85, I-90
Queue, input	I-111
QUIT	8-4-6*, 8-5-2, 8-6-2, 8-7-11*, H-55*
QUIT (ICF)	I-138*
Random file	I-24
RATES	8-1-6, 8-2-14, 8-5-3, 9-2-12, 9-3-5, H-139*, I-90*
RBF	9-1-1, I-17
Read	H-108
READ	9-4-3*
Real-time	I-99
Recall	I-91
RECLAIM	9-2-8, I-91*
Record	I-24, I-25, I-86, I-106, I-131
Record identifier	9-5-1
Record information	I-63
Record manager	H-9
Record Manager	9-6-4, 9-6-6, I-48
Record, object	H-74
RECOVER	9-2-5, I-91*
REDO	9-2-5, I-91*
REDUCE	9-2-11, 9-6-3, I-92*
Reformat	H-99, I-76
Reformatting directive	9-1-7*
Register	8-1-1, 9-1-1, I-101
Release	I-41, I-96, I-122
RELEASE_RESOURCE	8-2-10, H-139*
Relocatable	9-5-1, 9-6-6
Remote Batch Facility	9-1-1
Remote terminal	8-1-1, 9-1-1
Remove	I-86
RENAME	9-2-6, 9-5-4*, I-93*
REORDER_MODULE	8-5-2, 8-7-11*
Repeat	H-49, H-99, H-117, H-140, H-169
REPEAT	8-2-1, H-140*
Repetitive execution	H-99, H-117, H-140, H-169
REPF	H-140*
Replace	9-5-3, I-26, I-27
REPLACE	9-2-8, 9-5-4*, I-93*
REPLACE_FILE	8-2-8, H-140*
REPLACE_LIBRARY	8-6-2
REPLACE_MODULE	8-5-2, 8-7-12*

REQMT	H-141*
REQT	H-143*
REQUEST	9-2-7, 9-2-9, I-93*, I-94*
Request, off-line	I-83*
REQUEST_MAGNETIC_TAPE	8-2-10, H-141*
REQUEST_TERMINAL	8-2-4, H-143*
Rerun	I-81, I-94
RERUN	9-2-3, I-94*
RESC	H-144*
RESERVE_RESOURCE	8-2-10, H-143*
Reset	I-82
RESFF	H-143*
RESOURC	9-2-3, I-95*
Resource, job	I-64
RESPF	H-144*
RESR	H-143*
Restart	9-2-9
RESTART	9-2-9, I-95*
RESTORE	9-4-3*
RESTORE_FOREIGN_FILES	8-2-10, H-143*
RESTORE_LOG	8-2-12
RESTORE_PERMANENT_FILES	8-2-9, H-144*
RESTRICTED (search)	8-8-2
Restructure	H-10
Resubmit	H-18
Resume	I-41
RESUME (ICF)	I-138*
RESUME_COMMAND	8-2-3, H-144*
RETRY	H-3*
Return	H-139, I-20, I-96, I-122
RETURN	9-2-7, I-96*
REVERT	9-2-1, 9-2-10, I-96*
REWF	H-144*
REWIND	9-2-7, 9-4-3*, 9-5-4*, I-97*
REWIND_FILE	8-2-8, H-144*
Re-execute	I-91
RFL	9-2-3, 9-2-11, 9-6-3, 9-6-4, I-97*
Ring	H-147
Root segment	9-6-4, 9-6-6
ROUJ	H-145*
Route	I-111
ROUTE	9-1-7, 9-2-7, I-98*
ROUTE_JOB	8-2-6, H-145*
Routines, new	I-80
RTIME	9-2-3, I-99*

Run
RUNTIME_CHECKS

H-88
H-5*

%S
SATISFY
SATISFY_EXTERNAL_REFERENCE
SAVE
SCL

8-2-5, 9-2-4, I-5*
9-2-11, 9-6-3, I-99*
8-7-12*
9-2-8, I-93, I-100*
8-2-1, 8-5-2, H-1*, H-24, H-78,

SCOPE
SCOPY
Screen

H-103, H-163
H-6*
9-2-7, I-100*
8-1-3, 8-3-1, 9-1-3, 9-3-2, H-56,
I-51

SCREEN
Screen formatting
Screen mode
Scroll
SCU

9-1-3, 9-1-5, 9-2-4, I-101*
I-87*, I-105
H-19, I-101
I-73
8-6-1*, 8-6-3, H-25, H-34, H-84,

Search mode
Search order
SEGLOAD
Segment

H-93, H-96, H-159*
8-8-2, H-16
8-8-1
9-6-4
9-6-6

Segmentation
SELDUMP
SELECT_USER_MENU
SELUM
Semi-private

9-6-2, 9-6-4*, 9-6-6
I-11
8-2-14, H-145*
H-145*
I-88

Send
SEND_OPERATOR_MESSAGE
SENOM
Sense switch
SEQUENCE

I-35
8-2-14, H-145*
H-145*
H-151, I-83, I-84, I-112
9-4-3*

SEQUENCE_DECK
SEQUENCE_MODIFICATION
SET
SET (Proc)
SETASL

8-6-2
8-6-2
9-2-1, I-101*
9-3-4*
9-2-3, I-102*

SETCORE
SETDLP
SETDR
SETFA
SETFL

9-2-3, I-103*
H-147*
H-147*
H-147*
H-146*

SETFS

9-2-7, I-103*

SETJOB	9-2-3, I-103*
SETJSL	9-2-3, I-104*
SETML	H-23*
SETPA	H-149*
SETPR	9-2-3, I-104*
SETPW	H-22*
SETSS	H-151*
SETTA	H-26*
SETTL	9-2-3, I-104*
SETWC	H-28*
SET_COMMAND_LIST	8-2-6
SET_DEBUG_LIST	8-2-6, H-146*
SET_DEBUG_RING	8-2-6, H-147*
SET_DEFAULT_LOGIN_PROJECT	8-2-6, 8-5-3, H-147*
SET_DISPLAY_OPTION	8-5-2, 8-7-13*
SET_FILE_ATTRIBUTES	8-2-9, H-147*
SET_LINK_ATTRIBUTES	8-2-4
SET_LIST_OPTIONS	8-6-2
SET_MESSAGE_LEVEL	H-23*
SET_PASSWORD	H-22*
SET_PROGRAM_ATTRIBUTES	H-149*
SET_SENSE_SWITCH	8-2-3, H-151*
SET_TERMINAL_ATTRIBUTES	H-26*
SET_WORKING_CATALOG	H-28*
Severity	H-3, H-4
Shift	I-28, I-29
SHOF	H-152*
SHOV	H-153*
SHOW	9-2-5, I-105*
SHOW_FILE	8-2-8, H-152*
SHOW_VALUE	8-2-14, H-153*
SI (tape)	I-117
SIMII5	9-7-1
Single space	I-28, I-29
SJN	H-7*
Skip	9-3-1, I-38, I-39, I-62
SKIP	9-2-1, I-39, I-105*
SKIPEI	9-2-7, I-105*
SKIPF	9-2-7, 9-4-3*, I-105*
SKIPFB	9-2-7, I-106*
SKIPR	9-2-7, I-106*
SKIP_TAPE_MARK	8-2-11, H-153*
SKITM	H-153*
Slave	I-134, I-135
SLOAD	9-2-11, 9-6-3, I-107*

Software	8-10-1*, 9-7-1
Sort	H-119, I-107
SORT	8-2-8, H-154*, I-107*
SORT5	I-107*
SOUCU	H-159*
Source	9-4-1, 9-4-6
Source library	8-6-1, H-34, H-96, H-159
SOURCE_CODE_UTILITY	8-2-13, 8-6-1*, H-159*
SOURCE_CODE_UTILITIY	H-25
SPY	H-118
SRU	I-39, I-65, I-102, I-104, I-111
STAC	H-160*
Start	I-75, I-136
Statement	H-13
Statement, structured	H-92
Status	9-2-4, H-6*, H-7, H-70, H-77, H-79, H-160, I-5, I-20, I-31, I-39, I-69, I-81, I-103, I-136, I-138
STATUS	H-6*
STATUS (ICF)	I-138*
Status message	8-4-1, 8-4-8
STATUS_CRAY	8-2-3, 8-2-10, 8-5-3, H-160*
STIME	9-2-3, I-111*
Store	H-125
Stream, command	H-110, H-111
String	H-2, H-110, I-1
Structure	8-8-2
Structured statement	H-92
Stype, interaction	H-19
SUBCJ	H-161*
Subcommand	8-8-1, H-9
SUBDJ	H-161*
SUBJ	H-162*
Submit	H-18, H-161, H-162, I-30, I-32
SUBMIT	9-1-7, 9-2-3, I-111*
SUBMIT_CRAY_JOB	8-2-3, 8-2-10, 8-5-3, H-161*
SUBMIT_DETACHED_JOB	8-2-3, H-161*
SUBMIT_JOB	8-1-5, 8-2-3, H-162*
Subprogram library	8-7-14
Substitution mark	H-7
substitution, parameter	8-4-7
SUBSTITUTION_MARK	H-7*
Subsystem	9-2-4, I-5, I-10, I-35, I-51, I-83
Suggestions	I-58
Suspend	H-168

SUSPEND (ICF)	I-138*
SWITCH	9-2-3, I-112*
Switch, sense	H-151
Synchronous	H-163
Syntax	H-1
System Control Language	8-2-1
System utility	8-2-11
SYSTEM_JOB_NAME	H-7*
Table, command	H-105, H-163
Table, function	H-105
Table, Utility command	H-32
TABLEND	8-2-12, H-163*
TAPDMP	9-2-9, 9-3-5, I-112*
Tape	9-2-9, H-14, H-15, H-25, H-36, H-48, H-57, H-78, H-139, H-141, H-143, H-153, I-11, I-13, I-65, I-74, I-83, I-93, I-95, I-117, I-119, I-120, I-128
Tape dump	I-112, I-114
Tape management	8-2-10
TAPEDMF	9-2-9, I-114*
Task	H-51, H-57, H-79, H-88, H-89, H-163, H-165
TASK	8-2-3, 8-8-2, H-163*
TASKEND	8-8-2
TCOPY	9-2-7, I-117*
TCP/IP	9-1-9
TDU	9-2-4, I-118*
TDUMP	9-2-7, I-119*
Tektronix	9-1-3
*TEL	H-7
Telephone	8-1-2, 9-1-2
Telnet	9-1-9
Template	8-4-7
TERC	H-163*
TERI	H-164*
TERJ	H-164*
Terminal	H-17, H-26, H-57, H-63, H-79, H-80, H-112, H-143, I-5, I-31, I-73, I-91, I-101, I-120, I-134, I-135
Terminal control	8-2-4, 9-2-4
Terminal definition	I-118, I-120
Terminal definition file	H-51

Terminal keys	8-1-3, 9-1-3
Terminate	H-55, H-117, H-163, H-164, H-165, I-4, I-15, I-20, I-38, I-39, I-57, I-75, I-76, I-134, I-135, I-136, I-138
TERMINATE_COMMAND	8-2-6, H-163*
TERMINATE_INPUT	H-164*
TERMINATE_JOB	8-1-6, 8-2-6, H-164*
TERMINATE_OUTPUT	8-2-8, H-165*
TERMINATE_PRINT	H-165*
TERMINATE_TASK	8-2-6, H-165*
TERMINATION_ERROR_LEVEL	H-7*
Terminator	I-137
TERO	H-165*
TERP	H-165*
TERT	H-165*
Text	H-7, H-74, H-93
TEXT	8-6-3, 9-4-4*
Text, message	8-4-7*
Text module	9-4-1
TEXTEND	8-6-3
Time	H-2, H-168, I-33, I-39, I-65, I-99
TIME	9-6-8
Time limit	I-104
TITLE	H-7*
Topics	H-80
Topics manual	H-48
TPAUDIT	9-2-9, 9-3-6, I-119*
TPGET	9-2-9, 9-3-6, I-119*
TPRLS	9-2-9, 9-3-6, I-120*
Transfer	I-10
Transfer control	H-92
Transfer, file	H-116, H-173, H-175, I-65, I-133
Transfer file	9-1-9
Tree	9-6-4
TREE	9-6-4
Tree diagram	9-6-5*
TRIVIAL	H-4
TRMDEF	9-2-4, I-120*
Truncate	I-76
Type	9-5-1, H-166
TYPE	8-2-14, 8-5-1, 9-5-4*, H-166*
TYPEND	8-5-1, H-166*

UJN	H-7*, I-3*
ULIB	9-2-10, I-121*
UN	I-3*
UNICOS	I-122
UNIROUT	9-2-7, I-122*
UN=LIBRARY	9-7-1
UNLOAD	9-2-7, I-122*
UNLOCK	9-2-7, I-122*
UN=NSYS	9-7-2
Unsatisfied externals	I-99
UPDATE	8-6-3, 9-2-10, 9-4-1*, 9-4-5, 9-4-6, H-34, I-123*
UPDATE directive	9-4-1*
UPROC	9-2-3, I-126*
USER	9-2-3, I-127*
User ID	I-130
User initials	8-1-2, 9-1-2, G1-1
User library	I-71, I-121
User name	I-3, I-17, I-129
Username	H-171, G1-1
Users	H-169
Users, list of	I-130
User-break-1	I-4
USER_JOB_NAME	H-7*
USE_LIBRARY	8-6-3
Utility	8-8-1, 9-2-10, H-28, H-92, H-104, H-166
UTILITY	8-2-12, 8-7-14*, 8-8-2, 9-5-5, 9-7-2, H-9, H-166*
Utility, command	H-166
Utility command table	H-32
Utility, system	8-2-11
UTILITYEND	8-8-2
Validate	I-19, I-73, I-127
Value	H-1, H-80
Value descriptor	H-2
VAR	8-2-14, 8-5-1, H-167*
VAREND	8-5-1, H-167*
Variable	8-2-13, H-42, H-44, H-55, H-81, H-108, H-131, H-137
VAX	H-48
VEIAF	8-1-1, 8-1-2, H-9
Vendor	9-7-1

Verify	9-5-4
VERIFY	9-2-7, I-127*
VFYLIB	9-2-10, 9-5-4*, I-128*
Viking 721	9-1-3
Volume serial number	I-67, I-128
VSN	9-2-9, I-67, I-128*
VT-100, DEC	8-1-3, 9-1-3
WAIT	8-2-3, 8-5-1, H-168*
WARNING	H-4, H-7
WEOF	9-4-2*
WEOP	8-6-3
WEOPC	8-6-3
WHATJSN	9-2-5, I-129*
WHEN	8-2-3, 8-5-1, H-3, H-34, H-49*, H-168*
WHENEND	8-5-1, H-3, H-168*
WHILE	8-2-3, 9-2-1, H-169*, I-39, I-129*
WHILEND	H-169*
WHO	8-2-15, 8-5-4, H-169*, I-130*
WHOAMI	8-2-15, 8-5-4, 9-2-12, 9-3-6, H-171*, I-130*
WIDTH	9-4-2*, H-7*
Width, page	H-27
Wildcard	H-25
Window	H-7, H-8, H-152, H-153
Working catalog	H-28, H-82
WORKING_CATALOG	H-131, H-138
Write	H-138, I-75
WRITEF	9-2-7, I-131*
WRITER	9-2-7, I-131*
WRITE_LIBRARY	8-6-3
X	9-2-5, I-131*
X (windowing)	H-8*
X,BASIC	I-10, I-131*
Xerox	I-83
XEROX	8-2-15, 8-5-4, 9-2-12, 9-3-6, H-171*, I-132*
Xerox 8700	H-171
XMODEM	9-2-5, I-133*
XMODEM_RECEIVE	8-2-6, H-173*
XMODEM_SEND	8-2-6, H-175*

XMOR
XMOS

H-173*
H-175*

Y (windowing)
YANK
YANKDEC

H-8*
9-4-3*
9-4-3*

%1
180, CDC CYBER

9-1-9, 9-2-4, H-163, I-4*
8-1-1*, 9-1-1

%2

9-1-9, 9-2-4, I-4*

\$LOCAL
\$LOCAL.LGO
\$NULL
\$\$SYSTEM

8-8-1
8-2-13
H-4, H-5, H-30, H-101, H-102, H-129,
H-131, H-150, H-174, H-176
8-8-1, 8-8-2

%
%A
%D
%E
%HELP

I-4*
8-2-5
8-2-5, 9-2-4, I-5*
9-2-4, I-5*
9-2-4

%J
%S
%1
%2

8-2-5
8-2-5, 9-2-4, I-5*
9-1-9, 9-2-4, H-163, I-4*
9-1-9, 9-2-4

*CORE
*DA
*MSG
*OP
*PL

9-6-8*
9-6-8*
9-6-8*
9-6-8*
9-6-8*

*TIME

9-6-8*

Initial Distribution

Copies:

2 Director
Defense Technical Information Center (DTIC)
Cameron Station
Alexandria, Virginia 23314

Center Distribution

Copies:

1	35/3509	Gray, G. R.
1	3501	Minor, L. R.
1	3502	
1	351	Willner, S. E.
150	3511	Willner, S. E.
20	3511(A)	Sommer, D. V.
1	3512	Brady, M.
1	3513	Wessel, J.
1	353	Yearick, R.
1	3533	Hayden, H. F.
1	3533	Annapolis Computer Center
1	355	Kearney, E.
1	3551	
1	3552	Singla, D.
1	357	Weachter, R.
1	3571	Knowles, H.
1	3572	Smith, T.
1	522	TIC (C)
1	522.2	TIC (A)

**END
FILMED**

DATE:

4-91

DTIC