VAX–11 RSX Compatibility Mode Reference Manual

Order Number: AA-CJ40A-TE

digital software

VAX–11 RSX Compatibility Mode Reference Manual

Order Number: AA-CJ40A-TE

September 1984

Revision/Update Information:	This is a new manual.
Operating System and Version:	VAX/VMS Version 4.0
Software Version:	VAX–11 RSX Version 1.0

digital equipment corporation
maynard, massachusetts

September 1984

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.

The postpaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC
DEC/CMS
DEC/MMS
DECnet
DECsystem-10
DECSYSTEM-20
DECUS
DECwriter
DIBOL
EduSystem
IAS
MASSBUS
PDP
PDT
RSTS
RSX
UNIBUS
VAX
VAXcluster
VMS
VT
digital

ZK-2314

This document was prepared using an in-house documentation production system. All page composition and make-up was performed by TEX, the typesetting system developed by Donald E. Knuth at Stanford University. TEX is a registered trademark of the American Mathematical Society.

Contents

FIGURES

TABLES

Preface

Manual Objectives

The *VAX-11 RSX Compatibility Mode Reference Manual* is the main source of technical information needed for using VAX-11 RSX on a VAX/VMS system. It also describes some important differences between the RSX-11 and VAX/VMS operating systems.

Intended Audience

This manual is for VAX-11 RSX users who have some experience with computers and a general understanding of the RSX-11 and the VAX/VMS operating system. It is not a tutorial manual; inexperienced VAX-11 RSX users should refer first to the appropriate introductory manuals in the RSX-11 and VAX/VMS documentation sets. (See the Associated Documents section of this Preface.)

Structure of This Document

This manual is organized into four parts, each part consisting of one or more sections.

Part I—Introduction to the VAX-11 RSX Environment

- Section 1, Introduction to the Software and Hardware, defines some differences between VAX/VMS and RSX-11 and provides general information on the software and hardware associated with VAX-11 RSX.
- Section 2, The VAX-11 RSX User Environment, provides information and examples for getting started on the system.
- Section 3, The VAX-11 RSX Programming Environment, provides information on programming and task execution and describes various RSX-11 and VAX/VMS functions used by the programmer.

Part II—VAX-11 RSX Command Dictionary

- Section 4, MCR Commands, describes all VAX-11 RSX MCR commands and provides examples of each.
- Section 5, Indirect Command File Processing, describes all Indirect Command Processor (Indirect) directives and symbols supported on VAX-11 RSX and also includes examples and a discussion of error message processing.
- Section 6, DCL Commands, describes all VAX-11 RSX Back Translator (BACKTRANS) DCL commands and provides examples of each.

Part III—Directives and I/O

- Section 7, System Directives, describes VAX-11 RSX system directives and, where applicable, provides examples and notes on the differences between VAX-11 RSX and RSX-11 support of directives.

- Section 8, I/O Functions, describes I/O functions, I/O drivers, status returns, and function codes.

Part IV—VAX-11 RSX Messages

- Section 9, System Messages, describes compatibility mode error messages for the Monitor Console Routine (MCR), the Applications Migration Executive (AME), the Back Translator (BACKTRANS), and Indirect.
- Section 10, Error Codes, describes RSX-11 File Control Services error codes and directive error codes.

Associated Documents

The *VAX-11 RSX Compatibility Mode Reference Manual* is primarily self-contained. However, you may also need to be familiar with certain system features and release information that are described in the *VAX-11 RSX Installation Guide and Release Notes*. In addition, several documents in the VAX/VMS and the RSX-11M/M-PLUS documentation sets provide valuable information to the VAX-11 RSX user.

The following VAX/VMS documents may be useful:

- *Introduction to the VAX/VMS Document Set*
- *VAX/VMS Glossary*
- *VAX/VMS Master Index*
- *VAX/VMS Release Notes*
- *Introduction to VAX/VMS*
- *VAX/VMS DCL Dictionary*
- *Guide to Using DCL and Command Procedures on VAX/VMS*
- *VAX/VMS Utilities Reference Volume*

The *General Information Volume* provides a complete list of all VAX/VMS documents and briefly describes each.

The following RSX-11M/M-PLUS manuals may be useful:

- *RSX-11M/RSX-11S* or *RSX-11M-PLUS Information Directory and Master Index*
- *RSX-11M/M-PLUS and Micro/RSX Executive Reference Manual*
- *RSX-11M/M-PLUS Guide to Program Development*
- *RSX-11M/M-PLUS and Micro/RSX Task Builder Manual*
- *RSX-11M/M-PLUS Utilities Manual*
- *IAS/RSX-11 ODT Reference Manual*
- *PDP-11 MACRO-11 Language Reference Manual*
- *RSX-11M/M-PLUS RMS-11 User's Guide*
- *RSX-11M/M-PLUS RMS-11 Utilities*

The *RSX–11M/RSX–11S Information Directory and Master Index* or the *RSX–11M–PLUS Information Directory and Master Index* lists all RSX–11M, RSX–11M–PLUS, RSX–11S, and RMS–11 documents, including the order numbers and brief descriptions of each document. Each RSX–11M, RSX–11M–PLUS, RSX–11S, and RMS–11 manual must be ordered separately.

Conventions Used in This Document

The following conventions are observed in this manual.

Convention	Meaning
CTRL/x	A symbol that indicates the CTRL key; it must be held down while another key is pressed. For example, CTRL/C means hold down CTRL while pressing C.
xxx	A one- to three-character key symbol. For example, ESC indicates the ESCAPE key, LF indicates the LINE FEED key, and RET indicates the RETURN key.
n	A lowercase n; indicates a variable for a number.
,	Comma; separates parameters in commands.
.	A period; separates the file name from the file type in a file specification.
;	A semicolon; separates the file type from the file version number in a file specification.
. . .	Horizontal ellipsis; indicates that you can enter additional parameters, values, or other information.
. . .	Vertical ellipsis; indicates that not all the statements in an example or figure are shown.
MCR>	The explicit prompt of the Monitor Console Routine (MCR), the command interpreter provided by VAX–11 RSX.
>	The MCR prompting character that appears whenever control is returned to the compatibility mode terminal and you can type input.
-	A hyphen; when typed as the last element on a command line, means that the input continues. (After each continued command line, MCR prompts you for additional input with a right-angle bracket followed by an underscore.)
>_	The MCR prompt for additional input.
$	The VAX/VMS DIGITAL Command Language (DCL) prompting character. (Whenever control is returned to the native mode terminal and you can type input, this prompt appears.)
[ ]	Square brackets; usually indicate optional syntax. Brackets that are part of directory names, however, do not indicate optional syntax. In addition, certain MCR directives use brackets as part of their required syntax.
gray shading	Shading used to indicate differences in VAX–11 RSX support of directives.
red ink	Color of ink used to show all user-entered commands in examples.

Convention	Meaning
black ink	Base color of ink used throughout manual. In examples, however, black ink has a special meaning: All output lines and prompting characters that the system prints or displays are shown in black ink.

PART I Introduction to the VAX–11 RSX Environment

1 Introduction to the Software and Hardware

VAX-11 RSX is the name of the optional software product that allows RSX-11 task images to execute in compatibility mode and makes it possible for task images from an RSX-11 operating system to migrate to a VAX/VMS system and to execute there. Compatibility mode allows the creation, assembly or compilation, linking, execution, and debugging of PDP-11 programs that were migrated from RSX-11 systems or that are developed for target RSX-11 systems.

For VAX-11 RSX compatibility mode execution to occur, the program must be satisfied on two levels:

- The hardware instruction set level
- The level of program interface to the operating system

Some VAX-11 processors may not provide the hardware support required for VAX-11 RSX. Consult your Software Product Description (SPD) to determine whether your VAX-11 processor supports VAX-11 RSX.

Since there are the two instruction sets—the VAX-11 set and the PDP-11 compatibility mode set—the VAX-11 processor has two basic modes of operation: native and compatibility. The processor operates in native mode to execute native mode instructions. It operates in compatibility mode to execute PDP-11 instructions. VAX-11 RSX selects the appropriate processor mode. Thus, when an RSX-11 program has been prepared for execution, VAX-11 RSX places the processor in compatibility mode just before passing control to the program. This is accomplished in a manner that is transparent to the user.

At times, while executing an RSX-11 program in compatibility mode, the VAX-11 processor must switch back to native mode because the RSX-11 program requires a service from VAX-11 RSX. The task's attempt to interface with the operating system results in a hardware-generated trap. Hardware-generated traps are the mechanism the processor has for notifying VAX-11 RSX that emulation of the RSX-11 operating system's environment is temporarily required for some specific service the task needs from the operating system. The occurrence of such a compatibility mode trap automatically places the processor in native mode. Then VAX-11 RSX, executing in native mode, effectively duplicates the RSX-11 task/system interface. Finally, VAX-11 RSX returns to the task in compatibility mode to allow the task to continue to execute.

Some aspects of RSX-11 have no counterpart in VAX/VMS. Because no parallel function exists in VAX/VMS, VAX-11 RSX cannot translate those aspects of RSX-11 to VAX/VMS functions. For example, partitions, significant events, and priorities that range from 1 through 250 have no counterpart in VAX/VMS. Although VAX-11 RSX does not duplicate these RSX-11 features, it does accept task image requests related to them and takes appropriate action to allow a task image to continue to execute.

Other aspects of RSX–11 are similar to VAX/VMS functions. For example, both systems employ user identification codes (UICs). In RSX–11M Version 4.1 and RSX–11M–PLUS Version 2.1, UICs are used as account (login) identifiers that provide a default User File Directory (UFD) and are employed for file protection. In VAX/VMS, UICs are primarily used for file protection and are associated with a user account instead of being employed as account identifiers. UICs have no relationship to the default file directory. Section 2 provides more information about UICs under VAX–11 RSX.

Version 1.0 of VAX–11 RSX supports execution of RSX–11M Version 4.1, RSX–11M–PLUS Version 2.1, and Micro/RSX Version 1.1 nonprivileged task images in compatibility mode. In RSX–11M and RSX–11M–PLUS most nonprivileged user mode task images run under VAX–11 RSX without program modification or rebuilding. Others may require modification to execute, and still others may not execute at all.

For an RSX–11 task image to execute successfully under VAX–11 RSX, it must adhere to the requirements for compatibility mode operation. These requirements are detailed in 1.3.

1.1 Distinction Between a Task and a Process

In VAX–11 RSX, the concept of an RSX–11 task is separated into two basic components: (1) the program image that executes and (2) the control information and virtual address space required for an image to execute. These two components correspond to the VAX/VMS concepts of process and image.

A **process** is the basic schedulable unit executed by the VAX–11 processor. It consists of a virtual address space and control information (such as saved register contents and status information) that both the hardware and the software require. This control information is called the **process context**.

An **image** is the result of one or more object modules being linked together. It can be linked by the VAX/VMS Linker to execute in native mode or by the Task Builder to run in compatibility mode. An image executes in the virtual address space provided by a process and under control of the process.

A process's virtual address space is divided into two areas: the program region and the control region, as illustrated in Figure 1–1. Essentially, the program region provides virtual memory for an image. The control region contains information required by the system to control the process.

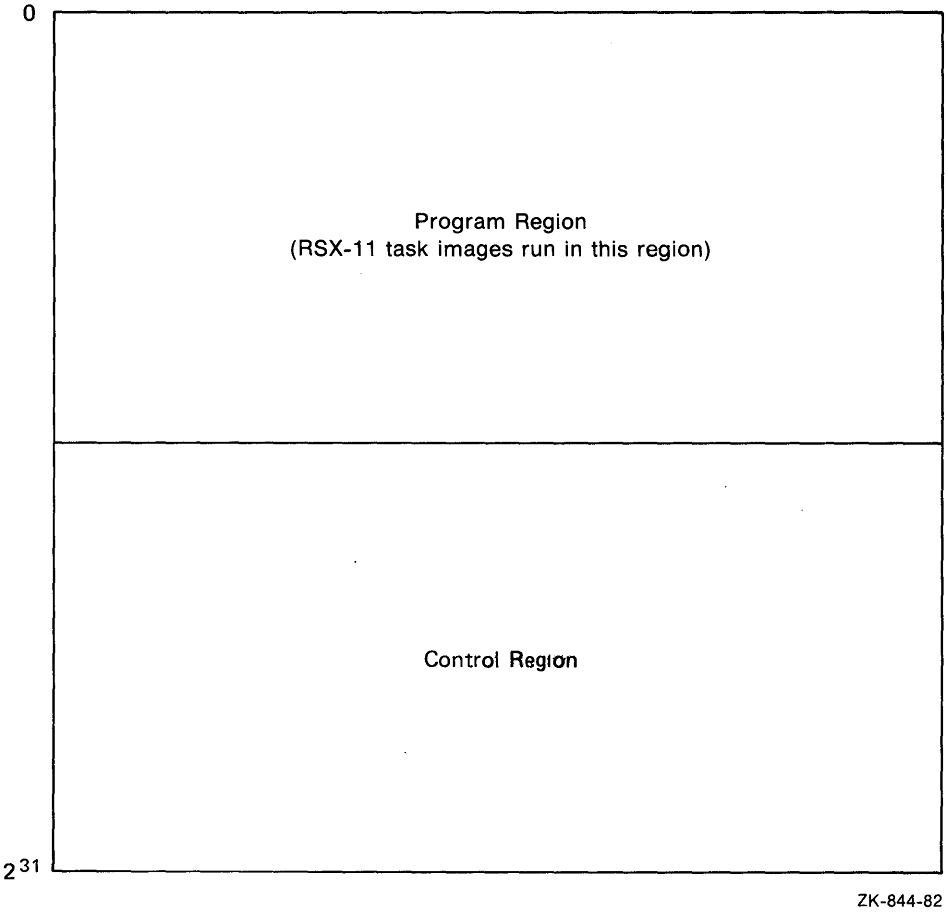
The combination of process and image is similar to the RSX–11 task. The main difference is that a **task** is a form of process that executes a specific image, whereas a process can execute any image. Furthermore, a process can execute subsequent images after the current image exits.

In RSX–11, a reference to a specific task also implies a specific reference to an image. Since this is not the case with VAX/VMS, it is useful to state explicitly whether an operation affects a process or its image.

1.2 VAX–11 RSX Compatibility With PDP–11 User Mode Operations

VAX–11 RSX compatibility mode supports PDP–11 user mode operations. In other words, any PDP–11 program that operates only in user mode (not in PDP–11 supervisor or kernel mode), and that assumes instruction space and

Figure 1–1 Process Virtual Address Space



data space (I-space and D-space) are overmapped, can run in VAX–11 RSX compatibility mode.

Any instruction or operation denied to a user mode program executing on a PDP–11 processor is not allowed in compatibility mode. For example, privileged instructions such as HALT and RESET are not permitted. Attempts to use a privileged instruction cause a trap to VAX–11 RSX and subsequent error messages.

Table 1–1 lists the VAX–11 compatibility mode instruction set.

Table 1–1 VAX–11 Compatibility Mode Instruction Set

Opcode	Mnemonic (octal)
000002	RTI
000006	RTT
0001DD	JMP
00020R	RTS
000240-000277	Condition codes
0003DD	SWAB
000400-003777	Branches
100000-103777	Branches
004RDD	JSR
.050DD	CLR(B)
.051DD	COM(B)
.052DD	INC(B)
.053DD	DEC(B)
.054DD	NEG(B)
.055DD	ADC(B)
.056DD	SBC(B)
.057DD	TST(B)
.060DD	RCR(B)
.061DD	ROL(B)
.062DD	ASR(B)
.063DD	ASL(B)
0065SS	MFPI[1]
0066DD	MTPI[1]
1065SS	MFPD[1]
1066DD	MTPD[1]

[1] The MFPI, MTPI, MFPD, and MTPD instructions execute exactly as they would on a PDP–11 in user mode with I-space and D-space overmapped: they ignore the previous access level and act like PUSH and POP instructions referring to the current stack.

Table 1–1 (Cont.) VAX–11 Compatibility Mode Instruction Set

Opcode	Mnemonic (octal)
0067DD	SXT
070RSS	MUL
071RSS	DIV
072RSS	ASH
073RSS	ASHC
074RSS	XOR
077RNN	SOB
.1SSDD	MOV(B)
.2SSDD	CMP(B)
.3SSDD	BIT(B)
.4SSDD	BIC(B)
.5SSDD	BIS(B)
06SSDD	ADD
16SSDD	SUB

1.3 VAX–11 RSX Compatibility With RSX–11 Task Execution

VAX–11 RSX supports the capabilities of RSX–11 by allowing the execution of RSX–11 task images. However, to run under VAX–11 RSX, a task image must meet the following requirements:

- It must adhere to the hardware requirements for compatibility mode.
- It must have been built by the Task Builder provided with RSX–11M Version 4.1, RSX–11M–PLUS Version 2.1, or Micro/RSX Version 1.1, or by the Task Builder provided with VAX–11 RSX.
- It must be executable in a mapped RSX–11 system.
- It must not depend on environmental features of RSX–11 that are not available in VAX–11 RSX, such as partitions, PLAS, or significant events. Section 3 describes the environmental differences between RSX–11 and VAX–11 RSX further.
- It must execute within the limitations of task/Executive interaction described in the *RSX–11M/M-PLUS and Micro/RSX Executive Reference Manual*. It must not be privileged for overmapping the RSX–11 Executive. The RSX–11 Executive is not present in a VAX–11 RSX system.
- It must not overmap the I/O page. The PDP–11 I/O page is not present in the VAX–11 RSX environment.
- It must not depend on the 32-word memory granularity of the KT11 memory management unit.
- It must not use any DECnet–11 functions.

- It must not depend on special memory management features available to RSX–11 privileged tasks. It can, however, perform privileged functions that do not involve mapping of the Executive.

Task images developed under RSX–11D or IAS that are compatible with RSX–11M Version 4.1, RSX–11M–PLUS Version 2.1, or Micro/RSX Version 1.1 can execute under VAX–11 RSX if they meet the requirements in the preceding list. However, RSX–11D or IAS task images must be rebuilt using the RSX–11M Version 4.1, RSX–11M–PLUS Version 2.1, or Micro/RSX Version 1.1 Task Builder or using the Task Builder provided by VAX–11 RSX. RSX–11 task images do not have to be rebuilt to run under VAX–11 RSX unless program modification or different Task Builder options are required.

RSX–11 task images must be rebuilt to run under VAX–11 RSX if the user wishes to use one or both of the following:

- The logical name extensions of FCS that allow the processing of VAX/VMS file specifications
- The extensions to FCS that allow correct propagation of Access Control Lists for files created on ODS–II disk volumes
- RMS–11

1.4 VAX–11 RSX Compatibility With RSX–11 System Directive Operation

In RSX–11, a task image interfaces with the operating system by issuing a directive request. RSX–11 responds by performing the desired function and then returning control to the task.

VAX–11 RSX duplicates the functions of most RSX–11 directives. When an RSX–11 task image issues a directive, the hardware traps to VAX–11 RSX. VAX–11 RSX responds to the request with either of the following results:

- The RSX–11 directive function is duplicated by VAX–11 RSX and the task continues execution.
- The RSX–11 directive function is not duplicated, but VAX–11 RSX does take whatever action is necessary to allow the task to continue execution.

When VAX–11 RSX cannot duplicate an RSX–11 directive, it is due to differences in the basic concepts of the two systems. For example, the RSX–11 capability to declare a significant event does not exist in VAX/VMS. Therefore, VAX–11 RSX cannot declare a significant event when it receives such a directive request. Rather, it performs no operation and returns a success status to the requesting task image, which continues execution.

Section 7 of this manual describes the details and implications of directive handlng in VAX–11 RSX.

1.5 Utilities, Components, and Files Available Under VAX–11 RSX

VAX–11 RSX provides the RSX–11 and RMS–11 utilities needed for a VAX–11 processor to function as the host for programs that are developed to run on a PDP–11 processor under RSX–11, on a VAX–11 processor under compatibility mode, or on both systems. Each utility runs in compatibility mode under VAX–11 RSX. These utilities provide the same functions they provide in RSX–11. In addition, VAX–11 RSX provides copies of the files and libraries needed for program development.

With the utilities and the files listed in the following sections, you can use VAX–11 RSX as you would use RSX–11 for program development. For example, you can enter and edit a program using EDI or native mode EDT, assemble it using the PDP–11 MACRO–11 Assembler (MACRO–11), build it using the Task Builder (TKB), and run it using VAX/VMS as the host system. Task images that are to be debugged using the On-Line Debugging Tool (ODT) and the MCR command DEBUG can be used with ODT.OBJ.

1.5.1 RSX–11 Utilities and Components

VAX–11 RSX provides the RSX–11 utilities listed in Table 1–2.

Table 1–2 RSX–11 Utilities and Components

RSX–11 Utility or Component	MCR Command	DCL Command
Bad Block Locator Utility	BAD	[1]
Backup and Restore Utility	BRU	[1]
Cross-Reference Processor	MAC/CR[2]	[1]
File Dump Utility	DMP	[1]
Disk Save and Compress Utility	DSC1	[1]
Line Text Editor	EDI	[1]

[1] To run this program from DCL, you can enter the DCL command MCR and then enter the acronym for the utility. You can also enter the command RUN SYS$SYSTEM:name where the RSX–11 utility name immediately follows the RUN SYS$SYSTEM: command.

[2] Generally, the Cross-Reference Processor is not invoked directly by the user.

Table 1–2 (Cont.) RSX–11 Utilities and Components

RSX–11 Utility or Component	MCR Command	DCL Command
File Transfer Program	FLX	[1]
Indirect Command Processor	@[3]	MCR ICM@[3]
Librarian Utility Program	LBR	LIBRARY /RSX11
PDP–11 MACRO–11 Assembler	MAC	MACRO/RSX11
Object Module Patch Utility	PAT	[1]
Peripheral Interchange Program	PIP	[1]
Source Language Input Program	SLP	EDIT/SLP
Task Builder	TKB	LINK/RSX11
Task/File Patch Program	ZAP	[1]

[1] To run this program from DCL, you can enter the DCL command MCR and then enter the acronym for the utility. You can also enter the command RUN SYS$SYSTEM:name where the RSX–11 utility name immediately follows the RUN SYS$SYSTEM: command.
[3] See Section 5 for more information.

The following list describes each RSX–11 utility or component. For a complete description of each, refer to the appropriate RSX–11 documentation. Indirect is documented in Section 5 of this manual.

- BAD—Bad Block Locator Utility. BAD determines the number and location of bad blocks on a volume.

- BRU—Backup and Restore Utility. BRU transfers files from a Files–11 Structure Level 1 volume to one or more backup volumes and retrieves files from the backup volume(s).

- CRF—Cross-Reference Processor. CRF produces cross-reference listings for the PDP–11 MACRO–11 Assembler and the Task Builder.

- DMP—File Dump Utility. DMP is a file listing program that allows you to examine the contents of a file or volume of files.

- DSC1—Disk Save and Compress Utility. DSC1 copies Files–11 Structure Level 1 disk files to tape or disk, and from DSC1-created tape back to disk.

- EDI—Line Text Editor. EDI is a line-oriented, interactive editor used to create and maintain text and source files.

- FLX—File Transfer Program. FLX is a file transfer and format conversion program that transfers files among DOS–11, RT–11, and Files–11 volumes.

- Indirect—Indirect Command Processor. Indirect processes files that contain Indirect, MCR, and utility commands. The task image name is ICM.EXE.

- LBR—Librarian Utility Program. LBR is a library maintenance program that creates and modifies library files.

- MACRO–11—PDP–11 MACRO–11 Assembler. MACRO–11 assembles one or more ASCII source files containing MACRO–11 statements into a single relocatable binary object file. The task image name is MAC.EXE.
- PAT—Object Module Patch Utility. PAT updates, or patches, relocatable binary object modules.
- PIP—Peripheral Interchange Program. PIP copies files and performs file maintenance functions.
- SLP—Source Language Input Program. SLP is a noninteractive editing program that is used to maintain and edit source files.
- TKB—Task Builder. TKB converts relocatable object modules into a single task image output file.
- ZAP—Task/File Patch Program. ZAP is a patch utility that examines and directly modifies locations in a task image file or data file.

1.5.2 RSX–11 Libraries and Files

VAX–11 RSX provides the following RSX–11 libraries and files:

- ODT.OBJ—On-Line Debugging Tool object module. ODT.OBJ is the DIGITAL-supplied object module built into an RSX–11 task image that requires debugging.
- QIOSYM.MSG—System message file. QIOSYM.MSG contains error messages and informational messages.
- RSXMAC.SML—System macro library. RSXMAC.SML contains the macro definitions for all system directives and File Control Services (FCS) file processing calls. It is the default library for MACRO–11.
- SYSLIB.OLB—System object library. SYSLIB.OLB contains register handling, arithmetic, data conversion, output formatting, FCS, and MCR command line processing subroutines. It is the default library for TKB.
- VMLIB.OLB—Virtual memory management object library. VMLIB.OLB contains dynamic memory, core allocation, virtual memory, and page management routines.

1.5.3 Additional RSX–11 Facilities

The following RSX–11 facilities can be performed under VAX–11 RSX:

- RSX–11M system generation
- RSX–11M–PLUS system generation
- RSX–11S system generation
- RSX–11M/RSX–11S DECnet–11 network generation
- Operation of various PDP–11 language compilers

To perform a DECnet–11 network generation, or an RSX–11M, RSX–11M–PLUS, or RSX–11S system generation, or to operate a PDP–11 language compiler, you must purchase a separate license for each component.

1.5.4 RMS–11 Utilities

The following RMS–11 Version 2.0 utilities are available on VAX–11 RSX:

- BCK Command Utility. RMS BCK transfers the contents of any RMS–11 disk file to another disk file or to a container file. A container file is a magnetic tape file containing RMS–11 files that were backed up using RMS BCK.
- CNV Command Utility. RMS CNV reads records from an RMS–11 file of any type and loads them into another RMS–11 file of any type.
- DEF Interactive Utility. RMS DEF allows you to define the attributes of a file. Although this utility has been superseded by RMS DES, it is still available and is described in the appropriate RMS–11 manual.
- DES File Design Utility. RMS DES allows you to design and create indexed, sequential, and relative files. You can either specify the file attributes interactively or read in the attributes of an existing, external data file that you want to re-create with little or no modification.
- DSP Command Utility. RMS DSP displays a concise description of any RMS–11 file, including RMS–11 backup files on disk or on magnetic tape (container files).
- IFL Command Utility. RMS IFL reads records from any RMS–11 file and loads them into an indexed output file.
- RST Command Utility. RMS RST transfers, to disk, files that were backed up on disk or magnetic tape using RMS BCK.

1.5.5 RMS–11 Libraries and Files

VAX–11 RSX provides the following RMS–11 Version 2.0 libraries and files:

- RMSLIB.OLB—Record Management Services object module library. RMSLIB.OLB contains the routines for sequential, relative, and indexed I/O.
- RMSMAC.MLB—Record Management Services macro library. RMSMAC.MLB contains the macro definitions for RMS–11.
- DAP11X.ODL—ODL file. DAP11X.ODL is structured to add about 14KB to the task size. This file provides sequential, relative, indexed, and (on systems with the required DECnet support) remote access facilities in 16 overlay segments.
- RMS11.ODL—Prototype ODL file. RMS11.ODL is similar to the standard RMS11X.ODL file except that it contains comments and instructions in addition to ODL statements.
- RMS11S.ODL—Standard ODL file. RMS11S.ODL is structured to add about 6.5KB to the task size. This file provides only sequential and relative file organization routines in 11 overlay segments.
- RMS11X.ODL—Standard ODL file. RMS11X.ODL is structured to add about 10KB to the task size. This file provides sequential, relative, and indexed file organization routines in 35 overlay segments.
- RMS12S.ODL—Standard ODL file. RMS12S.ODL is structured to add about 9KB to the task size. This file provides only sequential and relative file organization routines in 5 overlay segments.

- RMS12X.ODL—Standard ODL file. RMS12X.ODL is structured to add abdut 12KB to the task size. This file features sequential, relative, and indexed file organization in 13 overlay segments.

1.6 Micro / RSX File Transfer

VAX–11 RSX supports the Micro/RSX File Transfer Utility (MFT). MFT is a file transfer utility that transfers files from VAX–11 RSX to Micro/RSX.

For more information about MFT, refer to the appropriate Micro/RSX documentation.

1.7 FILES-11 and Transportability Considerations

Files–11, the file structure used for disk volumes under both RSX–11 and VAX/VMS, provides two structure levels: Files–11 Structure Level 1 and Files–11 Structure Level 2. RSX–11 supports only Files–11 Structure Level 1; VAX/VMS supports both structure levels.

The RSX–11 utilities, BRU and DSC1, operate only with Files–11 structure Level 1 volumes.

By default, VAX/VMS creates Files–11 Structure Level 2 volumes, but it can also create and read Files–11 Structure Level 1 volumes (using the INITIALIZE command and the keyword /STRUCTURE=number). Thus, Files–11 Structure Level 1 volumes can be transported among RSX–11, IAS, and VAX/VMS systems.

The UIC directory format ([g,m]) must be used for RSX–11M Version 4.1 and RSX–11M–PLUS Version 2.1 transportability.

For magnetic tape use, ANSI (17-character) file names are supported under compatibility mode.

1.8 FCS and RMS–11 Under VAX–11 RSX

File Control Services (FCS) and RMS–11 are available under VAX–11 RSX. RSX–11 task images that call FCS/RMS–11, interface with the same record management services they used in RSX–11. VAX–11 RSX converts FCS QIO requests from the RSX–11 format to VAX/VMS equivalent native mode QIOs.

Record locking is not supported for RSX–11 task images that use FCS or QIOs in compatibility mode. It is supported for tasks that use RMS–11.

2 The VAX–11 RSX User Environment

Using VAX–11 RSX MCR, you can do the following types of tasks:

- Specify file protection
- Assign logical names
- Prepare and run an RSX–11 task image or a VAX/VMS native image
- Execute RSX–11 indirect command files

This section explains how to log in to the VAX/VMS system and perform the first three tasks listed above. Section 5 describes how to execute RSX–11 indirect command files.

2.1 Logging in to VAX/VMS

There is no LOGIN command. Instead, you get the attention of the system from a terminal not currently in use by pressing the RETURN key (or press CTRL/C or CTRL/Y).

The system first prompts for your user name and then for your password:

```
Username:
Password:
```

Since your password is considered confidential, the system does not display it when you enter it.

Using your entry in the User Authorization File (UAF), the system verifies your right to gain access to the system and sets up the default characteristics for the terminal session. When you log in, the system may execute a number of login command files, including a command file that you created. Finally, the Command Language Interpreter (CLI) prompts for command input.

2.1.1 Selecting a Command Interpreter

The system uses the following factors to determine which command interpreter is to be associated with your process:

- The /CLI keyword that you specify after your user name to override the default command interpreter for the terminal session
- The default command interpreter named in your UAF entry

If the MCR command interpreter is not your customary default, you must enter your user name followed by /CLI=MCR, as follows:

```
Username: CALEB/CLI=MCR
```

Then enter your password in the normal manner.

Once you log in with MCR as your command interpreter, MCR remains in control of your process until you log out. However, you can access the DCL command interpreter when you are logged in with the MCR command interpreter if you specify the following:

```
> SPAWN/CLI=DCL
```

The SPAWN command runs a subprocess with DCL as the CLI.

2.1.2 LOGIN.CMD File

The LOGIN.CMD file used under VAX–11 RSX MCR is identical in function to the LOGIN.CMD file under RSX–11. If your default command interpreter is MCR, the MCR command interpreter searches your default directory for the file LOGIN.CMD. The LOGIN.CMD must contain valid VAX–11 RSX MCR commands and directives. If a login command file is specified in the UAF, then that login command file is executed instead of LOGIN.CMD.

If your default command interpreter is DCL, you must request MCR using the /CLI keyword to log in.

2.2 Terminal Functions

Several of the terminal functions are the same as those available under RSX–11. Under VAX–11 RSX, you do not have to wait for one command or program to terminate before typing the next request. Unprocessed commands are stored in the type-ahead buffer.

VAX–11 RSX also allows command line editing using a combination of control key sequences and function keys. These are described in the following subsection.

2.2.1 Terminal Function Keys and Control Key Sequences

Under MCR, only the last line you typed is recalled and edited. Table 2–1 provides information on the operation of the terminal function keys.

Table 2–1 Terminal Function Keys

Key	Function
RETURN	Before a terminal session, initiates the login sequence. (Carriage return) Transmits the current line to the system for processing.
Control Character	Define the functions to be performed. You enter a control character by pressing the CTRL key at the same time as you press a letter key.
CTRL/A	Toggles insert /overstrike mode for command line editing.[1]
CTRL/B	Recalls the last command typed for editing and execution.[1]

[1] The control character is effective only when the terminal is set for line edit mode (use the command SET TERMINAL/LINE_EDIT).

Table 2–1 (Cont.) Terminal Function Keys

Key	Function
CTRL/C	Before a terminal session, initiates a login sequence. During command entry, cancels command processing. Certain system and user programs provide special routines to handle CTRL/C interrupts. If CTRL/C is pressed to interrupt a program that does not handle CTRL/C, then CTRL/C has the same effect as CTRL/Y and echoes as "Interrupt." CTRL/C echoes as "Cancel" if there is no CTRL/C handler.
CTRL/D	Moves the cursor to the left.[1]
CTRL/E	Moves the cursor to the end of the command line.[1]
CTRL/F	Moves the cursor to the right.[1]
CTRL/I	Duplicates the function of the TAB key.
CTRL/O	Alternately suppresses and continues display of data at the terminal. CTRL/O echoes as "Output on" or "Output off."
CTRL/Q	Restarts terminal output that was suspended by CTRL/S.
CTRL/R	Retypes the current line during input and does not change cursor position in the line.
CTRL/S	Suspends terminal output until you press CTRL/Q.
CTRL/U	Cancels the current line to the left of the cursor and discards it.
CTRL/V	Allows use of characters other than RETURN as line terminators.[1]
CTRL/X	Cancels the current line to the left of the cursor and deletes data in the type-ahead buffer.
CTRL/Y	Interrupts command or program execution and returns control to the command interpreter. CTRL/Y echoes as "Interrupt."
CTRL/Z	Signals end-of-file for data entered from the terminal. CTRL/Z echoes as "Exit."
TAB	Moves the printing element or cursor on the terminal to the next tab stop. Most terminals have tab stops at every 8-character position on a line (for example, 1, 9, 17, 25,...).
BACKSPACE	Moves the cursor to the beginning of the command line.[1]

[1] The control character is effective only when the terminal is set for line edit mode (use the command SET TERMINAL/LINE_EDIT).

Table 2–1 (Cont.) Terminal Function Keys

Key	Function
DELETE	Deletes the last character entered at the terminal and backspaces over it. On some terminals this key is labeled RUBOUT. On other terminals this key is identified by a special graphic character that indicates the key's function.
ESCAPE	Has special uses for particular commands or programs, but generally performs the same function as RETURN. On some terminals, this key is labeled ALTMODE, ESC, or ESC(SEL). If your terminal is in line edit mode, the ESCAPE key is treated as the start of an escape sequence, not as a terminator.
LINE FEED	Deletes the previous word in a command line string and backspaces over it.[1]
–>	Moves the cursor to the right.[1]
<–	Moves the cursor to the left.[1]
^	Recall last command typed for editing and execution.[1]

[1] The control character is effective only when the terminal is set for line edit mode (use the command SET TERMINAL/LINE_EDIT).

2.3 Process and Image Naming Conventions

When you log in to VAX/VMS, the system automatically creates a process for you. That process provides the virtual address space and control information you need to perform interactive user operations. In other words, the **process** provides the environment within which you operate under VAX/VMS. Within your process, you run whatever programs you need to perform the desired functions. Because a process can exist without a program, VAX/VMS schedules processes instead of programs for execution. Your privileges, protection, and other information are associated with the process and, thus, contribute to the environment (context) within which an image executes.

Under VAX/VMS, an image is a program made executable as a result of linking either by the VAX/VMS Linker or by the RSX–11 Task Builder (TKB). The Linker, which turns programs into images, is itself an example of a native image that executes in native mode; the Peripheral Interchange Program (PIP) and TKB are examples of RSX–11 task images that execute in compatibility mode. You can run both native and RSX–11 task images within your process using MCR.

Images in VAX/VMS, like tasks in RSX–11, reside in files on disk. When you request the execution of an image (for example, by typing the RUN command), VAX/VMS activates that image in your process. As each subsequent image is requested, it overlays the previous image in the process's virtual address space. The sections that follow describe the features of a process that are significant for an RSX–11 user.

2.3.1 Login Process, Detached Process, or Subprocess

VAX/VMS recognizes the following types of processes:

- An **interactive process**, which is the process created for you when you log in to the system
- A **subprocess**, which is a process that you create to run a specified task image but that remains under the control of your login process
- A **detached process**, which is a fully independent process

Normally, you run task images in your login (interactive) process. Because that process is associated with you and your terminal, you can request only one task image at a time. In most program development sessions, in which the output from one step (for example, MACRO–11) is input to the next (for example, TKB), this serial approach is convenient.

However, in an application in which task images must run concurrently to coordinate their activities, each task image must run in a separate process. Therefore, the MCR command RUN allows you to create subprocesses and detached processes in addition to running task images in your current process.

Several other process types run on VAX/VMS (for example, a batch or network process). These and other process types are described in the appropriate VAX/VMS documentation.

2.3.2 Process Name

Every process has a process name that is created or modified as a result of one of the following events:

- When you log in, VAX/VMS defines a process name as your user name (for example, FELIX).
- When a native image running in your process issues a system service request to change the process name, a new name is created.
- When you enter the SET PROCESS/NAME=name CLI command, the name is changed to the name you specify.
- When you run an RSX–11 task image, VAX–11 RSX may modify your process name.

VAX–11 RSX will modify your process name when you run an RSX–11 task image under the following conditions:

1. If the task has a task name that is not a multiuser name (that is, if the first three characters do not form an ellipsis (. . .)) in its task image label block, the task name becomes the process name.
2. If the first condition is not true, but the task is running from a batch job with a default process name, then VAX–11 RSX will modify the process name. VAX–11 RSX forms a new process name of the form "Bn" where "n" is batch sequence number or the low-order five digits of the job sequence number if the number is longer than five digits.

3 If neither of the previous conditions applies, but your current process name contains characters that are not uppercase letters, digits, dollar signs ($), or a dot (.), or if the name is longer than six characters, VAX–11 RSX will modify the process name as follows:

- By converting all lowercase characters to uppercase
- By removing all characters that are digits, dollar signs ($), or dots (.)
- By truncating the resulting string to six characters
- If the resulting six-character name is not unique, by truncating the name to five characters and appending one digit (0–9)

If VAX–11 RSX modifies the process name to run an RSX–11 task, it restores the original process name when the task completes execution. Normally, a process name remains for the duration of the process or until a new name is specified.

A process running a native image can create a subprocess or a detached process, assign it a process name, and designate an image that the process is to execute. Thus, a process can create a named subprocess or detached process that executes an RSX–11 task image. Once the process is created, other processes can issue system service requests in native mode or directive requests in compatibility mode that designate the process as the target. The creator of a subprocess is always allowed to affect the subprocess. Any process with a matching UIC can affect the subprocesses. Other processes and subprocesses must have either group or world privilege to affect the subprocess.

VAX/VMS qualifies a process name by the group number of its UIC. Therefore, process names need to be unique only within the group.

If an RSX–11 task image is to issue directives that specify a process executing a native image as the target, you must be aware of the difference in the allowable lengths of task names and process names. A task name has a maximum length of 6 (Radix–50) characters. A process name has a maximum length of 15 ASCII characters. Therefore, if an RSX–11 task image is to refer to a process running a native image, RSX–11 multiuser, or unnamed task, that process's name must not exceed 6 characters. The characters you select for the process name should be within the set allowed for Radix–50 character strings.

2.3.3 Task Name

An RSX–11 task image should have a task name in its label block to provide a name for its process if any of the following are to occur:

- The task image is to receive data using the Receive Data, Receive Data Or Exit, or Receive Data Or Stop directives.
- The task image is to cooperate with other task images using event flags.
- The process containing the task image is to be the target of directive action; for example, the process is to be requested or resumed.

Each of the following RSX–11 directives accepts a task name as an argument:

- Abort Task (ABRT$)
- Cancel Time Based Initiation Requests (CSRQ$)

- Receive Data (RCVD$)
- Receive Data or Exit (RCVX$)
- Receive Data or Stop (RCST$)
- Request Task (RQST$)
- Resume Task (RSUM$)
- Run Task (RUN$)
- Send Data (SDAT$)
- Spawn Task (SPWN$)

VAX–11 RSX supports the RSX–11 convention of naming multiuser MCR tasks with a string that starts with an ellipsis, for example, . . . PIP. When VAX–11 RSX encounters a task image with a task name of this type, it recognizes that the task image can be run by more than one user simultaneously. Any task that is to affect a multiuser or unnamed task with the previously listed directives must use the target task's VAX/VMS process name as the task name. VAX–11 RSX can modify the process name, as described in 2.3.2.

Because VAX–11 RSX does not incorporate the concept of an installed task, an RSX–11 task image cannot acquire a task name except by task building.

2.3.4 Process Identification

VAX/VMS assigns every process and subprocess a process identification number (pid) that is unique throughout the system. The identification of a process does not change as long as the process exists.

You must specify a process identification number to affect processes outside your group.

You can use the SHOW PROCESS command to display your process name and identification. You can use the SHOW SYSTEM command, or the SHOW PROCESS command with the /IDENT keyword, to display the name and identification of other processes.

2.4 User Protection and Privilege Features

The system manager maintains a file called the User Authorization File (UAF). It contains one entry for each user who is allowed access to the system. Each entry includes the following information:

- User name
- User password
- User Identification Code (UIC)
- User account name
- User default disk device
- User default command interpreter
- User default command tables

- User's default directory name
- Name of an indirect command file or command procedure to execute at login (for example, LOGIN.CMD if the default command interpreter is MCR)

Note: **If no login command procedure is specified, then LOGIN is the default file name and CMD or COM is the default file type (depending on whether MCR or DCL is the CLI).**

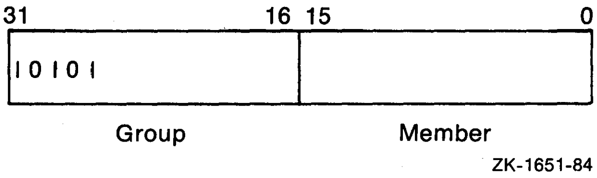
- User privileges allowed
- User access rights
- Process base priority
- Amount of resources available to user

When you log in, VAX/VMS uses your UAF entry to associate your UIC, privileges, and priority with the process it creates for you.

2.4.1 VAX/VMS UICs, RSX–11 UICs, and Mapping Considerations

Under VAX/VMS, a UIC is a unique 32-bit numeric identification of a particular user. It consists of a 14-bit octal group number and a 16-bit octal member number, as shown in Figure 2–1. The two high bits are set to zero, indicating that the value is a UIC. UICs form a proper subset of the set of VAX/VMS identifiers. (Refer to the appropriate VAX/VMS documentation for more information on identifiers.)

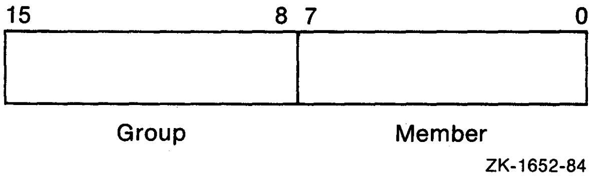
Figure 2–1 Format of VAX/VMS UICs



Note: **If your UIC is displayed by the system in alphanumeric format, refer to your system manager for the numeric value.**

Under RSX–11, a UIC is a unique 16-bit numeric identification of a particular user. It consists of an 8-bit octal group number and an 8-bit octal member number, as shown in Figure 2–2.

Figure 2–2 Format of RSX–11 UICs



Group and member numbers for the RSX–11 form of the UIC range from 0 to 377 (8). Group numbers for the VAX/VMS form of the UIC range from 0 to 37777 (8) and member numbers range from 0 to 177777 (8). Under VAX–11 RSX, UICs that contain a group or a member number exceeding 377 (8) are known as **long UICs,** a definition used in the remainder of this section.

2.4.1.1 UIC Mapping Concepts

The data structures and coding assumptions in the RSX–11 environment depend on a UIC's being limited to 16 bits in length. As a result, an RSX–11 task cannot accept a UIC greater than [377,377]. As a result, under VAX–11 RSX, all UICs with either a group or member number greater than or equal to 1000 are mapped to [377,377]. This method of reporting a process's UIC to an RSX–11 task image hides the existence of a long UIC from the task image.

2.4.1.2 Mapping Example

An RSX–11 task image can gain access to the UIC of its process by issuing a Get Task Parameters (GTSK$) directive. Two UICs are returned in the Get Task Parameters buffer as follows:

Protection UIC:

- If the process's UIC is not a long UIC, then the process UIC is always returned unmodified in word 17.
- If the process's UIC is a long UIC, then the UIC [377,377] is always returned in word 17.

Default UIC:

- If the process's default directory is in UIC format, then that UIC is returned in word 7. (A default directory format is of the form [g,m], where ggg and mmm are octal numbers from 0 to 377 (8).)
- If the process's default directory is not in UIC format, then the protection UIC is returned in word 7. VAX/VMS does not allow directories in long UIC format. The highest numeric directory allowed is [377,377].

2.4.1.3 Notes on UICs and Numeric Directories

Because the UIC [377,377] does not uniquely identify a user, it should not be assigned by the system manager to any user on a system in which VAX–11 RSX is installed.

The UICs returned in the Get Task Parameters directive buffer are made available for informational purposes only. The RSX–11 task image cannot use them to affect group protection or file protection.

An RSX–11 task image cannot assume that its default directory is related to its process UIC. VAX–11 RSX provides a special directive that is used by the File Control Services (FCS) and RMS–11 to access the actual default directory.

2.4.1.4 Process Protection

VAX/VMS uses a process's UIC, with the privileges assigned to it by the system manager, to control access to system services that affect other processes in the system. Such control provides protection by determining which processes you can affect using any of the following:

- An MCR or DCL command that accepts a process name as a parameter (for example, the commands CANCEL and STOP).
- A native image that issues system services specifying another process as the target (for example, the Suspend Process system service).
- An RSX–11 task image that issues directives specifying another process as the target (for example, the directive Abort Task (ABRT$)). VAX–11 RSX translates these directives into appropriate system services that are subject to restrictions based on the process UIC or privileges. VAX/VMS uses the process UIC when checking for restrictions on the system service, regardless of the UIC's length. The fact that long UICs are hidden from RSX–11 task images is irrelevent to these checks.

A process can always affect the subprocesses it creates. In addition, by using UIC-based process protection in conjunction with process privileges, VAX/VMS allows a process to affect other processes in the following categories:

- Processes having the same process UIC group number as the requesting process (with GROUP privilege)
- All processes in the system regardless of process UIC (with WORLD privilege)

2.4.1.5 File Ownership and Protection

When a file is created on a Files–11 Structure Level 2 volume, the file's owner UIC is set equal to the creating process's UIC. The file owner UIC field for a file created on a volume is 32 bits wide. The process's UIC is always used as the file's owner UIC, even if it is a long UIC.

When a file is created on a Files–11 Structure Level 1 volume (RSX–11 understands only Files–11 Structure Level 1) and the process UIC is not a long UIC, the file's owner UIC is set equal to the creating process's UIC. The file owner field for a file created on a Files–11 Structure Level 1 volume is 16 bits wide. When the process's UIC is long, then VAX/VMS sets the file's owner UIC to [377,377], the maximum UIC that fits into 16 bits. Any process with a long UIC is considered an owner of the Files–11 Structure Level 1 file.

If a file on a Files–11 Structure Level 1 volume is being accessed and the process's UIC is long, then VAX/VMS temporarily maps the process's UIC to the value [377,377]. The UIC is then subject to the file's protection mask. This behavior is identical to the procedure just noted for creating Files–11 Structure Level 1 files from processes with a long UIC. For more information on file protection specification, see 2.5.5.

You may specify read, write, execute, and delete file protection for system, owner, group, and world access based on the process's UIC. This type of file protection is similar to that found under RSX–11. Like protection under RSX–11, VAX/VMS protection for each user category is represented by the four 4-bit fields RWED. The only difference in the interpretation of VAX/VMS and RSX–11 protection is in the E field. Under VAX/VMS, this field determines whether a user has the right to execute files containing executable program images (either VAX–11 or RSX–11 images). When applied to an entire volume, the E field determines whether users can create files on that volume. Under RSX–11, the E field determines whether a file can be extended.

Under VAX/VMS, the process UIC is not equivalent to the process default directory name. That directory name is specified as a separate entry in the UAF. Refer to 2.9.1 for further discussion of directory names.

2.4.2 Privileges

RSX–11 recognizes users as privileged or nonprivileged on the basis of the group number of the UIC. Group numbers equal to or less than 10 (8) are privileged. Once you log in, RSX–11 associates the privileged or nonprivileged status with the terminal from which you are issuing commands.

VAX–11 RSX returns the RSX–11 Directive Status Word (DSW) return code IE.PRI to an RSX–11 task image requesting a function for which it does not have the appropriate privilege. The individual directive descriptions in Section 7 indicate the DSW codes returned for each directive.

VAX/VMS privileges differ from RSX–11 privileges in two respects:

- User privileges are associated with the process VAX/VMS creates for you rather than being associated with your terminal.
- Users are not categorized as privileged or nonprivileged; rather, the VAX/VMS system manager can associate any of a wide range of privileges with each user, depending on that user's needs.

VAX/VMS controls the functions you are allowed to request on the basis of the privileges specified in your UAF entry. MCR users usually require more privileges than DCL users to execute RSX–11 procedures (for example, to run the system generation procedure).

The individual command descriptions in Section 4 indicate the privileges required for each command. If you attempt to use a command for which you do not have the appropriate privilege, the system issues an error message. Error messages are described and listed in Section 9 of this manual. Additional information on user privileges is provided in the appropriate VAX/VMS documentation.

2.4.3 Process Priority

The UAF file entry also specifies the priority of your process. VAX/VMS priorities range from 1 through 15 for normal processes and from 16 through 31 for real-time processes. Under VAX/VMS, all images execute at the requesting user's process priority. On the other hand, in RSX–11, a task has a priority assigned to it.

2.4.4 Resource Usage Limits

VAX/VMS controls a process's use of system resources by enforcing usage limits defined in the UAF entry. The following resource quotas may be relevant to an RSX–11 task image running under VAX–11 RSX:

- Number of active buffered I/O requests
- Number of bytes of system dynamic memory used for buffered I/O
- Number of active direct I/O requests
- Number of files open simultaneously
- Disk quotas

By default, VAX/VMS places an image that attempts to exceed a resource limit in a wait state until the function can be accomplished without exceedin the limit (for example, until other active I/O requests have completed). Native images can disable and enable resource waiting. If an RSX–11 task image attempts to exceed a limit when resource waiting is disabled, the task image receives a DSW code of IE.UPN (insufficient dynamic memory). The DCL and MCR commands RUN provide an option for controlling resource wait mode for subprocesses and detached processes.

2.5 Command Line Syntax Rules

VAX–11 RSX MCR allows you to enter a command on multiple lines, place comments on a command line, and abbreviate command names.

2.5.1 Continuing Commands on More Than One Line

You can enter a command string on more than one line by using the continuation character, a hyphen (-), as the last element on a line. For example:

```
> COPY/LOG-
>-SCITUATE.DAT,COHASSET.DAT,HINGHAM.DAT,DUXBURY.DAT-
>-SOUTHSHORE.DAT
```

No restriction is placed on the number of lines you can use to enter a command string.

After each continued line, MCR prompts for additional input (> _).

2.5.2 Using Comments in Command Lines

VAX–11 RSX MCR distinguishes command lines from comments using the conventions of RSX–11M–PLUS, as follows:

- A semicolon (;) as the first significant character on a line indicates that the entire line is a comment. A full-line comment cannot be continued with the hyphen (-).
- An exclamation point (!) usually indicates the beginning of a comment that ends a command line. An exclamation point can also indicate the beginning and end of a comment embedded within the actual command text. Embedded comments and the command they describe can be continued over multiple lines. For example:

```
> DIRECTORY/FULL !VIEW FILE PROTECTION FOR THESE-
>_FILES!AVDEGREE.DAT,!DEGREE DAY DATA AVERAGE!-
>_CURDEGREE.DAT!THIS YEAR'S ACTUAL DATA
```

2.5.3 Truncating Command Names and Keywords

All MCR command names can be truncated to their first four characters. You can truncate command names to fewer than four characters as long as the name remains unique. For example, because the TYPE command is the only command beginning with the letters TY, you can truncate it to two letters. For convenience, you can truncate the RUN command to one letter.

You can truncate all other portions of a command line (for example, keywords and keyword values) as long as they remain unique within the command.

2.5.4 Placing Keywords

VAX–11 RSX MCR is more flexible than RSX–11 MCR about the placement of keywords in the command line. MCR allows the use of command and file keywords. Command keywords have the same meaning whether they appear after the command name or after a command parameter. For example, the following two commands are identical:

```
> RUN /UIC=[200,230] DOTS.TSK
> RUN DOTS.TSK /UIC=[200,230]
```

File keywords, on the other hand, have different meanings, depending on their placement in the command line. If specified immediately after a file specification, they affect only the file so qualified. If specified after the command name, they affect all files specified as parameters. For example, the first command that follows results in the printing of two copies of both files, whereas the second results in the printing of two copies of SANDPIPER.MAP and one copy of YELLOWLEG.MAP:

```
> PRINT/COPIES=2 SANDPIPER.MAP,YELLOWLEG.MAP
> PRINT SANDPIPER.MAP/COPIES=2,YELLOWLEG.MAP
```

Some file keywords are not valid as command keywords. Individual keywords described in Section 4 list any restrictions.

2.5.5 Specifying File Protection

VAX–11 RSX supports two types of file protection: UIC-based file protection and ACL-based file protection. These types of protection are described in the following subsections.

2.5.5.1 UIC-Based File Protection

All disk and tape volumes have protection codes that restrict access to them Protection is assigned when the volume is initialized and can be reassigned when a volume is mounted. For disk volumes, each file on the volume, including the directory file, can have a different protection associated with it

Volumes and files can be accessed by users in four categories:

- **SYSTEM**—all users who have low group numbers, usually 1 through 10 (8) or users who have SYSPRV, BYPASS, or READALL privilege. The exact range of group numbers is determined by the system manager usin the VAX/VMS system generation utility (SYSGEN) and may range from 1 to 37777 (8). These group numbers and privileges are generally for system managers, system programmers, and operators.
- **OWNER**—the UIC of the person who created and, therefore, owns the volume or file. For more information on file ownership, see 2.4.1.2.
- **GROUP**—all users who have the same group number in their UICs as th owner of the volume or file.
- **WORLD**—all users who do not fall into any of the other three categories

Each of these categories of users can be allowed or denied the following type of access:

- **READ**—the right to examine, print, or copy a file or files on a volume.
- **WRITE**—the right to modify files or to write files on a volume.
- **EXECUTE**—the right to execute files that contain executable program images. In applying protection to an entire volume, this field is interprete as the right to create files on the volume.
- **DELETE**—the right to delete a file or files on a volume.

There are no restrictions on the order in which you specify user categories and protection types.

When you specify a protection code, you must:

- Abbreviate protection types to one character.
- Omit a user category to preserve current access rights for that category.
- Omit a protection type for a user category to deny that type of access.
- Separate each user category from the access type with a colon (:) or equal sign (=).
- Specify the category without listing any protection types (with or without a trailing colon) to deny all access.
- Separate each category with a comma (,) and enclose the entire code specification in parentheses to specify more than one category.

When you specify volume protection, omitting a category denies that user category all access.

2.5.5.2 ACL-Based File Protection

An **Access Control List (ACL)** is an ordered list of **Access Control Elements (ACE)**, each of which specifies the access allowed a user or a particular group of users. User groups are selected based on the UIC group or member number, or the access rights identifiers held by users. Access rights identifiers control file access for all users. The system manager can assign additional access rights to any user.

Any file or directory on a Files–11 Structure Level 2 disk may have an ACL associated with it. When an ACL is specified for a file, it replaces the UIC-based file protection for the file.

When a new file is created in a directory, it acquires the directory's default ACLs. When a new version of a file is created in a directory, it acquires the ACL of the previous version of the file. To determine what the directory is at the time of file creation, VAX/VMS uses a function that both creates and enters the file. The version of File Control Services (FCS) that is distributed with VAX–11 RSX uses this function (see Section 3 for more information).

ACLs are set or modified by the following:

- The MCR or DCL command SET FILE/ACL
- The DCL command EDIT/ACL
- VAX/VMS utilities
- VAX/VMS system services (which are called from native mode programs)

Refer to the appropriate VAX/VMS documentation for information on using any of the previously mentioned commands, utilities, and system services.

Compatibility mode tasks cannot directly examine or modify ACLs. However, if a task uses the VAX–11 RSX version of FCS-11 or RMS–11 to create files, or the appropriate Ancillary Control Processor (ACP) QIO functions, the VAX/VMS file system will propagate ACLs to new file versions.

2.5.6 Entering Dates and Times

When a command accepts a keyword specifying a time value, the time value is an absolute time, a delta time or a combination:

- An absolute time is a specific date and time of day (for example, 20-DEC-1984 10:53:22.10).
- A delta time is a future offset from the current date and time of day (for example, two days and five hours from now).
- An absolute and delta time combination consists of an absolute time value plus or minus a delta time value (for example, tomorrow plus one day).

2.5.6.1 Absolute Time

An absolute time has the following format:

```
[dd-mmm-yyyy[:]][hh:mm:ss.cc]
```

You can specify the date, the time, or both. The variable fields are as follows:

Field	Meaning
dd	Day of month (1 through 31)
mmm	Month; the month must be specified as one of the following three-character abbreviations: JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC
yyyy	Year
hh	Hour of the day (0 through 23)
mm	Minute of the hour (0 through 59)
ss	Seconds (0 through 59)
cc	Hundredths of seconds (0 through 99)

You may also specify one of the following keywords whenever an absolute time is appropriate:

Keyword	Description
TODAY	The current day, month, and year at 00:00:00:0 o'clock
TOMORROW	24 hours after 00:00:00:0 o'clock today
YESTERDAY	24 hours before 00:00:00:0 o'clock today

The punctuation marks indicate how the system interprets the time value you enter:

- If you specify both the date (dd-mmm-yyyy) and the time (hh:mm:ss.c), you must type a colon between the date and the time.
- You can truncate the date and the time on the right. However, if you are specifying both a date and a time, the date format must contain at least one hyphen.
- You can omit any of the fields within the date or time as long as you type the punctuation marks; the system supplies default values.
- The period between seconds and hundredths of seconds is a delimiter. It is not a decimal point.

When you omit the date or any of its fields from an absolute time value, the system supplies the current day, month, and year by default.

When you omit any fields from the time, the system supplies a value of 0 for the field.

Examples:

Time Specification	Result
28-DEC-1984:12	12:00 noon on December 28, 1984
28-DEC	Midnight (00:00 o'clock) on the 28th of December this year
15	3:00 P.M. today
15-	The 15th day of the current month and year at midnight
18:30	6:30 P.M. today
15–::30	00:30 o'clock on the 15th day of the current month

When you specify an absolute time that has already passed, the system performs the request immediately.

2.5.6.2 Delta Time

A delta time has the format:

```
[dddd-][hh:mm:ss.cc]
```

The variable fields are as follows:

Field	Meaning
dddd	Number of days, 24-hour units (0 through 9999)
hh	Number of hours (0 through 23)
mm	Number of minutes (0 through 59)
ss	Number of seconds
cc	Number of hundredths of seconds (0 through 99)

When you specify a delta time value, you can truncate the time field on the right. You can also omit any of the variable fields as long as you supply the punctuation marks. When any field is omitted from a delta time value, the system supplies a value of 0 for the field.

Examples:

Time Specification	Result
3-	3 days from now (72 hours)
3	3 hours from now
:30	30 minutes from now
3-:30	3 days and 30 minutes from now
15:30	15 hours and 30 minutes from now

2.5.6.3 Absolute and Delta Time Combinations

An absolute and delta time combination is an absolute time plus (+) or minu (−) a delta time value. If you omit the absolute time value, the delta time is offset from the current date and time of day. The delta time value must always be preceded by a plus or minus sign.

The variable fields and default fields for absolute time values are the same as those described in the preceding sections.

Whenever a plus sign (+) precedes the delta time value, the entire time specification must be enclosed in quotation marks ("). However, if a minus sign (−) is used, no quotation marks are needed.

Examples:

Time Specification	Result
"+5"	Current time plus 5 hours
"+:5"	Current time plus 5 minutes
"TOMORROW+1-00:00"	Tomorrow plus 1 day
–:5	Current time minus 5 minutes

2.6 Logical Name Types and Uses

The VAX/VMS logical name capability is an extension of the capability available under RSX–11. Using the VAX–11 RSX MCR commands ASN (assign) and DEFINE, you can assign a logical name to the following:

- A physical device name, logical device name, or pseudodevice name
- A file specification or a portion of one

Note that VAX–11 RSX compatibility mode tasks cannot handle logical names that translate to search lists.

VAX/VMS categorizes logical names differently from RSX–11. Rather than limiting them to login, local, and global logical names, VAX/VMS allows logical names to be defined in many different logical name tables. VAX/VMS defines four standard types of logical name tables:

1. **Process**—available only to individual processes
2. **Job**—available to all processes in a job, the main process, and all the subprocesses
3. **Group**—available to the group
4. **System**—available to the entire system

VAX/VMS also allows you to create logical name tables and to specify to whom they are available (through the use of Access Control Lists).

The SHOW LOGICAL command displays the logical names and their equivalent according to the table specified.

When VAX–11 RSX encounters a logical name in a command or program, it translates the logical name to its equivalence name. To do so, it searches the tables specified in the default search list in the order in which they are specified and uses the first match it finds. Once VAX–11 RSX has translated a logical name, it applies any appropriate file specification defaults.

Refer to 2.10.3 of this manual and the *VAX/VMS DCL Dictionary* for further discussion of the use of logical names.

2.6.1 Process Logical Name Table

Each process logical name table contains the logical names local to each process. Once an entry is assigned to the process logical name table, it is available to all task images that run in the process until it is deassigned, or until the process is deleted (for example, as a result of logging out).

2.6.2 Job Logical Name Table

Logical names are shared by an interactive or batch process and all its subprocesses. The job logical name table is deleted when the interactive or batch process terminates.

No privilege is required for any process in the job to insert, translate, or delete logical names in this table.

2.6.3 Group Logical Name Table

Each group logical name table contains the logical names available to all processes that have the same group number in their UICs as the process that created the entry. Unlike RSX–11, VAX/VMS does not use the identification of the initiating terminal to determine whether a process has access to the group logical name table. Entries remain in this table until they are explicitly deleted.

The GRPNAM privilege is required for a user to place a name in or delete a name from the group logical name table.

2.6.4 System Logical Name Table

The system logical name table contains entries available to all processes in the system. Entries in this table are similar to global assignments under RSX–11. Entries remain in the table until they are explicitly deleted.

A user must have the SYSNAM privilege to place a name in or delete a name from the system logical name table.

2.6.5 System-Defined Logical Names

VAX/VMS defines logical names at the process and system levels. The system assigns logical names for every process created during login or for execution of a batch job. VAX/VMS automatically defines logical names for each process logical name table. Refer to the appropriate VAX/VMS documentation for a list of logical names.

2.7 Device Name Mapping

VAX–11 RSX automatically performs mapping to a physical device. (Note that this restriction is not applicable if you are using RMS–11.)

VAX–11 RSX automatically performs mapping for three categories of device names:

- RSX–11 physical device names (for example, DB2)
- VAX/VMS physical device names (for example, DBB2)
- RSX–11 pseudodevice (logical) names (for example, TI)

2.7.1 Mapping RSX–11 Device Names to VAX/VMS Physical Device Names

You can control the association of an RSX–11 physical device name with an actual VAX/VMS device, or VAX–11 RSX can perform the association automatically. To associate an RSX–11 device name with a specific VAX/VM device unit, issue an MCR ASN (Assign) or MOUNT command that specifie the RSX–11 device name as the logical name for the device.

If you do not use logical names, VAX–11 RSX automatically maps the RSX–11 device name to a VAX/VMS device name. VAX–11 RSX performs the mapping by retaining the device type (for example, MT) and converting the RSX–11 unit number into the corresponding VAX/VMS controller letter and unit number.

VAX–11 RSX performs the mapping to a physical device by converting the RSX–11 octal unit number to decimal and dividing it by 16 (10). The quotient is added to the ASCII value representing the character A. The result is the controller letter. The remainder becomes the VAX/VMS decimal unit number. (A mailbox is an exception; its device name is MBAn, where n is its unit number. Other exceptions are RTAn, NVAn, and LTAn.) For example, RSX–11 devices TT0 and DB22 become VAX/VMS devices TTA0 and DBB2, as follows:

Example:

To map RSX–11 device TT0 to VAX/VMS device TTA0:

```
1. VAX-11 RSX checks for a logical name previously defined for
   TTO.
2. If no logical name is found, the following algorithm is
   implemented:
     Convert octal 0 to decimal = 0
     Controller and unit ="A" + (0/16) = "A" + 0 with a remainder of 0
                           "A" + 0 = "A" = controller
                              0 = unit number
```

To map RSX–11 device DB22 to VAX/VMS device DBB2:

```
1. VAX-11 RSX checks for a logical name previously defined for
   DB22.
2. If no logical name is found, the following algorithm is
   implemented:
     Convert octal 22 to decimal = 18
     Controller and unit="A" + (18/16) = "A" + 1 with a remainder of 2
                          "A" + 1 = "B" = controller
                             2 = unit number
```

VAX–11 RSX performs this conversion during assignment of an I/O device to an RSX–11 task image when no logical name exists.

2.7.2 Mapping VAX/VMS Physical Device Names to RSX–11 Device Names

For this discussion, consider a VAX/VMS device name to be a VAX/VMS physical device name (for example, DRA0: or HSC01$DU100:), including any associated root directory name. Such associations are made with logical names. For example, the RSX–11 pseudodevice LB: is a VAX/VMS logical name, the translation of which includes a VAX/VMS physical device name that is the translation of SYS$SYSDEVICE:, and a root directory that refers to the current VAX/VMS system ([SYS0.], [SYS1.], . . . [SYSF.]). For this discussion, consider LB: to be the VAX/VMS device name. An example of this (that is not from RSX–11) is the logical name SYS$SYSROOT:. The VAX/VMS physical device name and the root directory name are inseparable.

When VAX–11 RSX attempts to convert a VAX/VMS device name into a format acceptable to a compatibility mode task, it must consider that many VAX/VMS device names are not directly representable to an RSX–11 task. A VAX/VMS device name can be longer than two characters (it includes a controller letter, and possibly a VAXcluster node name and a root directory name), and the unit number can be greater than 255 (10) or 377 (8). The RSX–11 environment is limited architecturally to only the two ASCII characters for the device name and unit numbers that fit into a byte.

VAX–11 RSX implements the following algorithm in an attempt to map the larger VAX/VMS device name space into a device name that an RSX–11 task can accept. The Applications Migration Executive (AME) translates a predefined series of logical names, looking for the first of this series that accurately represents the VAX/VMS device name being mapped. Each of the logical names in this series has a format that can be expressed in the RSX–11 device format of two ASCII characters ($$) and a unit number of 0 to 377 (8). If none of the logical names in the series accurately represents the VAX/VMS device name being mapped, then a direct mapping (described in detail in 2.7.2.5) is attempted.

The sequence of events that the AME follows is summarized as follows:

1. Attempt to match the VAX/VMS device name against each device already assigned to a Logical Unit Number (LUN) (see 2.7.2.1).
2. Attempt to match the VAX/VMSdevice name against the translation of the standard VAX–11 RSX systemwide logical names LB or SP (see 2.7.2.2).
3. Attempt to match the VAX/VMS device name against the translation of any of the logical names in the series $$0 to $$377 (8) (see 2.7.2.3).
4. Attempt to match the VAX/VMS device name against the translation of VAX–11 RSX internal logical names TI, SY, CO, CL, or OV (see 2.7.2.4).
5. Attempt to convert the VAX/VMS device name to an RSX–11 device name by merging the controller letter into the unit number (see 2.7.2.5).
6. Store the results of the mapping in the VAX–11 RSX LUN table.
7. Return the appropriate RSX–11 device name to the compatibility mode task issuing a Get LUN Information directive.

If the VAX/VMS device name cannot be mapped successfully, then the directive that presented the device name to VAX–11 RSX (such as Assign LUN or Extended Logical Parse) will fail with the error status IE.IDU (invalid device unit) or IO.BDV (bad device name).

2.7.2.1 Logical Unit Table

The first step that VAX–11 RSX takes to map the VAX/VMS device name is to examine the LUN table. The RSX–11 task may already have a LUN assigned to the VAX/VMS device name being mapped. To determine this, the VAX–11 RSX AME searches the logical unit table for a LUN assigned to that device. If such an assignment is found, the AME uses the RSX–11 device name stored in the LUN data structure at the time of the assignment. This caching technique accelerates the mapping procedure for frequently referenced devices.

2.7.2.2 System Defined Logical Names

If the preceding step fails to map the VAX/VMS device name, the VAX–11 RSX AME compares the name to the translations of the standard VAX–11 RSX systemwide logical names LB and SP. If the translation of either of these logical names accurately represents the VAX/VMS device name being mapped, then that logical name, either LB0 or SP0, is used as the RSX–11 representation for that particular VAX/VMS device name.

2.7.2.3 $$n Logical Names

If the VAX/VMS device name is not accurately represented by any assigned LUN or by the translation of LB, or SP, the VAX–11 RSX AME will begin translating each logical name in a predefined sequence until either the entire sequence is exhausted, or a logical name is found with a translation that accurately represents the VAX/VMS device name. This predefined sequence involves all logical names that consist of the two ASCII characters "$$" followed by an octal unit number ranging from 0 to 377. These logical names are referred to as "$$n logical names" (for example, $$0, $$1, and $$377).

The $$n logical names can be defined by any of the following:

- The DCL command DEFINE
- The DCL command ASSIGN
- The MCR command DEFINE
- The MCR command ASN

The $$n logical names can be placed in any desired logical name table (shared or private) as long as that logical name table is included in the logical name search list LNM$FILE_DEV.

The VAX–11 RSX AME translates each of the logical names in sequence, from unit number 0 to unit number 377. The first logical name in the sequence that translates to the VAX/VMS device name being mapped is used as the RSX–11 device name. For example, if $$100 is defined by the following command:

```
> DEFINE $$100 DRA1:[FOO.]
```

the VAX/VMS device name is DRA1:[FOO.] or DRA1:<FOO.>.

In general, when a VAX/VMS device fits one of the following qualifications and is intended for use by compatibility mode tasks, a $$n logical name must be defined for it, unless the VAX/VMS device is SY or LB:

- The VAX/VMS device name includes a root directory name.

- The VAX/VMS physical device name has a unit number higher than 15 (10) or an equivalent RSX–11 unit number (described in 2.7.2.5) that is higher than 377 (8).
- The VAX/VMS physical device name is in a VAXcluster.

Restrictions apply to compatibility mode utilities that depend on the device name to determine the device type. These restrictions are described in 2.7.2.5.

If $$n logical names are defined for VAXcluster shareable devices, ideally the same $$n logical name should be used on each node of the VAXcluster. This improves consistency of RSX–11 device naming across the VAXcluster and results in less confusion and fewer errors. For the same reason, devices shared by many users should have $$n logical names defined in shared logical name tables, rather than in private logical name tables. Private $$n logical names should be defined in the user's LOGIN.COM (or LOGIN.CMD) file, rather than interactively, so that they are consistent from one interactive session to the next, and consistent between multiple interactive processes and batch processes.

To avoid confusion or overlap, DIGITAL recommends the following usage for $$n logical names:

- When many users on the system will share the $$n logical names, the $$n names should be defined in the SYSTEM logical name table. The unit numbers for these logical names should begin with unit number 0 ($$0) and ascend to higher unit numbers.
- When a user must define personal $$n names, either in the process or job logical name tables, the logical names should begin with unit number 347 ($$347) and descend to lower unit numbers.
- When applications are running on the same system or within the same UIC group, you should use job logical names rather than group or system names to minimize conflict. The $$n logical names created dynamically in the job logical name table should have unit numbers in the range of 350 to 357 ($$350 to $$357).
- When $$n logical names must be shared by tasks that are not in the same job, they should be created in the group logical name table and should have unit numbers in the range of 360 to 367 ($$360 to $$367).
- When $$n logical names must be shared by tasks that are not within the same UIC group, they must be created in the system logical name table and should have unit numbers in the range of 370 to 377 ($$370 to $$377).
- When an application must use group or system dynamically defined $$n logical names, the application should allow modification of the logical names it will use. This allows various applications to run on the same system. DIGITAL strongly recommends the use of only jobwide dynamic $$n logical names to minimize the conflict between applications.
- When a native mode image must define $$n names to allow sharing access of devices (for example, mailboxes with compatibility mode tasks), the application should allocate a $$n unit number in the range reserved for dynamic use. The application should deassign the $$n names when there is no further use for them.

2.7.2.4 VAX–11 RSX Internal Logical Names

If the VAX/VMS device name is not accurately represented by any assigned LUN, by the translation of either LB or SP, or by any $$n name, then the VAX–11 RSX AME attempts to match it against the translation of RSX–11 pseudodevices defined within the AME. These pseudodevices are, in order of searching, TI, SY, CO, CL, and OV. For example, if the VAX/VMS device name is SYS$DISK (your current default device and root directory, if any), then the equivalent RSX–11 device is SY0. If the VAX/VMS device name is mapped to TI, SY, CO, CL, or OV, then the RSX–11 device name (by which the device is known to any compatibility mode task) is as follows:

1. $I0 (SYS$INPUT)
2. SY0 (SYS$DISK)
3. $C0 (SYS$COMMAND)
4. $E0 (SYS$ERROR)
5. OV0 (the VAX/VMS device from which the image ran)

2.7.2.5 Direct Mapping of VAX/VMS Device Names

If none of the steps outlined in 2.7.2.1 through 2.7.2.4 result in an accurate representation of the VAX/VMS device name, then the system makes one final attempt to map the device name directly to an RSX–11 compatible device name, without the use of logical names. This mapping is performed by the following algorithm:

1. Subtract the value representing the ASCII character "A" (65 (10)) from the VAX/VMS device name controller letter.
2. Multiply the result of step 1 by 16 (10).
3. Add the result of step 2 to the VAX/VMS device unit number, and convert the result to octal.
4. Append the result to the VAX/VMS device name, deriving an RSX–11 compatible equivalent device name that can be restored to a VAX/VMS device name (using the algorithm described in 2.7.1).

VAX/VMS device names that include root directory names, that contain VAXcluster node names, that have unit numbers higher than 16 (10), or that have RSX–11 equivalent unit numbers (results of step 3) higher than 377 (8) cannot be represented by this algorithm. If attempted, an illegal device error is returned to the compatibility mode task.

Example:

To map VAX/VMS device LPB1 to RSX–11 device LP21:

```
Unit = (("B" - "A") * 16) + 1 = (1 * 16) + 1 = 17 (decimal)
Convert decimal 17 to octal = 21
```

Unless a VAX/VMS device name is mapped using this step, the RSX–11 device name returned by the Get LUN Information directive will have no relationship to the physical type of the device. Some RSX–11 tasks use the device name to determine information. These tasks will not function correctly under VAX–11 RSX unless the device name is mapped using this step. You should modify such tasks to use the VAX–11 RSX Get Device Information directive (GDVI$), instead of the device name.

2.8 Spooled Devices

Under VAX/VMS, spooling occurs as a result of cooperation among the I/O-related system services: Files–11 ACP, VAX RMS, and output symbionts. Spooling in RSX–11 requires interaction with the RSX–11 spooler. Use of VAX/VMS spooled devices is transparent to RSX–11 task images.

If a task image assigns a device that is spooled (for example, a line printer), the resulting assignment is actually to an intermediate device (for example, a disk). If the task image issues a Get LUN Information directive, VAX–11 RSX returns characteristics that are consistent with characteristics returned by RSX–11M–PLUS for a spooled device: the intermediate device characteristics and the output spooled bit.

If a task image uses RMS–11 or FCS to access a spooled device, the file is spooled when it is deaccessed.

Use of the Queue I/O Request directive to a VAX/VMS spooled device without preceding the request with an OPEN$ macro or appropriate ACP functions results in a privilege violation status return. Because the device to which the Queue I/O Request directive actually is directed is a file-structured device, the appropriate ACP functions (for example, access file) must occur before I/O to the device can be performed. Use of RMS–11 or FCS PUT$ requests ensures that the ACP functions occur.

VAX–11 RSX supports the FCS spooling macro PRINT$ and the services associated with it under RSX–11. Spooling in RSX–11 is accomplished by the PRT . . . task. When VAX–11 RSX detects a Send Data directive with PRT . . . as the target, it executes a Send Message to the Job Controller system service to spool the file.

2.9 File Specification Formats

RSX–11 and VAX/VMS file specifications are similar in that RSX–11 file specifications are a subset of VAX/VMS file specifications. Differences in the directory, file name, and file type fields result from differences in Files–11 Structure Level 1 and Files–11 Structure Level 2. VAX/VMS file specifications have the following format:

```
node::device:[directory]filename.type;version
```

where

node — is the node name, a 1- through 6-character alphanumeric string that must contain at least one alphabetic character. The DECnet node name identifies a location on the network. You can optionally include a 3- through 42-character access control string enclosed in quotation marks (") in the following format:

```
node"access-control-string"::
```

The node name is separated from the device name with two colons(::).

device

is the device name, specified in the following format:

```
clnod$ddcu:
```

clnod$dd is a mnemonic for the cluster device designation.

c is a controller designation.

u is the device unit.

The maximum length of the device name field is 15 characters.

The device name is separated from the directory name by a colon (:).

directory

is the directory name, specified in one of the following formats:

```
[g,m]
[name]
[name.name]
```

The [g,m] format consists of two numbers, separated by a comma, that represent the group number (g) and member number (m) that make up the UIC, as described in 2.4.1.

The other two directory formats allow the use of letters and numbers to form a directory name. A period present in a directory name indicates that the directory is a subdirectory of a higher-level directory. Each period in a directory name denotes a successively lower subdirectory level. Up to seven subdirectory levels are allowed. For Files–11 Structure Level 1 disks, each directory name can have as many as 9 characters. For Files–11 Structure Level 2 disks, each directory can have as many as 32 characters.

Suppose your directory contains files with extended file names or file types, or with file names or file types that have multinational characters in them. If you perform a RSX–11 find operation (for example, PIP /LI), the operation will list all the entries with standard RSX–11 file names and file types but will ignore the entries with extended file names or file types.

Angle brackets (<>) can be used instead of square brackets ([ ]) to delimit the directory name. For example, [SMITH] and <SMITH> are the same specification.

All the previously mentioned directory formats can be used for Files–11 Structure Level 1 and Files–11 Structure Level 2 disks. Note, however, that RSX–11 does not support hierarchical directories (subdirectories).

filename

is the file name, consisting of 0 through 9 alphanumeric characters on Files–11 Structure Level 1 disks. For Files–11 Structure Level 2 disks, a file name consists of 0 through 39 alphanumeric characters plus the following characters: underscore (_), hyphen (-), and dollar sign ($).

file type

is the file type, consisting of 0 through 3 alphanumeric characters on Files–11 Structure Level 1 disks. For Files–11 Structure Level 2 disks, a file type consists of 0 through 39 alphanumeric characters plus the following characters: underscore (_), hyphen (-), and dollar sign ($).

file version — is the file version number, ranging from 1 through 32,767 (10). You can use either a semicolon (;) or a period (.) to separate a file type from a file version number.

File names, types, and version numbers apply only to files on mass storage volumes. Directory names apply only to files on disk volumes. For record-oriented devices, the file specification requires only the device name field.

The maximum useful length of a directory name including subdirectories and the directory bracket characters is 252 (10) characters.

Unlike VAX/VMS, VAX–11 RSX does not accept file names or file types with multinational characters in them or extended file names or file types, that is, file names or file types with underscores (_), dollar signs ($), hyphens (-), and more than 9 alphanumeric characters in the file name, or more than 3 alphanumeric characters in the file type. No RSX–11 task can access a file with an extended file name or file type or with multinational characters in the file name or file type unless the task uses RMS–11. Refer to 3.16 for information on file names for RSX–11 tasks that use QIOs, FCS, or RMS–11.

You can use wildcard characters in the file name, file type, and version number fields of a file specification.

For repetitive operations, you can use logical names to specify all or part of a file specification.

2.9.1 Directory Names

Directory names are represented in the UIC ([g,m]) format or by an alphanumeric string of 1 through 39 characters in length. Using subdirectories, you can specify as many as seven directory levels after the main directory name. For example, a second-level directory has the following format:

```
[name.name]
```

UIC-format directory names cannot contain a group or member number greater than 377 (8). For example, [377,377] is a legal directory name, but [1000,1] is not because the group number 1000 (8) exceeds 377 (8). VAX/VMS does not support UIC- format directories with a group or member number that exceeds 377 (8).

The comma can be omitted from a UIC-format directory. For example, [123050] and [123,050] are equivalent on both VAX/VMS and RSX–11 systems. When subdirectories are used, the comma in a UIC-format directory or subdirectory name is omitted; leading zeros are required. An example of a directory name is [122020.MSG].

If you remove the comma from a UIC-format directory, you must provide leading zeros to expand the group amd member codes to three digits. For example, [10,1] would have leading zeros added to form the equivalent directory string of [010001].

Use the MCR command UFD as described in Section 4 to create directories on a Files–11 Structure Level 1 or Files–11 Structure Level 2 volume.

2.9.2 Version Numbers

RSX–11M Version 4.1 and RSX–11M–PLUS Version 2.1 display version numbers in octal. VAX/VMS displays version numbers in decimal. Howeve RSX–11 and VAX/VMS maintain version numbers in binary, so version numbers are compatible internally. To determine version numbers when you move files between systems, use PIP on either VAX/VMS or RSX–11 to list file name and version information as it appears in that system. The PIP supplied with VAX–11 RSX prints version numbers in decimal. You car also use the DIRECTORY command to display the contents of a directory o subdirectory.

Before you use negative version numbers, make sure you use them accordin to VAX/VMS conventions. Under RSX–11, version number -0 and 0 have identical meanings as the newest version of the file. Under VAX/VMS, however, only 0 means the newest version. Under VAX/VMS, version -0 means oldest version number, not -1 as under RSX–11. You can use -1, -2, and so forth under VAX/VMS to denote successively older versions. Under RSX–11, -1 means the oldest version and all lower negative values are illega In summary, under VAX/VMS you can use 0 as a means of specifying the newest version, -0 to specify the oldest version, -1 to specify the next to the newest version, and so forth.

Note: **Use of negative version numbers on Structure Level 1 disks is not supported under VAX/VMS.**

2.9.3 File Specification Defaults

When a field of a file specification is omitted, VAX–11 RSX supplies defaults Table 2–2 lists the defaults for each field. Table 2–3 lists the default file type that you are likely to encounter while using VAX–11 RSX. For a complete lis of VAX/VMS file types, see the *VAX/VMS DCL Dictionary.*

Table 2–2 File Specification Defaults

Field	Default
node	The local system is your default.
device	The disk is established as your default by (1) your entry in the user authorization file or (2) execution of the SET DEFAULT command.
directory	The directory is established as your default by (1) your entry in the UAF or (2) execution of the SET DEFAULT Command or the SET /UIC command.
file name	No defaults are applied to the first file name in an input file specification. Most commands apply default output file names based on the file name of an input file.
file type	This file type is the default for the particular command or component being used (see Table 2–3).

Table 2–2 (Cont.) File Specification Defaults

Field	Default
file version	The rules for establishing default file version numbers are the same as those used in RSX–11. For input files, the system assumes the most recent version, that is, the highest version number. For output files, the system increases the version number by 1 for existing files and supplies a version number of 1 for new files.

Table 2–3 Default File Types

File Type	File Contents
CBL	Source input to the PDP–11 COBOL-74/VAX Compiler
CLB	Indirect Command Processor command library
CLD	Command language description file
CMD	RSX–11 indirect command file to be executed with the @ (Execute Command Procedure) command under VAX–11 RSX MCR or by various VAX–11 RSX utilities
COM	VAX/VMS command procedure file that is executed with the @ (Execute Command Procedure) or submitted for batch execution with the VAX/VMS DCL command SUBMIT
COR	Source input for the VAX–11 CORAL 66 Compiler
DAT	Input or output data file
DIF	Output listing created by the DIFFERENCES command
DIR	Directory file
DIS	Distribution list for the MAIL command
DMP	Output from the DUMP command
EDT	Initialization command input file for the EDT editor
EXE	Image created by the VAX/VMS Linker or the VAX–11 RSX version of the RSX–11 Task Builder
FDL	File definition language (FDL) file that defines file characteristics for a VAX/VMS file
FTN	Source language input for the PDP–11 FORTRAN IV, or the PDP–11 FORTRAN-77 Compiler
HLB	Help text library file
HLP	Input source file for help libraries
INI	Initialization file
JNL	Journal file output from the VAX–11 PATCH Utility
JOU	Journal file/audit trail from EDT
LOG	Batch job output file
LIS	Default input file type for the PRINT and TYPE commands
LST	Listing file produced by a PDP–11 compiler or assembler

Table 2–3 (Cont.) Default File Types

File Type	File Contents
MAC	Source input to the PDP–11 MACRO–11 Assembler
MAI	Mail message file
MAP	Memory allocation (map) listing produced by the VAX/VMS Linker or the RSX–11 Task Builder
MEM	Output file for DIGITAL Standard Runoff (DSR) text formatter
MLB	VAX/VMS or RSX–11 macro library
MSG	Source file that specifies the text of messages
OBJ	Object module produced by a PDP–11 or VAX–11 assembler or compiler
ODL	Overlay description input to RSX–11 Task Builder
OLB	VAX/VMS or RSX–11 object module library
RNO	Source input for DIGITAL Standard Runoff (DSR) text formatter
SML	File Control Services (FCS) and system directives macro library
STB	Symbol table file created by the VAX/VMS Linker or the RSX–11 Task Builder
TMP	Temporary file
TXT	Input file for text libraries or output file for the MAIL command
UPD	Update file of changes for a VAX–11 MACRO source program; also input to the SUMSLP editor

2.10 Preparing to Run an Image

The basic steps required to prepare an image for execution are the same for VAX/VMS and RSX–11 images:

- Allocate and mount any physical devices that the image requires
- Equate VAX/VMS physical device names to any logical device names or RSX–11 physical device names that the image uses

For an image to be executed, it must reside on a Files–11 disk volume. The volume must be mounted and you must have the right to gain access to it (that is, you must be allowed execute (E) access as described in 2.5.5).

2.10.1 Allocating Devices

VAX–11 RSX has essentially the same device allocation as RSX–11. That is, you can use the command ALLOCATE to reserve a device. Once allocated, the device is not available to other users; it remains allocated until you deallocate it or log out of the system. Under RSX–11, allocated devices are reserved for any tasks that you initiate. Under VAX–11 RSX, on the other hand, allocated devices are reserved for any task image that runs in your current process.

You can allocate and mount devices in an indirect command file. The devices will remain mounted after the command file executes. However, if you allocate the device, but do not mount it in the command file, the device will not remain allocated after the Indirect Command Processor exits.

The VAX–11 RSX MCR command ALLOCATE accepts as a parameter either a logical device name or a VAX/VMS physical device name. If you specify a physical device name, it can be either explicit or generic. The ALLOCATE command also accepts a logical name assigned to the device allocated as an optional second parameter.

A **generic device name** explicitly specifies the device type and optionally specifies the controller or unit. A generic device name allows the system to select the actual device unit to be allocated from available devices of a specific type. For example, the following command requests VAX/VMS to allocate any available TE16 tape drive:

```
> ALL MT:
_MTB2: allocated
```

The system response indicates that Drive 2 on Controller B has been allocated. When the device type and controller are specified and the unit number is not, the ALLOCATE command looks for an available unit on a specific controller. Similarly, when the device type and unit number are specified, the ALLOCATE command looks for a controller that has the specific device unit available.

Unlike RSX–11, when the controller and unit number are omitted, VAX/VMS does not default to Unit 0 of Controller A. This difference may affect indirect command files that allocate devices for Unit 0. For more information, refer to Section 5.

The following command requests VAX/VMS to allocate any available RK07 device:

```
> ALLOCATE/GENERIC RK07
_DMA0: allocated
```

For more information on the previous command, refer to Section 4.

2.10.2 Mounting Volumes

Volumes are mounted in essentially the same way under VAX/VMS as they are under RSX–11. Both systems support a MOUNT command, and both systems define similar functions requested by specifying MOUNT command keywords. The actual keyword names, however, differ between systems. The VAX/VMS command MOUNT also has additional options that function as follows:

- Define a logical name for the device on which the volume is mounted.
- Specify volume owner and volume protection.
- Specify the degree of sharing allowed on the volume (for example, specify a private volume or one that is available to the group or world).
- Specify whether operator or user intervention is allowed.

All the MCR MOUNT command options are described in Section 4 in the MOUNT command description.

The main purpose of the MOUNT command is to gain access to the volu However, under VAX–11 RSX, you can use certain disk volumes withou issuing a MOUNT command for them. Any volume that another user (m likely, the system manager or operator) has mounted specifying the /SYS keyword is available to anyone logged in on the system. Any volume th another user has mounted specifying the /GROUP keyword is available t users with the same UIC group number as the user who issued the MOU command.

The MOUNT command accepts either a logical device name or a physica device name as a parameter. If you specify a physical device name, you do so either explicitly or by using defaults. The default for the controller and the default for the unit number is 0.

2.10.3 Assigning Logical Names

Under VAX–11 RSX, you can use logical names in RSX–11 task images, VAX–11 native mode images, and MCR commands. Using the ASN or DEFINE command, you can equate a logical name with a VAX/VMS phys device name, as in the following example:

```
> ASN DBB2:=IN0:
```

You also can equate a logical name with all or a portion of a file specificat as in the following example:

```
> ASN DBB2:[MYDIR]FILEA.DAT;2=INPUT
```

Unlike RSX–11 logical names, VAX/VMS logical names are not restricted two letters and a unit number. VAX/VMS logical names can consist of as many as 255 (10) alphanumeric characters. Note that the ASN command also delete a logical name. The following sections describe the use of logic names for RSX–11 task images and VAX–11 native mode images.

2.10.3.1 Using Logical Names for Task Images

RSX–11 task images issue Assign LUN directives to assign a LUN to an RSX–11 physical device unit or a logical name. Similarly, the TKB option ASG can be used to assign a LUN to a physical or logical device at build time.

When either method is used to assign a LUN to an RSX–11 physical devic unit, you can use the ASN command to define the RSX–11 physical device name as the logical name for a VAX/VMS device. Then, when VAX–11 R finds the RSX–11 name in the program, it translates the name to the assoc ated VAX/VMS physical device name. The following example illustrates t use of the ASN command.

```
> ASN DBA1:=DK2:
```

When the Assign LUN directive is issued or when the image activated has TKB option to assign a LUN to a logical device name, you can use the AS command to equate the logical name with a VAX/VMS physical device nar For example:

```
> ASN DBB0:=XY:
```

The logical name assignment can be made by entering the appropriate AS command before entering the RUN command to invoke the RSX–11 task image. Alternatively, an indirect command file can execute both the ASN a RUN commands, thus reducing the possibility of operator error at run time.

If you do not assign logical names, VAX–11 RSX attempts to map the RSX–11 device names used by the image to VAX/VMS device names. (See 2.7 for more information on device name mapping.)

2.10.3.2 Using Logical Names Instead of the RSX–11 MCR REA Command

Under RSX–11, a privileged user can assign a physical or logical device name to a LUN using the REA (reassign) command.

Under VAX–11 RSX, you must assign LUNs at either the program or task-build level; the VAX–11 RSX MCR command interpreter does not support the REA command. However, the logical name facility provides the flexibility needed to associate the RSX–11 device name used in an Assign LUN directive or ASG option with a VAX/VMS physical device, as just described.

2.10.3.3 Using Logical Names Instead of the RSX–11 MCR RED Command

Under RSX–11, a privileged user can redirect I/O operations from one device to another during task execution. VAX–11 RSX MCR does not support the RED (redirect) command, so you must assign logical names before running the task image.

2.10.3.4 Using Logical Names for VAX–11 Native Images

The VAX/VMS logical name capability provides more flexibility than the RSX–11 capability. Native images can use a logical name that is equated with all or part of a file specification starting with the leftmost item of the file specification. Similarly, you can use logical names that are equated to file specifications at the command level. The following example shows the use of logical names:

```
> ASN DBB1:[JONES]SNOW.DAT;4=INPUT
> RUN ANLZE
        .
        .
        .
```

The ASN command equates the logical name INPUT with the full file specification DBB1:[JONES]SNOW.DAT;4. When the task image ANLZE refers to INPUT, VAX–11 RSX automatically translates it to the associated file specification.

2.11 Running Images

VAX–11 RSX MCR lets you run VAX/VMS native images and RSX–11 task images. The basic format of the command to request task execution is as follows:

```
RUN file-spec
```

The parameter file-spec represents a standard VAX/VMS file specification or a logical name that translates to one. If the file type is not supplied, it defaults to EXE. To run RSX–11 task images that have a file type of TSK, you must either supply the file type or rename the image files. EXE is the default file type for image files produced by the VAX/VMS Linker and the VAX–11 RSX Task Builder.

The RUN command provides keywords that control whether the requested image is run in your process, a subprocess, or a detached process. Additional keywords allow you to control the process's priority, scheduling, resource quotas, and privileges. A detailed description of the RUN command and all its options can be found in Section 4.

The following are examples of the RUN command:

```
> RUN INVENTORY
```

This command runs the image in the highest version of the file INVENTORY.EXE located on the default disk device and directory.

```
> ASN DBB2:[RSX]IMAGE.TSK;8=MYTASK
> RUN MYTASK
```

This sequence assigns MYTASK as the logical name for the task image. The RUN command translates MYTASK to DBB2:[RSX]IMAGE.TSK;8 and runs it.

Instead of using the RUN command, you can run a VAX–11 RSX task (for example, PIP) in your process by entering the file specification of the task file in response to the MCR prompt (>) or by entering a logical name that translates to a file specification.

2.12 Error Message Format

Under VAX–11 RSX, when you enter an MCR command incorrectly, MCR issues a message in the standard VAX/VMS format. Such messages do not look like RSX–11 MCR error messages. Instead, they have the following general format:

where

```
%XXX-L-CODE, text
```

where

XXX is a mnemonic for the operating system program issuing the message.

L is a severity level indicator: S for success, I for information, W for warning, E for error, and F for fatal.

CODE is a shorthand code for the message text that follows the code.

For example:

```
%MCR-W-MAXPARM, too many parameters - reenter with fewer characters
```

Because VAX–11 RSX messages are descriptive, you can usually learn the corrective action from them. Section 9 lists all of the VAX–11 RSX system error messages and describes corrective measures for them.

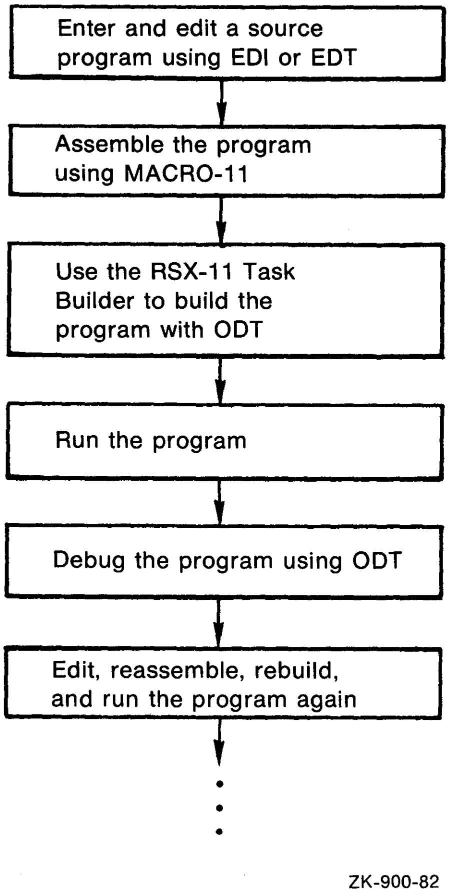
RSX–11 components (for example, PIP and TKB) issue the same messages under VAX–11 RSX that they issue under RSX–11. When using VAX–11 RSX MCR to run RSX–11 components, you should consult the appropriate RSX–11 documentation for error message information. For example, if you encounter an error message while using PIP, see the *RSX–11M/M–PLUS Utilities Manual* for an explanation of the message.

3 The VAX–11 RSX Programming Environment

3.1 Developing Programs Under VAX–11 RSX MCR

VAX–11 RSX users can develop RSX–11 task images with MCR as the command interpreter. Figure 3–1 illustrates the general flow of VAX–11 RSX (compatibility mode) program development using MCR.

Figure 3–1 RSX–11 Program Development Cycles Using VAX–11 RSX MCR


Enter and edit a source program using EDI or EDT
Assemble the program using MACRO-11
Use the RSX-11 Task Builder to build the program with ODT
Run the program
Debug the program using ODT
Edit, reassemble, rebuild, and run the program again
ZK-900-82

3.1.1 Using RSX–11 Components

You can use any of the RSX–11 components listed in Section 1 by issuing a request to VAX–11 RSX MCR. To request any of these components, either enter the component name and press RETURN or enter the component name followed by a valid command line for that component and press RETURN. Subsequent operation of the component is identical to that under RSX–11, as documented in such manuals as the *RSX–11M/M–PLUS Task Builder Manual,* and the *RSX–11M/M–PLUS Utilities Manual.*

In addition, you can build the On-Line Debugging Tool (ODT) into any RSX–11 task image that requires debugging. ODT operates just as it does under RSX–11. You also can interrupt task execution and type the DEBUG command to pass control to ODT if the task was built with ODT.

All the available RSX–11 components accept a subset of the standard VAX/VMS file specification format except the File Transfer Program (see 3.1.1.1 for more information). The RSX–11 components ignore the use of VAX/VMS file names that exceed nine characters or file types that exceed three characters. In addition, file names and file types that include dollar sign ($) or underscore (_) characters are not recognized by RSX–11 components. See Section 2 for more information on file specifications. Furthermore, VAX–11 RSX does not support the PIP /UF switch; you must use either the command UFD or the command CREATE/DIRECTORY (see Section 4).

3.1.1.1 Invoking RSX–11 Components for Program Development

To invoke VAX–11 RSX components for program development from VAX–11 RSX MCR, you must first create a source file using a text editor. Exiting from the editor causes MCR to prompt you for additional input.

For example, the following command requests the MACRO–11 Assembler (MACRO–11) to read the indirect command file MACINP.CMD for command input. After MACRO–11 processes your file and creates an object module, MCR prompts again.

```
> MAC @MACINP
>
```

As another example, typing TKB followed by a command line results in the task building of the object file GRAPES.OBJ on the default disk and directory to produce the image file APPLES.EXE, which is also stored on the default disk and directory. GRAPES.OBJ must have been produced by an RSX–11 compiler or assembler. When task building completes, MCR prompts again.

```
> TKB APPLES=GRAPES
>
```

The following sequence of commands invokes the RSX–11 Task Builder (TKB), supplies it with the names for the input and output files, and requests TKB to prompt for options. The two slashes cause TKB to exit, after which MCR prompts.

```
> TKB
TKB>AVERAGE,AVERAGE=AVERAGE
TKB>/
ENTER OPTIONS:
TKB> .
     .
     .
TKB>//
>
```

The following command invokes the Peripheral Interchange Program (PIP) to delete all listing (LST) files from directory [ADAMS] on DBB2. MCR prompts after the files are deleted.

```
> PIP DBB2:[ADAMS]*.LST;*/DE
>
```

In the following sequence, typing PIP in response to the MCR prompt causes PIP to prompt. PIP reprompts after each command until you press CTRL/Y to pass control to MCR. Note that PIP displays listed file version numbers in decimal instead of octal.

```
> PIP
PIP>SPICE.*/PU
PIP> .
     .
     .
PIP> CTRL/Y Interrupt
>
```

3.1.1.2 Using the File Transfer Program

The File Transfer Program (FLX) interprets only RSX–11 file specifications, that is, file specifications with the following format:

```
ddu:[g,m]filename.type;version
```

where the device name, directory, file name, and file type are in standard RSX–11 file specification format.

FLX cannot interpret a controller designation in the device name or a directory name (other than the [g,m] form) in the file specification, and it does not accept a logical name for a file specification. If you omit the directory from a file specification for a disk file, VAX/VMS uses your default directory. It can be in either the [g,m] form or a directory or a subdirectory name.

Because FLX does not accept VAX/VMS device names, your use of FLX depends on the device mapping that VAX–11 RSX performs. If you specify a device name that does not map to a physical device on VAX/VMS, FLX issues an error message indicating that it could not find the device. Device mapping is described in Section 2.

Before using FLX with a magnetic tape, you must allocate the tape drive. Unless you are using an ANSI-labeled magnetic tape, you must mount the tape drive using the /FOREIGN keyword for the MOUNT command after you allocate it. DOS–11 and RT–11 tapes are not ANSI-labeled and thus require the MOUNT/FOREIGN command. Also, note that ANSI (17-character) file names on magnetic tape are supported under VAX–11 RSX.

The following are examples of FLX command lines.

Examples

1

```
> FLX DB0:[120,30]=MT1:SYS1.MAC/DO
```

VAX–11 RSX maps the device name DB0 to the VAX/VMS physical device DBA0. It maps device name MT1 to device MTA1. File SYS1.MAC is transferred to directory [120,30] on DBA0.

2

```
> FLX
FLX>DR3:=DM0:[133,10]MYFIL.MAC/RT
```

VAX–11 RSX maps device name DR3 to the VAX/VMS physical device DRA3 and DM0 to DMA0. The user's default directory on DRA3 contains the output file.

3.1.2 Using VAX/VMS Components

VAX–11 RSX users who are not familiar with VAX/VMS utilities should be aware of certain differences between RSX–11 and VAX/VMS utilities. Under RSX–11, some optional parameters are specified by using "switches." Certain switches allow a minus sign (-) to precede the switch name, indicating that the switch is to be turned off, or negated, usually to override a default. The switches and any required parameters are usually specified on the command line following the command name.

Under VAX/VMS, these optional parameters are called "qualifiers," not switches. To negate a qualifier (switch) while using a VAX/VMS utility or command, you should precede the qualifier with the word NO instead of a minus sign (-). With required parameters, most VAX/VMS utilities will prompt for the required parameter if it was omitted from the entered command line. If you are not sure of the order in which required parameters must be entered, you can omit one or more required parameters and prompts will appear. Like an RSX–11 switch, each VAX/VMS utility qualifier must be preceded by a slash (/).

For information about the qualifiers and the operation of VAX/VMS utilities, see the appropriate VAX/VMS documentation.

3.2 Building RSX–11 Task Images for Execution

VAX–11 RSX can serve as the host for the following:

1 Modifying existing RSX–11 task images so that they execute under VAX–11 RSX in compatibility mode

2 Developing new task images in compatibility mode to run under RSX–11 or under VAX–11 RSX

RSX–11 program development utilities can be used to develop or modify programs; however, programs that will execute only under RSX–11 should be debugged under RSX–11.

3.2.1 Task Builder Switches

You can use most of the switches defined for the RSX–11 Task Builder while you are building an image to execute under VAX–11 RSX. One RSX–11 switch that you cannot use is /-MM (unmapped system); that is, the task image must be built to execute on a mapped system.

Table 3–1 lists the switches known to be effective under VAX–11 RSX. With the exception of /-MM, unlisted switches have no effect under VAX–11 RSX and do not cause an error when the task image is executing under VAX–11 RSX. As a general rule, switches that are intended to affect how RSX–11 treats the task image during execution are ignored. Switches requiring privilege are also ignored.

Defaults and the use of a minus sign (-) to negate switches are identical to those in RSX–11.

Table 3–1 Valid RSX–11 Task Builder Switches

Switch	Function
/CC	Input file consists of concatenated object modules.
/DA	Task image contains a debugging aid.
/DL	Specified library is a replacement for the system default object module library.
/FP	Task image uses the floating-point processor (FPP only).
/FU	All co-tree overlay segments are searched for matching definition or reference when modules from the default object module library are being processed.
/HD	Task image includes a header.
/LB	Input file is a library file.
/MA	Memory allocation output includes information from the file.
/MM	System has memory management; for a task image to run under VAX–11 RSX, the /MM switch must be present either explicitly or by default.
/MP	Input file contains an overlay description.
/MU	Task image is separated into shareable and nonshareable portions.
/PI	Task image is position-independent.
/PM	Postmortem dump is requested.
/SH	Short memory allocation file is requested.
/SP	Spool map output.
/SQ	Task image program sections are allocated sequentially.
/SS	Selective search is to be made for global symbols.
/TR	Task image is to be traced.
/WI	Memory allocation file is printed at a width of 132 characters.
/XT:n	TKB exits after n diagnostics.

Use of overlays can sometimes be avoided to take advantage of VAX/VMS virtual address space. VAX–11 RSX provides 64K-bytes of virtual address space for RSX–11 task images.

3.2.2 Task Builder Options

You can specify any of the RSX–11 Task Builder options for RSX–11 task images run under VAX–11 RSX. The options that specify User Identification Code (UIC), priority, and partition have no effect; UIC and priority are associated with the user running the task image under VAX/VMS, and partitions do not exist. Two options, TASK and ASG, may be required for RSX–11 task images to run successfully under VAX–11 RSX, as described in the following:

TASK: RSX–11 task images that are the target of directive requests should have a task name assigned using the TASK option. If no task name is assigned, the task is assigned a task name based on the process name in effect when the task is invoked.

ASG: RSX–11 task images that do not associate LUNs with either a physical device name or a logical device name within the program must be built using the ASG option to make the required association. MCR does not support the REA (Reassign) command.

3.2.3 Noncontiguous Task Image Files

TKB issues an informational message if it produces a noncontiguous task image file. VAX–11 RSX, however, allows you to run task images contained in noncontiguous files.

3.3 Using the ELP$ Directive for Parsing

The Extended Logical Parse directive (ELP$) allows an RSX–11 task image to use VAX/VMS file specifications that are not fully qualified (since they use logical names). Use of the ELP$ directive logically replaces the operation of the RSX–11 versions of the File Control Services (FCS) .PARSE, .PRSDR, and .PRSDV routines. The VAX–11 RSX versions of these routines use ELP$ when running under VAX–11 RSX.

An RSX–11 task image requests the parsing of a file specification by issuing an ELP$ directive. The ELP$ directive supplies the addresses of a file name block and data structures that contain default information. VAX–11 RSX uses the information supplied by the image and the information contained in the VAX–11 RSX and the VAX/VMS logical name tables to build the primary and default strings that VAX–11 RMS requires to perform the actual parsing. Then VAX–11 RMS returns the expanded name to VAX–11 RSX. VAX–11 RSX, in turn, uses the expanded name to fill in the appropriate RSX–11 data structures. The result is that the task image receives the information in the normal RSX–11 formats.

The ELP$ directive does support ANSI "a" file specifications, as defined in the appropriate RSX–11 documentation.

VAX–11 RSX must return the expanded name string in appropriate RSX–11 data structures. Therefore, the file specification for RSX–11 tasks must be compatible with the standard RSX–11 file specification format (see Section 2). This restriction does not apply to RSX–11 task images that use RMS–11 Version 2.0 (see 3.16.2).

The task image can request four different types of parsing:

1. Parsing of the full file specification (normal mode)
2. Parsing of the device name only (device-only mode)
3. Parsing of the file name using the default file name block as the major source of input (dfnb mode)
4. Parsing of the full file specification for RMS Version 1.8

3.3.1 Normal Mode Parsing

When the mode parameter of the ELP$ directive is equal to 0, VAX–11 RMS parses the full file specification. VAX–11 RSX builds the primary string required as input to VAX–11 RMS by concatenating fields of the dataset descriptor, as follows:

- Device
- Directory
- Filename.type;version

It builds the default string from fields of the default file name block and from the default directory descriptor, as follows:

- It builds the device from the LUN or default file name block.
- It builds the default directory from the task image's default directory descriptor.
- It builds the filename.type;version from the default file name block.

VAX–11 RSX returns a filled-in file name block and directory string descriptor to the RSX–11 task image. The directory string is returned at the address specified in the descriptor.

3.3.2 Device-Only Parsing

When the mode parameter is equal to 1, VAX–11 RMS parses only the device and directory portion of the file specification. It uses the same sources for the primary and default strings as it does for a normal parsing operation.

3.3.3 Default Filename Block Parsing

When the mode parameter is equal to 2, VAX–11 RSX uses the Radix–50 file name in the default file name block to build the ASCII file name for the primary string.

For the default string, VAX–11 RSX takes the device name from the default file name block. It takes the directory name from the default directory descriptor, and the file name, type, and version from the default file name block.

The Directive Status Word (DSW) return codes for default file name block parsing are the same as for normal mode parsing.

3.3.4 RMS–11 Parsing

When the mode parameter is equal to 3, VAX–11 RSX parses the file specification using the same method as RMS–11 Version 1.8. The format for the DPB is slightly different from that used for modes 0, 1, and 2, as described in 3.3.5.

This does not apply to RMS–11 Version 2.0, but is noted for compatibility with previous versions.

3.3.5 Directive Call and DPB Formats

The ELP$ directive does not have a macro defined for it. The ELP$ name is used for documentation and reference purposes because it is familiar to those who use other RSX–11 directive macros.

The ELP$ directive is called using DIR$, as follows:

```
DIR$   #pardpb
```

The DPB has the following format for modes 0, 1, and 2:

```
pardpb:    .BYTE  145.,8.
           .WORD  mode
           .WORD  lun
           .WORD  dspt
           .WORD  dfnb
           .WORD  dfdd
           .WORD  fnb
           .WORD  rtdd
```

mode = 0 for normal, 1 for device-only, or 2 for default file name block

lun = logical unit number

dspt = address of the data set descriptor

dfnb = address of the default name block

dfdd = address of the descriptor for the default directory string; see the note that follows

fnb = address of the file name block

rtdd = address of the descriptor for the returned directory string; see the note that follows

The DPB has the following format for mode 3:

```
pardpb:     .BYTE   145.,8.
            .WORD   mode
            .WORD   lun
            .WORD   pript
            .WORD   did
            .WORD   0 (not used)
            .WORD   fnb
            .WORD   expnam
```

The definitions of mode, lun, and fnb are the same as those for the DPB format previously listed.

mode = 3 for RMS–11 Version 1.8

pript = address of the primary input descriptor

did = address of the default input descriptor

expnam = address of the descriptor for the block in which to return the expanded name

The DSW return codes are as follows:

```
IS.SUC---Success
IE.BAD---Invalid mode, missing or bad parameter (default error)
IE.NSF---Directory not found (RMS$_DNF)
IE.BDI---Bad directory syntax (RMS$_DIR)
IE.BNM---Bad file name or non-RSX file name. (RMS$_(SYN,FNM,LNE,TYP,VER))
IE.DNR---Device not ready (RMS$_DNR)
IE.DUN---Device not available (RMS$_CHN)
IE.NSF---File not found (RMS$_FNF)
IE.BDV---Bad device specification (RMS$_DEV)
IE.SDP---Invalid DPB length
```

Note: All descriptor input parameters must be a two-word block with the following format:

```
.WORD    size
.WORD    address
```

3.4 Device Assignment

VAX–11 RSX assigns devices for RSX–11 task images as part of initialization of the task image when the image is loaded. It also assigns devices during execution of the task image as a result of an Assign LUN directive. For additional information, refer to 2.7.

3.5 Installing RSX–11 Task Images

Under VAX–11 RSX, task images that reside in SYS$SYSTEM do not have to be installed before execution. You run task images under VAX–11 RSX just as a nonprivileged user would run them under RSX–11, that is, by supplying the file specification of the task image as a parameter to the RUN command. VAX–11 RSX does not support the RSX–11 INSTALL command.

You can use logical names to create an effect similar to that of the INSTALL command. If you assign a logical name to the file specification of the task image file, the logical name serves as a task name. If you type the logical name in response to the MCR prompt (>), VAX–11 RSX translates logical name and runs the associated task image.

For example, the following command establishes REP as the logical name for the task image file DB4:[IMAGES]ALPHA.TSK contained on DBA4.

```
> ASN DB4:[IMAGES]ALPHA.TSK=REP/GBL
```

Because the logical name REP is placed in the system logical name table, any user can type REP to run the task image, as follows:

```
> REP additional-data
```

The task image can obtain the additional data by issuing a Get MCR Command Line directive, just as it does in RSX–11.

Note that for a process to place names in the system logical name table, the user must have SYSNAM privilege.

3.6 Process Hibernation

A hibernating process is known to the system, but is inactive. Suspending or stopping a task image causes the process executing the task image to hibernate.

A suspended task image can be reactivated by one of the following:

- An Asynchronous System Trap (AST)
- A Request Task directive
- A Resume Task directive
- A Run Task directive

A stopped task image can be reactivated only by one of the following:

- An Unstop Task directive located in an AST service routine
- An Unstop Task directive located in another process
- A specified event flag being set

In MCR, RUN command options allow a subprocess or detached process that is initially hibernating (rather than active) to be created. The VAX–11 RSX Applications Migration Executive (AME) is not started until the process reactivates. The loading of the compatibility mode task, the initial assigning of devices, and the library/common area mapping are not done until the process reactivates.

3.7 Task Image Termination

A task image running under VAX–11 RSX can terminate normally or abnormally. Normal termination occurs when the task image terminates itself by executing the appropriate directive. Abnormal termination occurs when the system or another process forces the task image to exit.

3.7.1 Normal Task Image Termination

When an RSX–11 task image terminates normally, VAX–11 RSX performs the same type of image cleanup operations as VAX/VMS does for a native image. If an RSX–11 task image issues a Task Exit directive, VAX–11 RSX executes an Exit system service and returns the termination status of SS$_NORMAL.

RSX–11 task images also can issue an Exit With Status directive to specify the appropriate status. For VAX/VMS native images and RSX–11 task images, the termination status is available to the command interpreter.

Both VAX/VMS DCL and VAX–11 RSX MCR use the termination status when processing indirect command files. DCL uses the termination status with the ON command for error handling. MCR uses the termination status with .ONERR for error handling.

3.7.2 Abnormal Termination

When a VAX/VMS native image incurs a potentially fatal error condition, either of the following can occur:

- The native image can handle the condition.
- The native image cannot handle the condition, so VAX/VMS forces the image to terminate.

VAX/VMS native images can react to fatal errors using the VAX/VMS condition-handling mechanism. Through that mechanism, an image can provide one or more condition-handling routines that are to be executed to handle an exception (error) condition. The condition-handling mechanism provides a function that is comparable to, but more flexible than, the RSX–11 Synchronous System Trap (SST) mechanism.

If a native image incurs and handles an exception, the image can continue execution or exit normally, as described in 3.8.1. If the native image does not handle the exception, the system terminates the image by issuing an Exit system service. The Exit system service initiates image-related cleanup operations and saves the termination status. This status is available to the command interpreter or the next native image to execute in the process.

Abnormal termination of an RSX–11 task image can occur as a result of any of the following:

- Violation of the hardware conventions for task images running in compatibility mode
- Issuance of an instruction, other than EMT 377, that causes a trap
- Use of an illegal JMP or JSR instruction format
- Occurrence of an odd address error
- Request for an abort from another process
- Attempt to exceed virtual memory usage limits

An RSX–11 task image can supply an SST service routine to handle some of the errors listed. If the address of an SST service routine for an error is supplied in the SST vector table and that error occurs, VAX–11 RSX continues allowing the task image to execute in the SST routine. This routine determines whether the task image is to exit or continue. If no SST address is supplied, VAX–11 RSX terminates the task image.

If the error is one that cannot be handled by an SST service routine or if no valid SST routine address is supplied, VAX–11 RSX issues a termination message on the device(s) assigned to SYS$ERROR and SYS$OUTPUT. VAX–11 RSX causes the task image to exit with a termination status that is available to the command interpreter.

Table 3–2 lists the reasons for task image termination and indicates which errors can be handled by an SST service routine. The status codes in parentheses following the termination messages are defined by the $RSXDEF macro.

Table 3–2 Reasons for RSX–11 Task Image Termination

Status Code	Message and Explanation
RSX$_IOT	IOT EXECUTION The task image executed an IOT instruction.
RSX$_BREAK	BPT EXECUTION The task image executed a BPT instruction.
RSX$_TBIT	T-BIT EXECUTION The task image executed an instruction requiring a T-bit trap.
RSX$_TRAP	TRAP EXECUTION The task image executed a trap instruction.
RSX$_NONRSXEMT	NON-RSX EMT EXECUTION The task image executed an invalid EMT instruction.

Table 3–2 (Cont.) Reasons for RSX–11 Task Image Terminatio

Status Code	Message and Explanation
RSX$_ILLINST	ILLEGAL INSTRUCTION The task image executed a JMP or JSR instruction with a register as the destination.[1]
RSX$_ACCVIO	MEMORY PROTECTION VIOLATION The task image addressed a location outside its virtual address space.[1]
RSX$_ODDADDR	ODD ADDRESS ERROR The task image addressed a word at an odd address (nonword boundary).[1]
RSX$_RESERVED	RESERVED INSTRUCTION The task image executed an instruction that is not allowed in compatibility mode (HALT, MARK, RESET, SPL, or WAIT).[1]
RSX$_BADSTACK	BAD STACK The stack pointer contained an address outside the task image's virtual address space.[2]
RSX$_INSFDYNMEM	NO DYNAMIC SPACE A requested service needed more dynamic memory than the process is allowed.[2]

[1] The returned PC is the address of the invalid instruction, not the address following it.
[2] This error cannot be trapped to the SST vector table.

3.8 PDP–11 Floating-Point Instructions Emulation

VAX–11 RSX supplements the compatibility mode instruction set with software emulation of the FPP (Floating Point Processor) instructions. However, FIS (Floating Instruction Set) instructions and EAE (Extended Arithmetic Element) instructions are not emulated.

The time required for emulation of an ADDF (register to register) or ADDF (memory to register) is approximately 25 times that required on a PDP–11/70. This timing is typical of most emulated FPP instructions.

Results produced during emulation are the same as those produced by PDP–11 processors with the following exceptions:

- The result of a MOD instruction is more accurate under emulation.
- If a floating point overflow or division by zero occurs during the emulation of a floating point instruction, the emulator generates a reserved operand in the destination of that instruction. The reserved operand is known as the **undefined variable.** The undefined variable has the bit pattern of the sign bit set with all other bits equal to zero. The floating point N and V condition code bits are set and bits Z and C are cleared.

- If a floating point underflow occurs during the emulation of a floating point instruction, the emulator generates a zero value in the destination of that instruction. The floating point N, V, and C condition code bits are cleared and bit Z is set.
- A floating point OP code exception interrupt cannot be disabled with the LDFPS instruction as it can be on the PDP–11/70. If a task specifies a floating point exception AST with the SFPA$ directive and a floating point OP error code is detected by the emulator, the task will always get the AST with the appropriate exception code placed on the stack. This occurs whether exception interrupts have been disabled with the LDFPS instruction.
- Floating point maintenance mode is not supported by the emulator. Setting the floating point maintenance mode bit in the emulated floating point status register with the LDFPS instruction has no effect on emulator execution.

The FPP instruction set is detailed in the *PDP–11 Architecture Handbook.*

9 Event Flag Emulation

An RSX–11 task can have up to three event flag clusters of 32 bits each. VAX–11 RSX emulation of these event flags is completely transparent to RSX–11 task images. The event flag clusters are the following:

- Local event flags, numbered 1 through 32
- Common event flags, numbered 33 through 64
- Group global event flags, numbered 65 through 96

Although event flag emulation is transparent to RSX–11 task images, the handling of these flags is important to interactions between native VAX/VMS processes and RSX–11 task images.

Event flag conversion for a VAX/VMS process is as follows:

Cluster	RSX–11	VAX/VMS
Local	1 through 32	32 through 63
Common	33 through 64	64 through 95
Group global	65 through 96	96 through 127

.9.1 Local Event Flag Emulation

Local event flags (flags 1 through 32) are visible only to the local task image. Every task image is associated with its local event flag cluster. A task image can read and set its local event flags. A VAX/VMS native image cannot associate with the local event flag cluster for an RSX–11 task image.

3.9.2 Common Event Flag Emulation

Common event flags (flags 33 through 64) are visible to all processes associated with the flag cluster. All RSX–11 task images are associated with a common event flag cluster. Any process can read and set flags in its associated common event flag cluster.

The name of a common event flag cluster is RSXCOMEFN. VAX/VMS qualifies this name with a UIC group number that protects the flags from processes with different UIC group numbers. Only processes with the correct UIC group number can associate with the common event flag cluster RSXCOMEFN for that UIC group.

A VAX/VMS native image can associate with the common event flags for a RSX–11 task image. To do this, the image must issue the Associate Commo Event Flag Cluster system service, giving an event flag number in the range 64 to 95 and giving the cluster name RSXCOMEFN. Note that the native image sees the common event flags as flags 64 through 95; an RSX–11 task image sees them as flags 33 through 64.

A native VAX/VMS process cannot associate with any common or group global event flag cluster outside its UIC group.

3.9.3 Group Global Event Flag Emulation

Group global event flags (flags 65 through 96) are visible to all task images associated with the flag cluster. Any process that is associated with a group global event flag cluster can read and set flags in that cluster. An RSX–11 task image is associated with a group global event flag cluster only if it issue a Create Group Global Event Flags directive.

The name of a group global event flag cluster is RSXGROUPEFN. VAX/VM qualifies this name with a UIC group number to protect the flags from processes with different UIC group numbers. Only processes with the correc UIC group number can associate with the group global event flag cluster RSXGROUPEFN for that UIC group.

A VAX/VMS native image can associate with a group global event flag cluste for an RSX–11 task image. To do this, the image must issue the Associate Common Event Flag Cluster system service, giving an event flag number in the range 96 to 127 and giving the cluster name RSXGROUPEFN. Note that the native image sees the common event flags as flags 96 through 127; an RSX–11 task image sees them as flags 65 through 96.

3.10 System Events

A **system event** in VAX/VMS is an occurrence that affects the ability of one or more processes in the system to execute. For example, an executing process can put itself into a wait state or it can set an event flag that makes another process a candidate for execution. VAX/VMS system events are similar to RSX–11 significant events. In VAX/VMS, however, a task image cannot request the declaration of a system event. There are no VAX/VMS equivalents for the RSX–11 directives Declare Significant Event and Wait For Significant Event. Issuing either of these directives has no effect on VAX/VMS; success status is returned to the issuing image.

As a result, task images that run under VAX–11 RSX must use only event flags, mailboxes, and ASTs for intertask communication; they cannot meaningfully use significant events.

11 System Status Codes

In VAX/VMS, the symbolic name for a system service status return has the following format:

```
SS$_name
```

When a task image issues an RSX–11 directive, VAX–11 RSX attempts to emulate the desired function and then returns a Directive Status Word (DSW) code to indicate success or failure to the image. In most cases, VAX–11 RSX calls the system service, which performs the equivalent of the requested RSX–11 function and converts the status code returned by the service to the equivalent RSX–11 DSW code. For example, the VAX/VMS code SS$_NORMAL becomes DSW code IS.SUC.

In some cases, however, a directive request results in a VAX/VMS error for which no exact RSX–11 equivalent exists. This situation occurs when a task image attempts to violate a VAX/VMS concept that has no RSX–11 equivalent. VAX–11 RSX handles the situation in one of the following ways:

- VAX–11 RSX returns a default DSW code.
- VAX–11 RSX returns a DSW code that is meaningful for the error, but that could not be returned for the directive if the task image were running under RSX–11.

Default return codes are used when no clear one-to-one relationship exists between VAX/VMS and RSX–11 codes, for example, when a VAX/VMS code is equally related to two DSW codes.

A new DSW code is returned when a VAX/VMS error has no counterpart in RSX–11. For example, IE.PRI indicates that the task image has attempted to issue a directive for which its process does not have the appropriate privilege.

In some cases after a directive failure, VAX–11 RSX returns an error code in the DSW that is more meaningful to I/O operations. In these cases, the high-order byte of the DSW contains 0. The DSW codes IE.PRI and IE.DUN (for Assign LUN) are examples of codes that are returned as bytes rather than words. RSX–11 task images can determine whether a DSW code is returned as a byte or word by testing the high-order byte of the DSW for 0.

DSW codes are listed in Section 6 with individual directive descriptions.

3.12 System Clock

On PDP–11 systems, the number of ticks per second varies depending on the type of clock used and its frequency. For the time-related directives, VAX–11 RSX emulates a 100-tick-per-second clock. This difference may affec emulation of the following directives, which have time-oriented arguments:

- Mark Time
- Run
- Get Time Parameters

3.13 Mailboxes

A **mailbox** is a record-oriented virtual device used in VAX/VMS for generalized communication among processes. VAX–11 RSX uses a mailbox to emulate the RSX–11 Receive Data, Receive Data Or Exit, Receive Data Or Stop, and Send Data directives. These directives are the normal means of intertask communication in the RSX–11 environment.

A mailbox has UIC-based protection associated with it. VAX–11 RSX can specify read and write privileges for system, owner, group, and world. Because the execute and delete privileges are not meaningful for mailboxes, VAX–11 RSX does not specify privileges for these functions.

When VAX–11 RSX creates a mailbox for emulating the Send Data and Receive Data directives, it specifies read access for the owner and write access for the group. The owner is the task image issuing the receive directives, and the group comprises the task images issuing the send directives. Owner and group are identified by the UIC under which they execute.

3.13.1 Mailboxes for Send Data and Receive Data Directives

When VAX–11 RSX loads a compatibility mode task image, the system creates a mailbox and associates it with the process just prior to actual task image execution. The mailbox is named as follows:

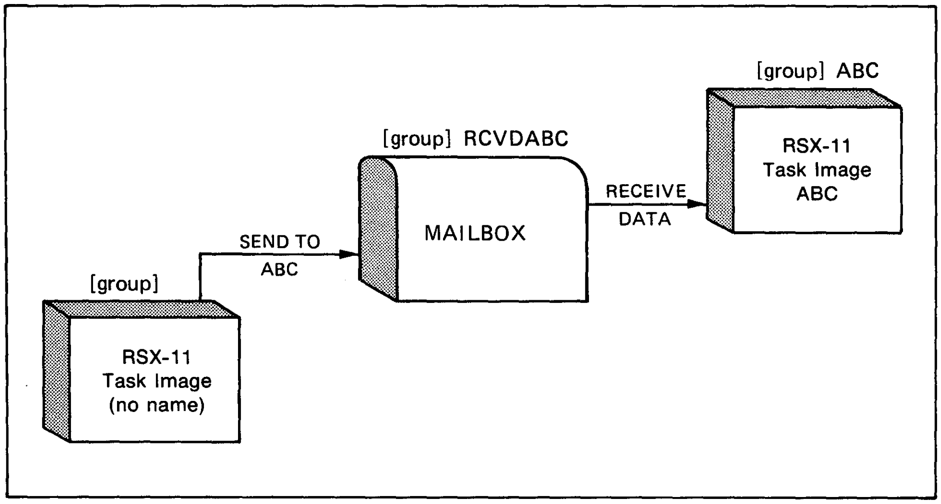
```
RCVDtaskname
```

The name is qualified by group number. Other task images that send data to the mailbox must be within the same group and have group or world privilege.

When VAX–11 RSX runs an RSX–11 task image, it converts the VAX/VMS process name to a form that is legal as an RSX–11 task name. If a TKB-assigned task name exists, VAX–11 RSX will use it. The conversion algorithm is explained in Section 2. The resulting task name is used to form the RCVDtaskname logical name.

Figure 3–2 illustrates the use of mailboxes for the Send Data and Receive Data directives.

Figure 3–2 Use of Mailboxes for Send Data and Receive Data Directives



ZK-850-82

3.13.2 I/O to Mailboxes

A mailbox has a device name of MBAn. The value of n is the unit number. VAX/VMS unit numbers are five-digit numbers ranging from 0 to 65,535. When a native image creates a mailbox, VAX/VMS assigns a unit number to it. Each time an image executes, the unit number assigned by VAX/VMS to the mailbox the image creates can change.

Because mailboxes are treated as devices under VAX/VMS, any RSX–11 task image can assign a channel to a mailbox by using its logical name and performing record I/O to it. The RSX–11 task image must use the logical name instead of the device name (MBAn) to refer to the mailbox, because RSX–11 task images can accept only a unit number in the range 0 to 255.

When a native mode image creates a mailbox that is intended for access by an RSX–11 task image, the native mode image must create a logical name resembling the format of an RSX–11 device name. If the native mode image does not do so, the RSX–11 task image will be unable to access the mailbox.

The logical name that the native mode image creates must have the two-ASCII-character device mnemonic $$. The logical name must also have an octal unit number within the range of 350 to 377 ($$350 to $$377). Logical names that meet these requirements are referred to as dynamic $$n logical names and are reserved for use by tasks that must create $$n names at run time. Refer to Section 2 for more information on device name mapping and $$n logical names.

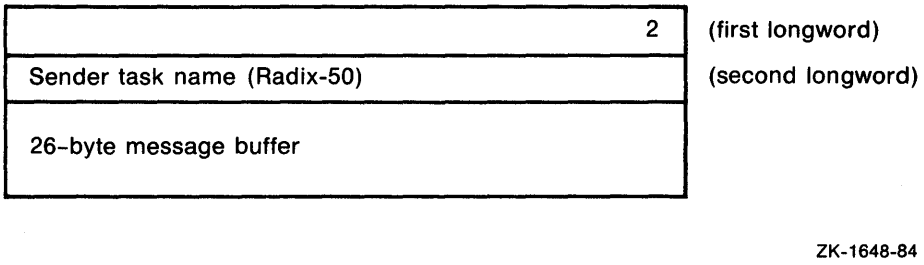
Mailboxes created by VAX–11 RSX for emulating Send Data and Receive Data directives do not have $$n logical names. Such mailboxes are not assigned directly by the RSX–11 task, and their existence is entirely transparent.

Either an RSX–11 task image or a VAX/VMS native image can assign a mailbox, but only a native VAX/VMS image can create a mailbox. A mailbox assigned by an RSX–11 task image must be either permanently available in the system or created by a native image. Assignment of a mailbox is treated the same as the assignment of other VAX/VMS devices for RSX–11 task images.

A mailbox can be shared by native images and RSX–11 task images. As a result, mailboxes provide a convenient means for native images to communicate with RSX–11 task images. The mailbox used for such communication can be created by a native image or created by VAX–11 RSX for emulating the Send Data and Receive Data directives.

A native image can send messages to a mailbox created for Send Data and Receive Data directive emulation by issuing write requests to it. The image can use either VAX RMS or the Queue I/O Request system service for I/O operations. The buffer written to this mailbox must have a particular format for the VAX–11 RSX AME to recognize it as a send data message (see Figure 3–3). A message sent to a native mode image from a compatibility mode task, using the Send Data directive, will have the same format.

Figure 3–3 Buffer Format for Send Data and Receive Data Emulation



The native mode task must calculate its own Radix–50 task name by taking the first six characters of its process name and converting them to Radix–50 format. Otherwise, the compatibility mode task cannot reply with the Send Data directive (see 3.13.1 for more information).

3.14 Memory Management

VAX/VMS memory management controls the use of both physical and virtual memory. It controls the use of physical memory through the implementation of two concepts:

- Balance sets
- Working sets

VAX/VMS divides virtual and physical address space into fixed-size memory areas called pages. When a process is created, VAX/VMS assigns it an initial number of pages that the process can use when it is in physical memory. The pages belonging to all processes in physical memory are referred to as the **balance set**. Processes in the balance set compete for access to system resources. The total number of pages that each process has in the balance set is called the process's **working set**.

3.14.1 Swapper Functions

In response to a need for additional physical memory resources, the VAX/VMS swapper can reduce the working set size of selected processes. If the demand for additional physical memory is not met by reducing working set sizes, VAX/VMS can write selected working sets to secondary storage. This method is known as **swapping**.

VAX/VMS reduces working sets and swaps processes to and from main memory to ensure that the highest priority processes are always available in memory for execution. The VAX/VMS swapper is more sophisticated than the RSX–11 checkpointing function, although it does provide an equivalent mechanism to allow emulation of the RSX–11 Enable Checkpointing and Disable Checkpointing directives.

The initial state of an RSX–11 task image in a process is for its swapping (checkpointing) function to be enabled. This state is identical to the initial state of a task image under RSX–11. To use the Disable Checkpointing directive, the process running an RSX–11 task image must have the VAX/VMS privilege to set its swapping mode, PSWAPM.

MCR commands can enable and disable swapping (for example, the SET PROCESS/[NO]SWAP command). The default state enables swapping. However, it is possible to change the initial state prior to running the task image.

Because VAX/VMS controls the use of physical memory by reducing working sets and swapping processes out of and into a balance set, it does not support partitioning of physical memory. As a result, when an RSX–11 task image issues a Get Partition Parameters directive, VAX–11 RSX returns a standard response for a hypothetical system-controlled partition named GEN. See Section 7 for a description of the Get Partition Parameters directive.

VAX–11 RSX ignores the partition name in the task image label block.

3.14.2 Pager Functions

The VAX–11 RSX **pager** controls the number of pages of a process's virtual address space that are in physical memory (that is, in the working set) at any time during process execution. When a process's working set is full and a page in virtual address space is required, the pager replaces the oldest page in the process's working set with the new page.

VAX–11 RSX does not support the RSX–11 memory management (PLAS) directives. Any task image issuing a memory management directive under VAX/VMS receives an error status return.

Every RSX–11 task image has 64K bytes (32K words) of virtual memory available to it. Because the address space is virtual rather than physical, RSX–11 task images can avoid overlaying; a task image executes more efficiently by depending on VAX/VMS memory management to determine which pages are needed in physical memory and when they are needed. Further efficiency can be gained by building shareable (/MU) RSX–11 task images. These RSX–11 task images can be partially shared under VAX/VMS. Note that the shareable feature is not supported by RSX–11M, although it is supported by RSX–11M–PLUS and Micro/RSX.

3.14.3 Memory Handling Options

VAX–11 RSX supports the use of overlays produced using the overlay descriptor language of TKB by RSX–11 task images. VAX–11 RSX loads overlays from the task image file at the appropriate point in task image execution.

VAX–11 RSX also supports use of shared regions by RSX–11 task images. RSX–11 task images can access both shared commons and libraries. Permanently available shared regions are identified to VAX/VMS by the system manager. Temporary regions are dynamically loaded when a task image requiring them executes.

In addition, VAX–11 RSX supports multiuser (shareable) task images. That is, when a task image is specified at task-build time as consisting of a shareable and nonshareable portion, VAX/VMS allows multiple users to access the shareable portion simultaneously. Each user has a private copy of the nonshareable portion. Note that such task images cannot be run under RSX–11M, although they can be run under RSX–11M–PLUS.

3.15 Global Sections

In VAX/VMS, **global sections** contain data or code that can be brought into memory and made available to processes for manipulation and execution. Global sections are created by execution of task images and by the system manager.

When a global section is created by VAX/VMS for native images, VAX/VMS assigns a set of characteristics to it. A global section can have the following characteristics:

- Read-only or read/write
- Temporary or permanent
- Group- or systemwide
- Disk file or paging file

A temporary global section remains in the system only as long as processes are mapped to it; when no processes are mapped to it, VAX/VMS deletes it automatically. A permanent global section remains in the system until it is explicitly deleted.

VAX/VMS provides group protection for group global sections. Any process can gain access to a system global section. A process must be privileged to create a permanent or system global section.

VAX/VMS imposes no limit on the number of global disk file sections to which a process can map. The system limit of paging file global pages is set during system generation.

A disk file global section is mapped to a disk file, whereas a paging file global section is not mapped to a disk file.

3.15.1 Mapping of Global Sections

When VAX–11 RSX loads an RSX–11 task image that was built specifying one of the options COMMON, LIBR, RESCOM, or RESLIB, it sets up the specified library or common for the image. When VAX–11 RSX loads the RSX–11 task image, it determines whether the global section for the library or common already exists.

If the global section exists, it is one of the following:

- A permanent global section created by the system manager
- A temporary global section created by VAX–11 RSX as a result of executing a previous RSX–11 task image

In either case, VAX–11 RSX maps the RSX–11 task image to the global section.

If the global section does not exist, VAX–11 RSX creates a temporary group global section for the library or common specified in the COMMON, LIBR, RESCOM, or RESLIB option to TKB. The task image file for either the library or common must be located on logical device and directory SYS$LIBRARY (LB:[1,1]). This can, however, be redirected by the use of logical names.

3.15.2 Characteristics of Global Sections

Global sections created by VAX–11 RSX for use by RSX–11 task images have the following characteristics:

- Global sections are accessed as either read-only or read/write and as either position-dependent or position-independent, according to the TKB specification.
- Global sections are group and temporary.
- The name for a global section is either the library name specified in a COMMON or LIBR option or the file name specified in a RESCOM or RESLIB option.

If the library or common area referred to is not found, VAX–11 RSX prints an error message on SYS$ERROR specifying the name of the library or common.

NOTE: **The disk file for a read/write global section is updated to reflect data manipulation by processes that map to it.**

VAX/VMS does not incorporate the concept of an installed global section that can be reinstalled to obtain a fresh copy. The disk file for a read/write global section is updated to reflect data written in the global section. Therefore, if it is necessary to maintain the original state of a read/write (common) global section, the user must keep a protected copy of the common file in a place other than SYS$LIBRARY.

3.16 RSX–11 Task Image Interface to the VAX/VMS I/O System

RSX–11 task images perform I/O by issuing requests to the FCS and RMS–11 level or by using the QIO$ directives. The number of steps required to perform each I/O operation varies depending on the level of the request. Figure 3–4 illustrates the interface between an RSX–11 task image and the VAX/VMS I/O system.

3.16.1 FCS Interface to the VAX/VMS I/O System

Task images issuing FCS requests use the same FCS routines available in RSX–11. Some of the routines have been modified to take advantage of VAX/VMS features, such as directory naming and file specification parsing. To take advantage of the modifications, the RSX–11 task image must be rebuilt under VAX–11 RSX using TKB. The VAX–11 RSX modifications are compatible with the RSX–11 version of FCS.

FCS runs in compatibility mode under VAX–11 RSX. When an RSX–11 task image issues an FCS request, FCS receives the request and reacts to it just as it would when it runs in RSX–11. That is, FCS issues the appropriate RSX–11 QIO$ directive.

From this point, the steps are identical to those taken when any RSX–11 task image issues a QIO$ directive:

1 The QIO$ directive traps to VAX–11 RSX.

2 VAX–11 RSX determines whether the QIO request was to a process-permanent file, such as TI or SYS$OUTPUT. If it was and that device is not a terminal, VAX–11 RSX issues a VAX–11 RMS $GET or $PUT request. Otherwise, VAX–11 RSX issues the VAX/VMS $QIO request that corresponds to the RSX–11 QIO request.

3 Upon completion of the QIO request, VAX–11 RSX returns the appropriate DSW code to the issuing task image.

4 Upon completion of the I/O operation, VAX–11 RSX returns status information in the I/O status block and sets an event flag or declares an AST, if requested.

If the routine to which the DSW code is returned is FCS, that component, in turn, makes the appropriate status return to the calling task image.

3.16.2 RMS–11 Interface to the VAX/VMS I/O System

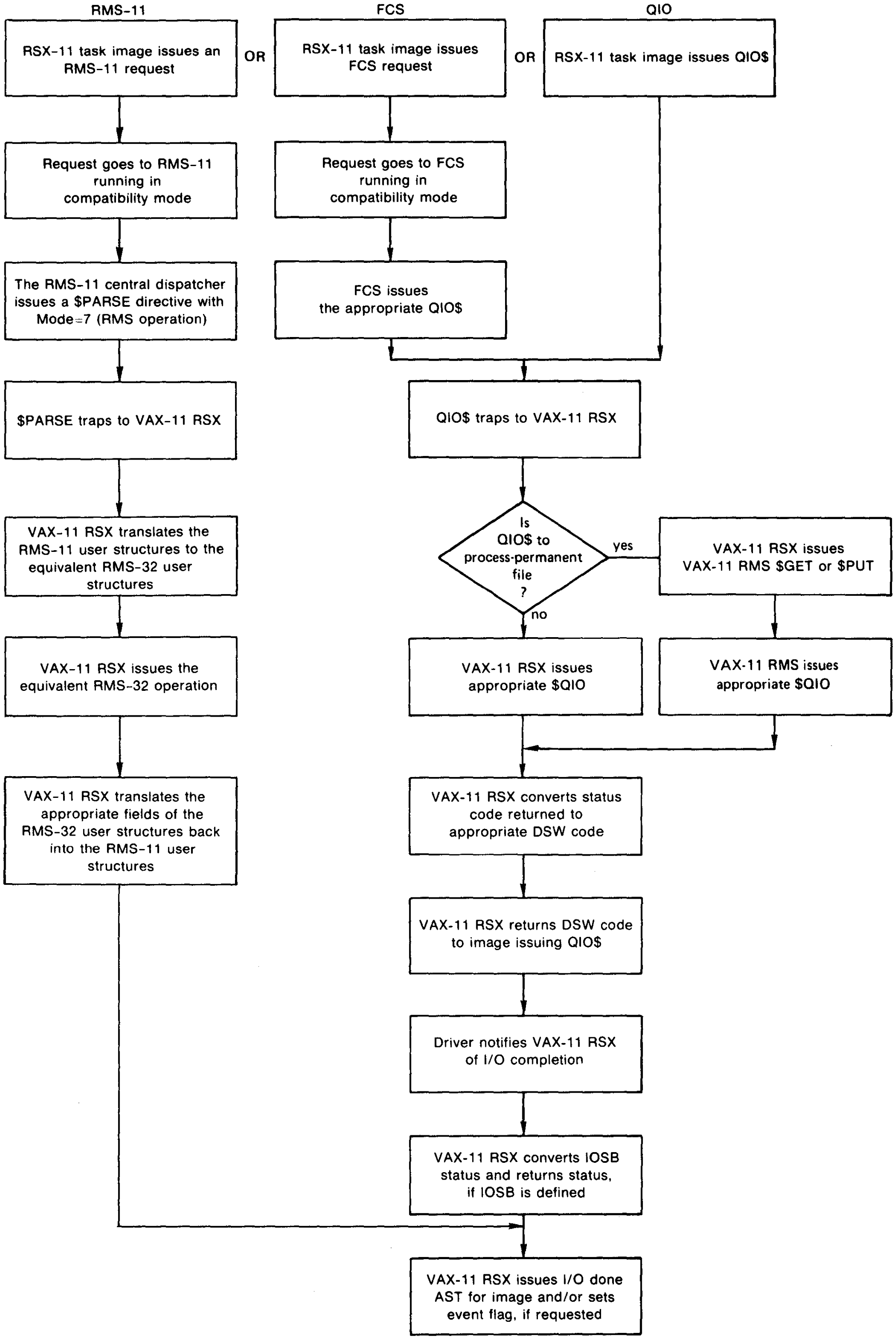
All RMS requests are dispatched through a central routine in RMS–11 as follows:

1 After some initial validation of the request and the RMS–11 user structure, the central routine issues the ELP$ directive with the mode set to 7. If the DSW is IE.SDP (indicating that the directive is not supported), RMS–11 assumes that the task is being run on an RSX–11 system.

2 RMS–11 proceeds to handle the request as it would under RSX–11. If the ELP$ directive succeeds, control is passed to VAX–11 RSX.

3 VAX–11 RSX translates the appropriate fields of the RMS–11 user structures to the corresponding fields on a VAX RMS user structure.

Figure 3–4 RSX–11 Task Image Interface to the VAX/VMS I/O System


RMS-11
FCS
QIO
RSX-11 task image issues an RMS-11 request
OR
RSX-11 task image issues FCS request
OR
RSX-11 task image issues QIO$
Request goes to RMS-11 running in compatibility mode
Request goes to FCS running in compatibility mode
The RMS-11 central dispatcher issues a $PARSE directive with Mode=7 (RMS operation)
FCS issues the appropriate QIO$
$PARSE traps to VAX-11 RSX
QIO$ traps to VAX-11 RSX
VAX-11 RSX translates the RMS-11 user structures to the equivalent RMS-32 user structures
Is QIO$ to process-permanent file ?
yes
no
VAX-11 RSX issues VAX-11 RMS $GET or $PUT
VAX-11 RSX issues the equivalent RMS-32 operation
VAX-11 RSX issues appropriate $QIO
VAX-11 RMS issues appropriate $QIO
VAX-11 RSX translates the appropriate fields of the RMS-32 user structures back into the RMS-11 user structures
VAX-11 RSX converts status code returned to appropriate DSW code
VAX-11 RSX returns DSW code to image issuing QIO$
Driver notifies VAX-11 RSX of I/O completion
VAX-11 RSX converts IOSB status and returns status, if IOSB is defined
VAX-11 RSX issues I/O done AST for image and/or sets event flag, if requested

ZK-1645-84

4 VAX–11 RSX then issues the VAX RMS request that corresponds to the RMS–11 request.

5 VAX–11 RSX translates the appropriate fields of the VAX RMS user structure back into the corresponding fields in the RMS–11 user structures.

6 Control passes back to the RMS–11 dispatch routine, which completes the operation by calling the user's success or error routine, if one is provided.

3.16.2.1 Advantages of the RMS–11 and VAX RMS Interface

There are several advantages to having the actual operations performed by VAX RMS:

- Record locking is now supported for RSX–11 tasks running in compatibility mode.
- A valid VAX/VMS file specification is accepted in a file string given to RMS–11.
- Operations over DECnet are supported.
- VAX RMS supports a different indexed file structure than RMS–11 (called Prolog 3).

3.16.2.2 Differences Between RMS–11 and VAX RMS

There are some differences between the RMS–11 and the VAX RMS user structures. In addition, there are some differences between what code is executed in the RSX–11 task when the task is run under VAX–11 RSX and when it is run under RSX–11. These differences result in the following incompatibilities:

- The DVI field in the VAX RMS name block is a counted string of 15 characters. In RMS–11, the DVI field consists of a two-character device name and a unit number. As a result, it may not be possible to return the device name in the RMS–11 name block. If this is the case, NS0 is returned. Under RSX–11, this is what an RMS–11 program would see if the operation was performed over the network. The correct device name remains associated with the LUN so that the RMS–11 operations will continue to work.
- In some cases, VAX–11 RSX and VAX RMS are more thorough than RMS–11 when validating user structures. This may cause a program that will run under RSX–11 to return errors when run under VAX–11 RSX. If such messages are returned, the user should correct the program or the errors will recur in the future. (RMS–11 may perform the same checks in the future.)

  For example, RMS–11 does not check the BLN field in XABs under some circumstances. However, VAX–11 RSX does perform the check and returns ER$XAB if the BLN field is not correct.
- When a task issues an RMS–11 request while running under VAX–11 RSX, most of the normal code in the task image is not exercised. In addition, space is not allocated from the task's buffer pools. As a result, a task that does operate correctly under VAX–11 RSX may not operate correctly under RSX–11. The errors may be due to an incorrect overlay structure or to inadequate buffer pool space.

17 Queue I/O Function Code Handling

VAX/VMS provides both device-independent and device-dependent functions at the Queue I/O Request service level. Device-independent functions include Read and Write Virtual Block, Read and Write Logical Block, and Read and Write Physical Block. Device-dependent functions include operations such as the handling of control and escape sequences for terminal I/O and positioning functions for magnetic tape.

For most RSX–11 function codes, VAX/VMS has a corresponding function code or system service. All disk function codes and most magnetic tape function codes have corresponding functions in VAX/VMS. However, two areas exist where discrepancies between RSX–11 and VAX/VMS device handling may appear:

- Handling of terminal devices
- Handling of spooled devices

Section 2 describes the implications of spooling for RSX–11 task images. Section 8 describes the differences for terminal handling.

PART II VAX–11 RSX Command Dictionary

4 MCR Commands

This section describes VAX–11 RSX MCR commands and their keywords. The syntax of the commands and restrictions on their use are given in detail. For a brief MCR command summary, see Table 4–1.

To be consistent with RSX–11 MCR commands, certain VAX–11 RSX MCR commands differ in syntax from their VAX/VMS DCL counterparts. Two examples of the differences follow.

- Unlike the DCL command ASSIGN, the MCR command ASN gives the option of placing an equal sign (=) instead of a space between the device (equivalence) name and the logical name. (If the equal sign is not used, a space must be inserted between the device name and the logical name.)
- The MCR command SET /UIC has the following format:

  ```
  SET /UIC=[g,m]
  ```

  The DCL command format, however, requires that the three command components be separated by a space, as follows:

  ```
  SET UIC [g,m]
  ```

RSX–11 utilities are not described in this section. Refer to the RSX–11M /M–PLUS documentation for information.

4.1 MCR Command Summary

The commands that make up VAX–11 RSX MCR come from two sources:

- From some of the general-purpose commands found in VAX/VMS DCL. Some of these commands are also found in RSX–11 MCR and DCL, although command formats may differ slightly.
- From some of the commands found in RSX–11 MCR (such as BYE, TIME, and UFD).

When a command exists only in VAX–11 RSX MCR or VAX/VMS DCL, a command or a utility that performs a similar function usually exists in the other command language.

VAX–11 RSX MCR provides some VAX/VMS DCL features in an RSX–11 command language environment. If you have been an RSX–11 user, VAX–11 RSX MCR can simplify your transition to VAX/VMS operation.

Table 4–1 summarizes the VAX–11 RSX MCR commands, grouped according to their general purpose: initialization, information, process control, program development, file handling, and file processing. In addition to the commands listed, VAX–11 RSX MCR recognizes commands that invoke the RSX–11 components listed in Section 1 and the various SET and SHOW commands associated with the VAX/VMS system.

Table 4–1 MCR Command Summary

Group	Command	Function
Initialization	ALLOCATE	Reserves an unmounted shareable device or an unallocated nonshareable device for exclusive use by the process
	ASN	Defines or deletes a logical name assignment
	CREATE/NAME_TABLE	Creates a new logical name table
	DEALLOCATE	Releases a previously allocated device
	DEASSIGN	Cancels one or more logical name assignments made with the ALLOCATE, ASN, DEFINE, or MOUNT command
	DEFINE	Creates a logical name table entry and assigns an equivalence name string to the specified logical name
	DMO	Releases (dismounts) a volume previously specified in a MOUNT command
	INITIALIZE	Initializes a Files–11 Structure Level 1 or 2 disk volume
	MOUNT	Makes a disk or magnetic tape volume or volume set available for processing
	SET COMMAND	Invokes the VAX/VMS Command Definition Utility to add commands to your process command table or to a specified command table file.

Table 4–1 (Cont.) MCR Command Summary

Group	Command	Function
	SET /UIC	Changes the directory default and user identification code (UIC) of the current process
Information	HELP	Provides information available in the system HELP files or in any HELP library you specify
	SHOW	Provides several options that let you display characteristics and defaults for your process, jobs, and devices
	TIME	Displays the current date and time
Communication	MAIL	Invokes the VAX/VMS Personal Mail Utility (MAIL), which is used to send messages to other users of the system
Process Control	ATTACH	Enables you to switch control of your terminal from your current process to another process in your job
	BYE	Terminates an interactive terminal session
	CANCEL	Cancels scheduled wake-up requests for a process
	CONTINUE	Resumes execution of an interrupted image or an indirect command file

Table 4–1 (Cont.) MCR Command Summary

Group	Command	Function
	LOGIN procedure	Verifies your right to use the system and allows you to log in to it
	LOGOUT	Terminates an interactive terminal session
	RESUME	Resumes execution of an indirect command file that was interrupted when a .PAUSE directive was issued by the indirect command file
	RUN	Runs images and, optionally, creates detached processes and subprocesses
	SET CONTROL	Determines whether control passes to the command interpreter when CTRL/Y is pressed
	SET PROCESS	Changes the execution characteristics associated with the specified process for the current terminal session or job
	SPAWN	Creates a subprocess of the current process; portions of the context of the subprocess are copied from the current process
	STOP	Terminates execution of the image currently running in a process
Program Development	DEBUG	Invokes the On-Line Debugging Tool (ODT) for an interrupted RSX–11 task image or invokes the VAX–11 Symbolic Debugger for an interrupted native mode image

Table 4–1 (Cont.) MCR Command Summary

Group	Command	Function
	DEPOSIT	Replaces the contents of specified locations in virtual memory
	EXAMINE	Examines the contents of specified locations in virtual memory
File Handling	APPEND	Adds the contents of one or more input files to the end of an output file
	BAD	Tests disks and tapes for the location and number of bad blocks
	COPY	Creates new files from one or more existing files
	CREATE	Creates one or more sequential disk files from records that follow the command in the input stream
	CREATE/DIRECTORY	Creates a new directory or subdirectory for cataloging files
	CREATE/FDL	Invokes the Create /FDL utility for using the specifications in an FDL file to create a new, empty data file
	DELETE	Deletes files, entries from print queues, and entries from batch job queues
	DIFFERENCES	Compares the contents of two disk files and creates a listing of those records that do not match
	DIRECTORY	Displays the names of one or more files in a directory or a subdirectory

Table 4–1 (Cont.) MCR Command Summary

Group	Command	Function
	DUMP	Displays or prints the contents of files or volumes in ASCII, hexadecimal, or octal representation
	PRINT	Queues files for printing
	PURGE	Deletes older versions of a file while saving the specified number of recent versions
	RENAME	Changes the name of a file
	SEARCH	Searches one or more files for a specified string or strings and lists all the lines that contain occurrences of the string(s)
	TYPE	Displays the contents of a file at your terminal
	UFD	Creates a directory or a subdirectory on a Files–11 Structure Level 1 or 2 disk
File Processing	EDT	Invokes the EDT interactive text editor to create or edit text files
	MERGE	Invokes the VAX–11 MERGE utility to combine similarly sorted input files and to create a single output file
	RUNOFF	Invokes the DIGITAL Standard Runoff (DSR) text formatter to format one or more ASCII files

Table 4–1 (Cont.) MCR Command Summary

Group	Command	Function
	RUNOFF/CONTENTS	Invokes the DIGITAL Standard Runoff (DSR) table of contents utility to create a file that can be processed by DSR to make a table of contents
	RUNOFF/INDEX	Invokes the DIGITAL Standard Runoff (DSR) indexing utility to create a file that can be processed by DSR to make an index
	SORT	Invokes the VAX–11 SORT utility to sort the records in a file based on one or more key fields within each record
	STOP/QUEUE/ABORT	Aborts a job that is currently being printed
	STOP/QUEUE/ENTRY	Deletes an entry from a batch queue while it is running
	STOP/QUEUE/REQUEUE	Stops the printing of the job currently being printed and places the job at the end of the output queue
	SUBMIT	Queues a batch job for processing

MCR does not check whether an invalid command was entered and uses command line parsing conventions that differ from DCL. If you enter a DCL command that is not found in MCR, MCR attempts to execute the command—but with erratic results. Problems are unpredictable and may not occur immediately after you enter the command. Table 4–2 lists the DCL commands not in MCR. If you need to use any of the commands in Table 4–2, use DCL instead of MCR.

Table 4–2 VAX/VMS DCL Commands Not Found in VAX–11 RSX MCR

Command	Command	Command
= (assignment)	EOD	READ
ACCOUNTING	EOJ	RECALL
ANALYZE[1]	EXCHANGE	REPLY
ASSIGN[1] (ASN in MCR)	EXIT	REQUEST
BACKUP	GOTO	SET CONTROL=T
CHECKSUM	IF	SET KEY
CLOSE	INITIALIZE/QUEUE	SET ON
CONNECT	INQUIRE	SET OUTPUT_RATE
CONVERT[1]	JOB	SET PROMPT
DEASSIGN/QUEUE	LEXICAL FUNCTIONS	SET VERIFY
DECK	LIBRARY	SHOW KEY
DEFINE/CHARACTERISTIC	LINK	START/CPU
DEFINE/FORM	MACRO	START/QUEUE[1]
DEFINE/KEY	MESSAGE	STOP/QUEUE/MANAGER
DELETE/CHARACTERISTIC	MONITOR	STOP/QUEUE/NEXT
DELETE/FORM	ON[1]	STOP/QUEUE/RESET
DELETE/KEY	OPEN	SYNCHRONIZE
DELETE/QUEUE	PASSWORD	UNLOCK
DELETE/SYMBOL	PATCH	WAIT
DISMOUNT (DMO IN MCR)	PHONE	WRITE
EDIT/EDT[1] (EDT IN MCR)		

[1] Indicates that the DCL command and all its subcommands are not provided by MCR. For example, the DCL command ASSIGN and its subcommands ASSIGN /MERGE and ASSIGN/QUEUE are not found in MCR.

4.2 MCR Command Descriptions

The descriptions of the commands given in this section observe the following conventions:

- Lowercase text indicates a variable whose actual value is determined when the command is entered.
- Square brackets ([]) indicate optional entries in a command line or file specification. Note that when you actually enter an option, you do not type the brackets.

  Square brackets are also used as part of the User File Directory (UFD) and for User Identification Code (UIC) syntax. When you use a UFD or UIC (for example, in a file specification), brackets are required syntax elements; that is, they do not indicate optional syntax entries.

Subheadings that appear in the command descriptions follow. Subheadings that do not apply to a particular command are omitted from the command description.

COMMAND NAME

This subheading identifies the MCR command name in capital letters. The text following the command name is an overview of the command's function.

Format

This subheading is followed by the command format, a list of command keywords (if any) and defaults, and a list of file keywords (if any) and defaults.

Restrictions

This subheading is followed by a list of restrictions related to the use of the command. Read the restrictions carefully before you use a command.

Prompts

This subheading is followed by the prompt(s), if any, evoked by the command.

Command Parameters

This subheading is followed by an explanation of each command parameter.

Description

Under this subheading is information that supplements the overview appearing after the command name.

Command Keywords

This subheading is followed by an explanation of each command keyword, if any.

Positional Keywords

This subheading is followed by an explanation of each positional keyword, if any.

File Keywords

This subheading is followed by an explanation of each file keyword, if any.

Examples

This subheading is followed by one or more examples illustrating the use of the command.

ALLOCATE

The ALLOCATE command provides exclusive access to a device and optionally establishes a logical name for the device. Once you allocate a device, other users cannot access the device until you specifically deallocate it or log out.

FORMAT

ALLOCATE *device-name[:][, . . .] [logical-name[:]]*

Command Keywords	**Defaults**
/[NO]LOG	*/LOG*
/GENERIC	

restrictions

- Once a device has been allocated, other users cannot access the device until you specifically deallocate it or log out.
- File-oriented devices (magnetic tapes and disks) cannot be allocated if they are mounted.
- Devices that are allocated but not mounted during an indirect command procedure are deallocated when the command procedure exits.

prompts

_Device: device-name or device-type
_Log name: logical-name

command parameters

device-name[, . . .]

Specifies the names or types of one or more devices to be searched for the allocation request. Device names can be generic device names, so that if no controller or unit number is specified, the system allocates the first available device that satisfies those components of the device names specified. If you specify more than one device name, the first nonallocated device in the list is allocated.

logical-name

Specifies a 1- to 255-character logical name to be associated with the allocated device. The logical name is placed in the process logical name table, with the name of the physical device allocated as its equivalence name. Subsequent references to the logical name result in automatic translation to the specified device name.

If you specify a trailing colon (:) with the logical name, the colon is removed from the name before the name is placed in the logical name table.

DESCRIPTION

The form of the ALLOCATE command that is compatible between VAX-11 RSX and RSX-11 MCR is

```
ALL device-name[:]
```

command keywords

/LOG
/NOLOG

Controls the message displayed at your terminal when you complete the allocate operation. The default is /LOG, which causes a message to be displayed indicating the name of the device allocated if the operation specifies a logical name that is currently assigned to another device.

The /[NO]LOG keyword can be placed anywhere in the command line.

/GENERIC

Allows you to perform a generic allocation operation by specifying a device type instead of a device name as the first command parameter. The first free, unallocated device of the specified name and type will be allocated.

EXAMPLES

1

```
> ALLOCATE  _DMB2:
%MCR-I-ALLOC, _DMB2: allocated
```

The ALLOCATE command requests the allocation of a specific RK06/RK07 disk drive, that is, Unit 2 on Controller B. The response from the ALLOCATE command indicates that the device was allocated successfully.

2

```
> ALLOCATE  MT,MF:   TAPE:
%MCR-I-ALLOC, _MTB2: allocated
   .
   .
   .
> SHOW LOGICAL TAPE
TAPE = _MTB2:     (process)
> DEALLOCATE TAPE
> DEASSIGN TAPE
```

The ALLOCATE command requests the allocation of any tape device whose name begins with MT or MF to be assigned the logical name TAPE. The ALLOCATE command locates an available tape device whose name begins with MT and responds with the name of the device allocated. (If no tape device beginning with MT had been found, the ALLOCATE command would have searched for a device beginning with MF.) Subsequent references to the device TAPE in user programs or command strings are translated to the device name MTB2.

When the tape device is no longer needed, the DEALLOCATE command deallocates it and the DEASSIGN command deletes the logical name. Note that the logical name, TAPE, was specified with a colon on the ALLOCATE command but that the logical name table entry does not have a colon.

APPEND

The APPEND command adds the contents of one or more specific input files to the end of a specific output file. The APPEND command is a variation of the COPY command.

FORMAT

APPEND *input-file-spec[, . . .] output-file-spec*

Command Keywords	**Defaults**
/BACKUP	*/CREATED*
/BEFORE[=time]	*/BEFORE=TODAY*
/BY_OWNER[=uic]	
/[NO]CONFIRM	*/NOCONFIRM*
/CREATED	*/CREATED*
/EXCLUDE=(file-spec[, . . .])	
/EXPIRED	*/CREATED*
/[NO]LOG	*/NOLOG*
/MODIFIED	*/CREATED*
/SINCE[=time]	*/SINCE=TODAY*

Positional Keywords	**Defaults**
/ALLOCATION=n	*See text*
/[NO]CONTIGUOUS	*See text*
/EXTENSION=n	
/FILE_MAXIMUM=n	
/[NO]NEW_VERSION	*/NONEW_VERSION*
/PROTECTION=(code)	*See text*
/[NO]READ_CHECK	*/NOREAD_CHECK*
/[NO]WRITE_CHECK	*/NOWRITE_CHECK*

restrictions

RSX–11 MCR does not support the APPEND command.

prompts

_From:	input-file-spec[, . . .]
_To:	output-file-spec

command parameters

input-file-spec[, . . .]

Specifies the name of one or more files to be appended. If you specify more than one input file, separate each file specification with either a comma (,) or a plus sign (+), which are equivalent.

You can use a wildcard character in place of the file name, type, or version field. When you use wildcard characters to concatenate Files–11 Structure Level 1 files, the input files are copied to the output file in random order. When you use wildcard characters to concatenate Files–11 Structure Level 2 files, the APPEND command copies files in alphanumeric order. When wildcard characters are used in the version field, files are copied in descending order by version number; that is, files with the same name are copied starting with the highest version number.

output-file-spec

Is the name of the output file. You must specify at least one field of the output file specification. For fields that you do not specify, the APPEND command uses the related field of the input file specification.

If you specify a wildcard character in any field of the output file specification, the APPEND command uses the related field of the input file specification(s).

If you specify only the device and directory portions of the output file specification, the APPEND command uses related fields of the input file specification.

DESCRIPTION

The APPEND command is similar in syntax and function to the COPY command. It adds the contents of one or more files to the end of an existing file without incrementing the version number. The /NEW_VERSION keyword causes the APPEND command to create a new output file if no file with that name exists.

command keywords

/BACKUP

Selects files according to the dates of their most recent backup. This keyword is only relevant when used with /BEFORE or /SINCE. Use of /BACKUP is incompatible with /CREATED, /MODIFIED, and /EXPIRED. The default is /CREATED.

/BEFORE[=time]

Selects only those files that are dated before the specified time.

You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values. You can also use the options TODAY, TOMORROW, and YESTERDAY. If no time is specified, TODAY is assumed.

/BY_OWNER[=uic]

Selects one or more files only if their owner User Identification Code (UIC) matches the specified owner UIC.

The UIC can be specified either as an eight-digit octal number or with the asterisk wildcard character.

If /BY_OWNER is specified without a UIC, the UIC of the current process is assumed.

/CONFIRM
/NOCONFIRM

Controls whether a request is issued before each individual APPEND operation to confirm that the operation should be performed on that file.

When the system issues the prompt, you can issue any of the following responses:

- Affirmative responses

  Yes
  True
  1

- Negative responses

  No
  False
  0
  <RET>

- Stop processing responses

  Quit
  <CTRL/Z>

- Continue processing without further prompts response

  ALL

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE). If you type a response other than one of those in the list, the prompt will be reissued.

/CREATED

Selects files based on their dates of creation. This keyword is relevant only when used with /BEFORE or /SINCE.

Use of /CREATED is incompatible with /MODIFIED, /EXPIRED, and /BACKUP. The default is /CREATED.

/EXCLUDE=(file-spec[, . . .])

Excludes any files that matched the listed file specification(s) from the APPEND operation. The file specification can contain a directory specification. If you specify two or more files, separate the file specifications with commas.

Wildcard characters are allowed in the file specifications. However, you cannot include either the device or directory field in the file specification(s) you supply with the /EXCLUDE keyword.

/EXPIRED

Selects files according to the dates on which they will expire. This keyword is relevant only when used with /BEFORE or /SINCE. Use of /EXPIRED is incompatible with /CREATED, /MODIFIED, and /BACKUP. The default is /CREATED.

/LOG
/NOLOG

Controls whether the file specification of each appended file is displayed.

When you specify the /LOG keyword, the APPEND command displays the file specifications of the input and output files and either the number of blocks or records appended after each append operation. The APPEND command displays the number of new files created at the end of command processing.

/MODIFIED

Selects files according to the dates on which they were last modified. This keyword is relevant only when used with /BEFORE or /SINCE. Use of /MODIFIED is incompatible with /CREATED, /EXPIRED, and /BACKUP. The default is /CREATED.

/SINCE[=time]

Selects only those files that are dated after the specified time.

You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values. You can also use the options TODAY, TOMORROW, and YESTERDAY. If no time is specified, TODAY is assumed.

positional keywords

/ALLOCATION=n

Forces the initial allocation of the new output file to the number of 512-byte blocks specified by n. The value n is interpreted as a decimal number by default.

/ALLOCATION is valid only if /NEW_VERSION is specified; that is, the allocation values are applied only if a new output file is actually created. If a new file is created and you do not specify /ALLOCATION, the initial allocation of the output file is determined by the size of the input file.

/CONTIGUOUS
/NOCONTIGUOUS

Indicates whether the output file is to be contiguous, that is, whether it must occupy consecutive physical disk blocks.

By default, the APPEND command creates an output file in the same format as the related input file. If an input file is contiguous, the APPEND command attempts to create a contiguous output file, but does not report an error if there is not enough space. If you append multiple input files of different formats to a single output file, the output file may or may not be contiguous. To ensure that the output file is contiguous, use the /CONTIGUOUS keyword.

/EXTENSION=n

Specifies the number of blocks to be added to the new output file each time it is extended.

/EXTENSION implies /NEW_VERSION, that is, the allocation values are applied only if a new output file is actually created.

/FILE_MAXIMUM=n

Specifies the maximum number of logical records that the output file can contain.

This keyword is valid only for new files. If you specify the /FILE_MAXIMUM keyword, the /NEW_VERSION keyword is assumed.

/NEW_VERSION
/NONEW_VERSION

Requests that, if a specific output file does not already exist, the APPEND command create one. Because /NONEW_VERSION is the default, the output file must already exist.

/PROTECTION=(code)

Defines the protection to be applied to the output file.

The protection is specified following the standard rules described in Section Any protection attributes not specified are taken from the current protection the output file or, if a new file is created, from the current default protection.

/READ_CHECK /NOREAD_CHECK

Indicates whether the APPEND command is to read a specific input file (or files) twice to verify that all records have been read correctly.

/WRITE_CHECK /NOWRITE_CHECK

Indicates whether the APPEND command is to read the output file after it ha been written. /WRITE_CHECK allows you to verify that the file has been appended successfully and that the output file can be read without error.

EXAMPLES

1

```
> APPEND [DAVIS]JOSEPH.OBJ [HICKS]EDWARD.OBJ
```

This command appends the file JOSEPH.OBJ from directory [DAVIS] to the file EDWARD.OBJ in directory [HICKS] on the default device.

2

```
> APPEND /LOG/NEW_VERSION FAST.DAT+SLOW.DAT+NORM.DAT TOTAL.DAT
%APPEND-I-CREATED, DBB1:[DAY]TOTAL.DAT;1 created
%APPEND-S-COPIED, DBB1:[DAY]FAST.DAT;2 copied to DBB1:[DAY]TOTAL.DAT;1 (10 blocks)
%APPEND-S-APPENDED, DBB1:[DAY]SLOW.DAT;5 appended to DBB1:[DAY]TOTAL.DAT;1 (8 records)
%APPEND-S-APPENDED, DBB1:[DAY]NORM.DAT;12 appended to DBB1:[DAY]TOTAL.DAT;1 (22 records)
```

This command concatenates the files FAST.DAT, SLOW.DAT, and NORM.DAT into a new file named TOTAL.DAT. The input and output files are on the default device and directory.

ASN

The ASN command defines or deletes a logical name assignment for a device in one of three logical name tables: process, group, or system.

FORMAT FOR ASSIGNING A LOGICAL NAME

ASN *[keywords] file-spec[=]logical-name*

FORMAT FOR DELETING A LOGICAL NAME

ASN *=logical-name [keywords]*

FORMAT FOR DELETING ALL LOGICAL NAMES

ASN =

Command Keywords	**Defaults**
/GBL	
/GROUP	
/TERM	*/TERM*

restrictions

- RSX-11 does not support the /GROUP keyword.
- To assign and delete logical names in the group or system logical name tables, GRPNAM or SYSNAM privilege is required.
- Assigning a disk device as SY: is not sufficient to cause the disk to become your default disk. To change your default disk, use a SET DEFAULT command.

prompt

_DEVICE: device-name

command parameters

file-spec

Indicates part or all of a file specification. When a physical device name is specified, terminate the device name with a colon (:). You can also specify a logical device name or pseudodevice name that translates to a physical device name or file specification.

logical-name

Specifies a logical name from 1 through 255 characters.

DESCRIPTION

The equal sign (=) between the two name parameters is optional. If you do not specify the equal sign, you must leave a space between the two parameters.

The SHOW LOGICAL command, described in the *VAX/VMS DCL Dictionary* can be used to display logical names in the process, group, and system tables

RSX–11 MCR supports the ASN command.

command keywords

/GBL

Indicates that the requested action is to occur in the system logical name table.

/GROUP

Indicates that the requested action is to occur in the group logical name table.

/TERM

Indicates that the requested action is to occur in the process logical name table. /TERM is the default.

EXAMPLES

1

```
> ASN MTB2:=MTO:
```

This command assigns the string MT0 as the logical name for device MTB2 in the process logical name table.

2

```
> SHOW LOGICAL MTO
   "MTO" = "MTB2:" (LNM$PROCESS_TABLE)
```

This command requests that the equivalence name associated with logical name MT0 be displayed.

ATTACH

Enables you to switch control of your terminal from your current process to another process in your job.

FORMAT

ATTACH *[process-name]*

Command Keyword	**Default**
/IDENTIFICATION=pid	*None*

restrictions

None.

prompt

_Process: process-name

command parameter

process-name

Specifies the name of the process to which the connection is to be made. Process names can contain from 1 to 15 alphanumeric characters. If a connection to the specified process cannot be made, an error message is displayed. You cannot connect to the process under the following circumstances:

- If the process is your current process
- If the process is not part of your current job
- If the process does not exist

If you specify the /IDENTIFICATION keyword, you cannot use the process name parameter. If you omit the /IDENTIFICATION keyword, you must specify a process name.

DESCRIPTION

The ATTACH command allows you to connect your terminal to another process. You can use ATTACH to change the control of your terminal from one subprocess to another subprocess or to the parent process.

When you issue the ATTACH command, the parent or "source" process is put into a hibernation state, and your terminal is connected to the specified destination process. You can use ATTACH to connect to a subprocess that is part of a current job left hibernating as a result of the SPAWN/WAIT command or another ATTACH command as long as the connection is valid. (No connection can be made to the current process. Also, no connection can be made to a process that is not part of the current job, that does not exist, or that does not share the same input stream as the current process.)

You can also use ATTACH in conjunction with the SPAWN/WAIT command to return to a parent process without terminating the created subprocess. See the description of the SPAWN command for more details.

command keyword

/IDENTIFICATION=pid

Specifies the process identification (PID) of the process to which terminal control will be switched. When you specify a PID, you can omit the leading zeros.

If the process is not part of the current job or does not exist, an error message is displayed.

When you use the /IDENTIFICATION qualifier, do not specify the process name parameter. If you omit the /IDENTIFICATION qualifier, you must specify a process name.

EXAMPLES

1
```
> ATTACH JONES_2
```

The ATTACH command changes the terminal's control to the process JONES_2.

2
```
> ATTACH/IDENTIFICATION=30019
```

The ATTACH command changes the terminal's control from the current process to a process having the PID 30019. Notice that since the /IDENTIFICATION keyword is specified, the process-name parameter is omitted.

BAD

The BAD command tests disks and tapes for the location and number of bad blocks.

FORMAT

BAD *device-name:*

Command Keywords	**Defaults**
/[NO]EXERCISE=PATTERN=list	*/NOEXERCISE*
/LIST	
/MANUAL	
/RETRY	*/NORETRY*
/UPDATE	

restrictions

This command functions similarly to the DCL command ANALYZE/MEDIA. Therefore, you may want to use the DCL command. See the *VAX/VMS Utilities Reference Volume.*

prompt

_Device: device-name:

command parameter

device-name:

Specifies the device containing the volume that BAD will analyze. The device name has the form ddcu: or logical name.

command keywords

/EXERCISE=PATTERN=list
/NOEXERCISE

Locates bad blocks by a user-specified longword data pattern. This allows you to specify the value of a test pattern for use as the "worst case" . You may specify up to an octaword of test pattern data in decimal (%D), hexadecimal (%X), or octal (%O) radixes. The default radix is decimal.

The pattern is specified in longwords. If two or more longwords are specified, they must be enclosed in parentheses and separated by commas.

/LIST

Causes all bad blocks to be printed by cylinder, track, and sector on your terminal. Manually entered blocks that are shown to be as reliable are not listed.

/MANUAL

Prompts you for bad block information and then performs data pattern checking. Any block that you enter is included in the bad block descriptor file or the SDBBF.

/RETRY
/NORETRY

Attempts a recovery of soft errors by the device driver. This means that soft errors, such as an Error Correction Code (ECC) correctable error, are recovered and the bad block is marked as good.

/UPDATE

Reads the bad block descriptor file immediately and prompts you for additional bad blocks. This keyword does not cause the writing of pattern checks.

EXAMPLES

1

```
> BAD device/LIST
```

The command in this example tests the media and displays information about the bad blocks at your terminal.

2

```
> BAD/EXERCISE=PATTERN=(%XFFEEFFEE,%XBADBADBA) device
```

This command specifies a hexadecimal test pattern two longwords in length.

BYE

The BYE command terminates an interactive terminal session. The system displays a termination message and performs any necessary cleanup operations, such as terminating the current task image if one exists, dismounting any private volumes that remain mounted, and deallocating devices. Finally, it deletes your process and subprocesses, if there are any.

FORMAT

BYE

Command Keywords	Defaults
None	*None*

restrictions

None.

prompts

None.

DESCRIPTION

In VAX-11 RSX MCR, the LOGOUT command and the BYE command perform the same function.

RSX-11 MCR supports the BYE command.

EXAMPLE

```
> BYE
  FOLEY    logged out at 13-SEP-1984 13:33:26.57
```

CANCEL

The CANCEL command requests the system to cancel a wake-up request for a specific process.

FORMAT

CANCEL *[process-name]*

Command Keyword	**Default**
/IDENTIFICATION=pid	*None*

restrictions

- You must have GROUP privilege to cancel wake-up requests for another process in the same group and WORLD privilege to cancel wake-up requests for another process in the same system. No privilege is required for a process to issue a cancel request for one of its subprocesses.
- RSX–11 MCR supports the CANCEL command when the VAX–11 RSX MCR process name is the same as an RSX–11 task name.

prompts

None.

command parameter

process-name

Specifies the name of the process for which wake-up requests are to be canceled. Because process names are implicitly qualified by UIC group number, you can cancel requests only for processes within your group by specifying a process name.

DESCRIPTION

The CANCEL command is used to cancel scheduled wake-up requests for the specified process. A wake-up request can be the result of any of the following:

- The RUN$ directive
- The Schedule Wake-up system service
- Any RUN commands that request time synchronization, such as the /SCHEDULE keyword for the MCR command RUN

Cancel does not affect the execution of an active process. It only removes time-based requests from the timer queue.

VAX–11 RSX uses the Schedule Wake-up system service to emulate the RSX–11 RUN$ directive. For additional information, refer to Section 3.

The STOP command can be used to delete a hibernating process for which all wake-up requests are canceled.

You can determine whether a subprocess has completed execution of an image by issuing the SHOW PROCESS command with the /SUBPROCESSES keyword described in the *VAX/VMS DCL Dictionary.*

command keyword

/IDENTIFICATION=pid

Specifies the process identification code (PID) of the process for which wake-up requests are to be canceled. If you specify both a process name and identification, the CANCEL command ignores the process name. If you specify neither, wake-up requests for your current process are canceled. When you specify the process identification, you can omit leading zeros. The PID value is interpreted as a hexadecimal number.

EXAMPLES

1

```
> CANCEL PROCA
```

This command cancels wake-up requests for a process executing in the same group as the requesting process. If the process is not a subprocess of the requesting process, the GROUP privilege is required.

2

```
> CANCEL/IDENTIFICATION=13012A
```

This command cancels wake-up requests for the process whose process identification number is 13012A. The command can be executing outside the group requesting a process; however, appropriate privilege is required.

CONTINUE

The CONTINUE command resumes execution of a command, program, or indirect command file that was interrupted by any of the following:

- Pressing CTRL/Y or CTRL/C
- A PAUSE statement issued by a FORTRAN program

For convenience, the CONTINUE command can be abbreviated to a single letter, C.

FORMAT

CONTINUE

Command Keywords	**Defaults**
None	*None*

restrictions

- When a program is interrupted or pauses, only those actions in the list that follows can be taken if the program is to continue to execute. Use of any commands not listed will cause the interrupted image to be terminated:

  ALLOCATE
  ASN
  ATTACH
  CANCEL
  CONTINUE
  CREATE/NAME_TABLE
  DEALLOCATE
  DEASSIGN
  DEBUG
  DEFINE
  DEPOSIT
  EXAMINE
  SET CONTROL
  SET DEFAULT
  SET PROCESS
  SET PROTECTION /DEFAULT
  SET UIC
  SHOW DEFAULT
  SHOW QUOTA
  SHOW PROTECTION
  SHOW STATUS
  SHOW SYMBOL
  SHOW TIME
  SHOW TRANSLATION
  SPAWN
  STOP
  TIME

- You cannot continue execution of interrupted images that are privileged, that is, images that have been installed with privileges. See the description of the Install Utility in the *VAX/VMS Utilities Reference Volume* for more information on creating privileged images.
- RSX-11 MCR does not support the CONTINUE command.

prompts

None.

EXAMPLE

```
> RUN TEST
      .
      .
      .
  CTRL/Y INTERRUPT
> TIME
  13-SEP-1984  14:00:55
> CONTINUE
```

The RUN command initiates execution of the image TEST. Pressing CTRL/Y interrupts TEST. The TIME command causes VAX–11 RSX to display the date and time. The CONTINUE command then requests resumption of TEST at the point where it was interrupted.

COPY

The COPY command creates a new file from one or more existing files. This command can be used to perform the following functions:

- Copy one input file to an output file
- Concatenate more than one input file into a single output file
- Copy a group of input files to another group of output files

FORMAT

COPY *input-file-spec[, . . .] output-file-spec*

Command Keywords	**Defaults**
/BACKUP	*/CREATED*
/BEFORE[=time]	*/BEFORE=TODAY*
/BY_OWNER[=uic]	
/[NO]CONCATENATE	*/CONCATENATE*
/[NO]CONFIRM	*/NOCONFIRM*
/CREATED	*/CREATED*
/EXCLUDE=(filespec[, . . .])	
/EXPIRED	*/CREATED*
/[NO]LOG	*/NOLOG*
/MODIFIED	*/CREATED*
/SINCE[=time]	*/SINCE=TODAY*

Positional Keywords	**Defaults**
/NOALLOCATION=n	*See text*
/[NO]CONTIGUOUS	*See text*
/EXTENSION=n	*See text*
/FILE_MAXIMUM=n	
/[NO]OVERLAY	*/NOOVERLAY*
/PROTECTION=(code)	*See text*
/[NO]READ_CHECK	*/NOREAD_CHECK*
/[NO]REPLACE	*/NOREPLACE*
/[NO]TRUNCATE	*/NOTRUNCATE*
/VOLUME=n	*See text*
/[NO]WRITE_CHECK	*/NOWRITE_CHECK*

restrictions

- Restrictions apply to the /FILE_MAXIMUM keyword. See the keyword description for more information.
- RSX-11 MCR does not support the COPY command. However, under RSX-11, you can use PIP instead of COPY.

prompts

_From:	input-file-spec[, . . .]
_To:	output-file-spec

command parameters

input-file-spec[, . . .]

Specifies the name of one or more files to be copied. If you specify more than one input file, separate the file specifications with either a comma (,) or a plus sign (+).

You can use wildcard characters in place of the file name, type, or version field. Then, all files that satisfy the remaining components are copied.

output-file-spec

Specifies the name of the output file (the file copied to). You must specify at least one field of the output file specification. For fields that you do not specify (file name, file type, version number), the COPY command uses the related field of the input file specification(s). If the device and directory are not specified, your current default device and directory are used.

If you specify a wildcard character in any field of the output file specification, the COPY command uses the related field of the input file specification(s) and does not concatenate the output.

If you specify only the device and directory portion of the output file specification, the COPY command uses related fields of the input file specification(s) and does not concatenate the output.

DESCRIPTION

When you use wildcard characters and /CONCATENATE to concatenate Files–11 Structure Level 1 files, the input files are copied to the output file in random order. When you concatenate Files–11 Structure Level 2 files, the COPY command copies files in alphanumeric order. When wildcard characters are used in the version field, files are copied in descending order by version number; that is, files with the same name are copied starting with the highest version number.

If you specify a list of files and do not use wildcard characters, the files are copied as concatenated files in the order specified.

command keywords

/BACKUP

Selects files according to the dates of their most recent backup. This keyword is relevant only when used with the /BEFORE or /SINCE keyword. Use of the /BACKUP keyword is incompatible with /CREATED, /MODIFIED, and /EXPIRED. /CREATED is the default.

/BEFORE[=time]

Selects only those files dated before the specified time.

You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values. You can also use the options TODAY, TOMORROW, and YESTERDAY. If no time is specified, TODAY is assumed.

/BY_OWNER[=uic]

Selects one or more files only if their owner User Identification Code (UIC) matches the specified owner UIC.

The UIC can be specified either as an eight-digit octal number or with the asterisk wildcard character.

If the /BY_OWNER keyword is specified without a UIC, the UIC of the current process is assumed.

/CONCATENATE /NOCONCATENATE

When multiple input file specifications are provided, this keyword indicates whether a single output file is to be created from all files that satisfy the inpu file specification.

By default, the COPY command produces a single output file from multiple input files unless one of the following conditions is true:

- One or more wildcard characters are present in the output file specificatior
- Only the device and directory portions of the output file specification are provided

/CONFIRM /NOCONFIRM

Controls whether a request is issued before each individual COPY operation to confirm that the operation should be performed on that file.

When the system issues the prompt, you can issue any of the following responses:

- Affirmative responses

  Yes
  True
  1

- Negative responses

  No
  False
  0
  <RET>

- Stop processing responses

  Quit
  <CTRL/Z>

- Continue processing without further prompts response

  ALL

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE). If you type a response other than one of those in the list, the prompt will be reissued.

/CREATED

Selects files based on their dates of creation. This keyword is relevant only when used with the /BEFORE or /SINCE keyword. Use of the /CREATED keyword is incompatible with /MODIFIED, /EXPIRED, and /BACKUP.

/EXCLUDE=(file-spec[, . . .])

Excludes any files that match the listed file specification(s). The file specification can contain a directory specification. If you specify two or more files, separate the file specifications with commas.

You can include wildcard characters in the file specifications. However, you cannot include either the device or directory field in the file specification you supply with the /EXCLUDE keyword.

/EXPIRED

Selects files according to the dates on which they will expire. This keyword is relevant only when used with the /BEFORE or /SINCE keyword. Use of the /EXPIRED keyword is incompatible with /CREATED, /MODIFIED, and /BACKUP. /CREATED is the default.

/LOG
/NOLOG

Indicates that the COPY command is to display the file specifications of each input file copied.

The COPY command displays either the number of records or the number of blocks copied, depending on whether the file is copied on a record-by-record or block-by-block basis.

At the end of command processing, the COPY command displays the number of files created.

/MODIFIED

Selects files according to the dates on which they were last modified. This keyword is relevant only when used with the /BEFORE or /SINCE keyword. Use of the /MODIFIED keyword is incompatible with /CREATED, /EXPIRED, and /BACKUP. /CREATED is the default.

/SINCE[=time]

Selects only those files dated after the specified time. You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values. You can also use the keywords TODAY, TOMORROW, and YESTERDAY. If no time is specified, TODAY is assumed.

positional keywords

/ALLOCATION=n

Forces the initial allocation of the output file to the number of blocks specified by n. If this keyword is not specified, the initial allocation of the output file is determined from the size of the input file being copied.

/CONTIGUOUS
/NOCONTIGUOUS

Indicates whether the output file is to be contiguous, that is, whether it must occupy consecutive physical disk blocks. This keyword can be applied to an output file only.

By default, the COPY command creates an output file in the same format as the corresponding input file. If the input file is contiguous, the COPY command attempts to create a contiguous output file. The command does not report an error if the output file is not contiguous.

If multiple input files of different formats are copied into a single output file, the output file may or may not be contiguous. To ensure that the output file is contiguous, use the /ALLOCATION and /CONTIGUOUS keywords.

The /CONTIGUOUS keyword has no effect when you copy files to or from tapes, because the size of the input file cannot be determined until the file is read. If you copy a file from tape, use the COPY command twice: once to copy the file from tape, and again to create a contiguous file.

/EXTENSION=n

Specifies the number of blocks to be added to the output file each time the output file is extended. If this keyword is not specified, the default extension attribute of the output file is determined from the input file.

The owner UIC of the output file is the UIC of the current process.

/FILE_MAXIMUM=n

Specifies the maximum number of logical records that the output file can contain. This keyword is valid only for relative files.

/OVERLAY
/NOOVERLAY

Indicates whether data in the input file is to be copied into an existing output file overlaying the existing data. If the new file is shorter than the overlaid file, the file is truncated at the end of the new file. The new file is in the same position as the overlaid file.

/OVERLAY is ignored if the output file is written to a non-file-structured device.

/PROTECTION=(code)

Defines the protection to be applied to the output file. The protection is specified following the standard rules described in Section 2. Any protection attributes not specified are taken from the current protection of the input file.

The owner UIC of the output file is the UIC of the current process.

/READ_CHECK
/NOREAD_CHECK

Indicates whether the COPY command is to read a specific input file or files twice to verify that all records have been correctly read. By default, records are read only once.

/REPLACE
/NOREPLACE

Requests that, if a file already exists with the same file specification as the output file, the existing file be deleted and replaced with the contents of the input file(s).

By default, the COPY command creates a new version of a file if a file with that specification already exists, incrementing the version number.

Under /NOREPLACE, an error is signaled when a conflict in version numbers occurs.

/TRUNCATE /NOTRUNCATE

Controls whether the COPY command will truncate an output file at the end-of-file when copying it. By default, the COPY command uses the allocation of the input file to determine the size of the output file.

When copying multiple input files into one output file, you can save space by specifying /TRUNCATE.

If you do not specify /TRUNCATE, all the blocks allocated to the specific file(s) are copied to the output file(s).

/VOLUME=n

Requests that the COPY command place the entire output file on the specified relative volume of a multivolume set.

If you omit the /VOLUME keyword, the file is placed in an arbitrary position within the multivolume set.

/WRITE_CHECK /NOWRITE_CHECK

Indicates whether the COPY command is to read the output file after it has been written to. /WRITE_CHECK verifies that the file has been copied successfully and that the output file can be read without error. By default, the output records are not read after writing.

EXAMPLES

1

```
> COPY [DAVIS]JOSEPH.OBJ [HICKS]EDWARD.OBJ
```

This command copies the file JOSEPH.OBJ from directory [DAVIS] to directory [HICKS] on the default device and names the file EDWARD.OBJ.

2

```
> COPY *.FOR ALLSRC.FOR
```

This command concatenates the highest versions of all FORTRAN source files in the default directory into a file named ALLSRC.FOR.

3

```
> COPY [JONES]ABC.DAT *
```

This command copies the file ABC.DAT from the directory [JONES] to the current default directory, preserving the name as ABC.DAT.

CREATE

The CREATE command creates one or more sequential disk files from records that follow the command in the input stream.

FORMAT

CREATE *file-spec[, . . .]*

Command Keywords	**Defaults**
/[NO]LOG	*/NOLOG*
/OWNER_UIC=uic	
/PROTECTION=(code)	*See text*
/VOLUME=n	*See text*

restrictions

Restrictions apply to the /OWNER_UIC keyword. See the keyword description for more information.

prompt

_File: file-spec[, . . .]

command parameter

file-spec[, . . .]

Specifies the name of one or more input files to be created.

If you omit either the file name or the file type, the CREATE command does not supply any defaults; the file name or file type is null. If you do not specify a file version number, and a file already exists with the same file name and file type as the file specification, the CREATE command creates a new version of the file.

No wildcard characters are allowed in the file specifications.

DESCRIPTION

The CREATE command creates a new sequential disk file. The contents of the file are determined by what you type after the command line. Each separate line that you type becomes a record in the newly created file. When you finish entering the records, use CTRL/Z to signal the end of the input.

When you issue the CREATE command from a command procedure file, the system reads all subsequent records in the command procedure file into the new file until it encounters a dollar sign in the first position in a record.

If you use an existing file specification with the CREATE command, the newly created file has a higher version number than any existing file(s) with the same specification.

command keywords

/LOG
/NOLOG

Controls whether the CREATE command displays the file specification of each file that it has created.

By default, the CREATE command does not display the names of files after it creates them.

/OWNER_UIC=uic

Specifies the User Identification Code (UIC) to be associated with the file being created. Specify the UIC in the following format:

[g,m]

where

g is an octal number in the range 0 through 37777 representing the group number

m is an octal number in the range 0 through 177777 representing the member number

The square brackets are required in the UIC specification. For example, /OWNER_UIC=[000,000].

If you do not specify an owner UIC when you create a file, the command assigns your UIC to the file.

You must have the SYSPRV user privilege to specify a UIC other than your own.

/PROTECTION=(code)

Defines the protection to be applied to the file.

If you do not specify a value for each access category, or if you omit the /PROTECTION keyword, the command applies the current default protection for each unspecified category.

/VOLUME=n

Requests that each file be placed on the specified relative volume number of a multivolume set.

If you omit the /VOLUME keyword, the files are placed arbitrarily within the multivolume set.

EXAMPLES

1

```
> CREATE   A.DAT,B.DAT
Input line one for A.DAT . . .
Input line two for A.DAT . . .
     .
     .
     .
<CTRL/Z>EXIT
Input line one for B.DAT . . .
Input line two for B.DAT . . .
     .
     .
     .
CTRL/Z EXIT
>
```

After you issue the CREATE command from the terminal, the system reads input lines into the sequential file A.DAT until CTRL/Z terminates the first input. The next set of input data is placed in the second file, B.DAT. Again, CTRL/Z terminates the input.

2

```
> CREATE  TOLEDO::XXX1:[MODEL.TEST]TEST.DAT
A.1, A.2, A.3, A.4, A.5
52974,53785,78935,57832,78549
B.1, B.2, B.3, B.4, B.5
76984,67486,39076,38676,57681
CTRL/Z EXIT
>
```

The CREATE command creates a sequential file named TEST.DAT on the remote node TOLEDO. The contents of the files are the four records that follow the CREATE command line. The CTRL/Z entry indicates the end of the file.

CREATE/DIRECTORY

The CREATE/DIRECTORY command creates a new directory or subdirectory for cataloging files. The /DIRECTORY keyword is required.

FORMAT

CREATE/DIRECTORY *directory-spec[, . . .]*

Command Keywords	**Defaults**
/[NO]LOG	*/NOLOG*
/OWNER_UIC[=option]	*See text*
/PROTECTION=(code)	*See text*
/VERSION_LIMIT=n	*See text*
/VOLUME=n	*See text*

restrictions

- To create a first-level directory, you must have write access to the Master File Directory (MFD) of the volume on which you are creating the directory.
- To create a subdirectory, you must have write access to the lowest-level directory that currently exists.

prompt

_File: directory-spec[, . . .]

command parameter

directory-spec[, . . .]
Specifies the name of one or more directories or subdirectories to be created.

The directory specification(s) must contain a directory name. A device name is optional. When you create a subdirectory, separate the names of the directory levels with periods.

No wildcard characters are allowed in the directory specification.

DESCRIPTION

The CREATE/DIRECTORY command is used to create new directories as well as subdirectories. Special privileges are needed to create new first-level directories (see restrictions). Generally, users have sufficient privileges to create subdirectories in their own directories. Use the SET DEFAULT command to change from one directory to another.

command keywords

/LOG
/NOLOG
Controls whether the CREATE/DIRECTORY command displays the directory specification of each directory after creating it.

By default, the CREATE/DIRECTORY command does not display the name of each directory after creating it.

/OWNER_UIC[=option]

Specifies the User Identification Code (UIC) to be associated with the directory being created. Specify the UIC with one of the following options:

- The UIC specification in the following format:

  [g,m]

  where

  g is an octal number in the range 0 through 37777 representing the group number

  m is an octal number in the range 0 through 177777 representing tne member number

  The brackets are required in the UIC specification. For example, /OWNER_UIC=[000,000].

- The option PARENT. This option specifies that the directory's UIC be changed to match the UIC of the directory in which it is cataloged.

  If you do not specify the /OWNER_UIC keyword when you create a directory, the command assigns ownership as follows:

- If you specify the directory name in either alphanumeric or subdirectory format, ownership defaults to your UIC.

- If you specify the directory name in UIC format, ownership defaults to the UIC in the directory name.

/PROTECTION=(code)

Defines the protection to be applied to the directory.

If you do not specify a value for each access category or if you omit the /PROTECTION keyword when you create a directory, the command uses the protection in effect for the next-highest-level directory, less any delete access for each unspecified category. If you are creating a first-level directory, then the next-highest-level directory is the MFD. (The protection of the MFD is established by the INITIALIZE command.)

/VERSION_LIMIT=n

Specifies that by default no more than n versions of each file created in this directory are to be kept. Whenever n versions exist and a new version is created, the lowest version is deleted automatically. If you omit the /VERSION_LIMIT keyword, the default is the number of versions permitted for the directory at the next-higher level.

When you specify /VERSION_LIMIT=0, the system creates a directory with no default version limit.

/VOLUME=n

Requests that the directory file be placed on the specified relative volume number of a multivolume set.

If you omit the /VOLUME keyword, the file is placed arbitrarily within the multivolume set.

EXAMPLES

1

```
> CREATE/DIRECTORY XXX2:[MALCOLM]
```

The CREATE/DIRECTORY command creates a directory named [MALCOLM] on the device XXX2.

2

```
> CREATE/DIRECTORY [MALCOLM.SUB]
> SET DEFAULT [MALCOLM.SUB]
```

The CREATE/DIRECTORY command creates a subdirectory named [MALCOLM.SUB]. This directory file is placed in the directory named [MALCOLM]. The command SET DEFAULT [MALCOLM.SUB] changes the current default directory to this subdirectory. All files subsequently created are cataloged in [MALCOLM.SUB].

3

```
> CREATE/DIRECTORY/PROTECTION=(SYSTEM:RWED,OWNER:RWED,GROUP,WORLD)-
>_[MALCOLM.SUB.HLP]
```

The CREATE/DIRECTORY command creates a subdirectory named [MALCOLM.SUB.HLP]. The protection on the subdirectory allows read, write, execute, and delete access for the system and owner categories, but it prohibits all access for the group or world categories.

CREATE/FDL

This command invokes the Create/FDL Utility (CREATE/FDL) to use the specifications in an FDL file to create a new, empty data file. You use this utility when you want to create a data file from a particular FDL specification. The /FDL qualifier is required.

For a complete description of the Create/FDL Utility, including more information about the CREATE/FDL command and its qualifier, see the description of the FDL Utility in the *VAX/VMS Utilities Reference Volume.*

FORMAT

CREATE/FDL=fdl-file-spec *[file-spec]*

Command Keyword	**Default**
/[NO]LOG	*/LOG*

CREATE/NAME_TABLE

Creates a new logical name table. The /NAME_TABLE keyword is required.

FORMAT

CREATE/NAME_TABLE *table-name*

Command Keywords	Defaults
/ATTRIBUTES[=(option[, . . .])]	
/EXECUTIVE_MODE	*/SUPERVISOR_MODE*
/[NO]LOG	*/LOG*
/PARENT_TABLE=table	*/PARENT_TABLE=LNM$PROCESS_DIRECTORY*
/PROTECTION=(code)	*/PROTECTION=(S:RWED,O:RWED,G,W)*
/QUOTA=bytes	*See text*
/SUPERVISOR_MODE	*/SUPERVISOR_MODE*
/USER_MODE	*/SUPERVISOR_MODE*

restrictions

- Restrictions apply to the use of the /EXECUTIVE_MODE keyword. See the keyword description for more information.
- You cannot use this command in an indirect command file to create a table with the process directory (LNM$PROCESS_DIRECTORY) as the parent (for example, a private table).

prompt

_Table name: table-name

command parameter

table-name

The name of the logical name table you are creating. This name is entered as a logical name in either the process directory logical name table (LNM$PROCESS_DIRECTORY) or the system directory logical name table (LNM$SYSTEM_DIRECTORY).

The table name can have from 1 through 255 characters. If the table name contains any characters other than alphanumerics, dollar signs, or underscores, enclose it in quotation marks.

DESCRIPTION

You can use the CREATE/NAME_TABLE command to create a new logical name table. Note that the minimum abbreviation for this command is CREATE/LOGI. If you enter CREATE/LOG, you will invoke the CREATE command with the /LOG keyword.

Use the /USER_MODE, /SUPERVISOR_MODE, or /EXECUTIVE_MODE keywords to specify the access mode for the table you are creating. If you specify more than one of these keywords, only the last one entered is accepted. If you do not specify an access mode, a supervisor mode table is created.

A table name is entered in the LNM$PROCESS_DIRECTORY directory table if the table is private. (A private table is available to your process only.) A table name is entered in the LNM$SYSTEM_DIRECTORY directory table if the table is shareable. (A shareable table is available to other users on the system.)

When you create a logical name table, you may specify a parent table. The parent table supplies a "descendant" with a size quota and certain attributes. You can specify a parent table with the same access mode as the table you are creating or with an inner mode. (User mode is the outermost mode; executive mode is the innermost mode.)

The location of the parent table determines which directory the descendant is entered in and, therefore, determines whether the descendant is private or shareable. If you do not specify a parent table, then the parent table is LNM$PROCESS_DIRECTORY.

To delete a logical name table, use the DEASSIGN command, specify the name of the table you want to delete, and use the /TABLE keyword to specify the directory table where the table was entered.

command keywords

/ATTRIBUTES[(option[, . . .])]

Specifies attributes for the logical name table. If you do not specify the /ATTRIBUTES keyword, no attributes are set.

You can specify the following options for attributes:

Option	Function
CONFINE	Indicates that the table is not to be copied into a subprocess by the SPAWN command. You can use this option only when you are creating a private logical name table. If a table is created with the CONFINE attribute, all names entered into the table are also confined.
NO_ALIAS	Indicates that no tables with the same name may be created in an outer access mode. If you do not specify NO_ALIAS, then the table may be "aliased" by an identical name created in an outer access mode. If you create a table with the NO_ALIAS attribute, and a table with the same name in the same or in an outer mode already exists, then that table is deleted.
CREATE_IF	Creates a new table only if there is no existing table with the same name, access mode, and directory table that you specify. If a name that meets these criteria exists, a new table is not created even if you specify a quota or protection that differs from the existing table's quota.

If you specify only one option, you can omit the parentheses. Only the attributes you specify are set.

/EXECUTIVE_MODE

Creates an executive mode logical name table.

You must have SYSNAM privilege to create a logical name table in executive mode.

/LOG
/NOLOG

Controls whether an informational message indicating the result of the command is generated when the SUPERCEDE attribute was specified, or when the table exists but the SUPERCEDE attribute was not specified. The default is /LOG.

/PARENT_TABLE=table

Specifies the name of the parent table. If you do not specify a parent table, the default table is LNM$PROCESS_DIRECTORY. In this case, LNM$PROCESS_DIRECTORY is both the parent table and the directory table for the table you are creating.

The parent table must have the same access mode as the table you are creating or an inner mode. The location of the parent table determines which directory the new table is created in.

/PROTECTION=(code)

Defines the protection to be applied to the table. By default, the system and owner have read (R), write (W), execute (E), and delete (D) access.

Protection applies only to shareable logical name tables; it does not apply to process-private logical names.

/QUOTA=bytes

Specifies the size limit of the table, in bytes. This quota is deducted from the parent table's quota.

If you do not specify the /QUOTA keyword, then the default is for the table to take bytes dynamically from the parent table's quota. This is called pooled quota.

/SUPERVISOR_MODE

Creates a supervisor mode logical name table. If you do not explicitly specify a mode, a supervisor mode logical name table is created.

/USER_MODE

Creates a user mode logical name table.

EXAMPLE

```
> CREATE/LOGICAL_NAME_TABLE TEST_TAB
> SHOW LOGICAL TEST_TAB
%SHOW-S-NOTRAN, no translation for logical name TEST_TAB
> SHOW LOGICAL/TABLE=LNM$PROCESS_DIRECTORY TEST_TAB
```

The CREATE/LOGICAL_NAME_TABLE command creates a new table called TEST_TAB. By default, the table is created in the process directory. The first SHOW LOGICAL command does not find the name TEST_TAB because it does not, by default, search the process directory table. You must use the /TABLE keyword to request that the process directory be searched.

DEALLOCATE

The DEALLOCATE command returns a device that was reserved for private use to the pool of available devices in the system.

The form of the DEALLOCATE command compatible with VAX–11 RSX and RSX–11 MCR is

```
DEA device-name [:]
```

FORMAT

DEALLOCATE *device-name[:]*

Command Keyword	**Default**
/ALL	*None*

restrictions

None.

prompt

_Device: device-name[:]

command parameter

device-name[:]

Specifies the name of the device to be deallocated. The device name can be a physical device name or a logical name.

If you omit the controller designator and/or unit number, they default to Controller A and Unit 0, respectively.

command keyword

/ALL

Requests that all devices you currently have allocated be deallocated.

If you specify /ALL, you cannot specify a device name.

EXAMPLES

1

```
> DEALLOCATE _DMB1:
```

The DEALLOCATE command deallocates Unit 1 of the RK06 device(s) on Controller B. The underscore character in the device name indicates that it is a physical device name; the DEALLOCATE command does not check to see if it is a logical name.

2

```
> ALLOCATE MT: TAPE:
%MCR-I-ALLOC, _MTB1:  allocated
    .
    .
    .
> DEALLOCATE TAPE
```

The ALLOCATE command requests that any magnetic tape drive be allocated and assigns the logical name TAPE to the device. The response to the ALLOCATE command indicates the successful allocation of the device MTB1. The DEALLOCATE command specifies the logical name TAPE to release the tape.

Note that a colon was specified on the logical name TAPE in the ALLOCATE command but that the colon can be omitted on the DEALLOCATE command.

3

```
> DEALLOCATE/ALL
```

The DEALLOCATE command deallocates all devices that are currently allocated.

DEASSIGN

The DEASSIGN command cancels one or more logical name assignments made with the ALLOCATE, ASN, DEFINE, or MOUNT command.

FORMAT

DEASSIGN *[logical-name[:]]*

Command Keywords	**Defaults**
/ALL	
/EXECUTIVE_MODE	*/SUPERVISOR_MODE*
/GROUP	*/PROCESS*
/JOB	*/PROCESS*
/PROCESS	*/PROCESS*
/SUPERVISOR_MODE	*/SUPERVISOR_MODE*
/SYSTEM	*/PROCESS*
/TABLE=table-name	*/TABLE=LNM$PROCESS*
/USER_MODE	*/SUPERVISOR_MODE*

restrictions

Restrictions apply to the following keywords:

/EXECUTIVE_MODE
/GROUP
/SYSTEM

See the keyword descriptions for more information.

prompt

_Log Name: logical-name[:]

command parameter

logical-name[:]

Specifies the logical name to be deassigned. Logical names can have from 1 to 63 characters. If the logical name contains any characters other than alphanumerics, dollar signs, or underscores, enclose it in quotation marks.

If you terminate the logical-name parameter with a colon, the command interpreter ignores the colon. (Note that the ASSIGN and ALLOCATE commands remove a trailing colon, if present, from a logical name before placing the name in a logical name table.) If a colon is present in the actual logical name, you must type two colons on the logical-name parameter for the DEASSIGN command, for example, DEASSIGN "SYS::TERM."

The logical-name parameter is required unless you use the /ALL keyword.

DESCRIPTION

The DEASSIGN command cancels a logical name assignment that was made with one of the following commands: ALLOCATE, ASSIGN, DEFINE, or MOUNT. You can use the /ALL keyword with DEASSIGN to cancel all logical names.

If you enter more than one of the keywords /PROCESS, /GROUP, or /SYSTEM, only the last one entered is accepted. If entries exist for the specified logical name in more than one logical name table, the name is deleted only from the last specified logical name table.

MCR deassigns all supervisor mode entries in the process logical name table when you log off the system. User mode entries are deassigned when any image exits. Names in the group or system logical name tables must be explicitly deassigned.

command keywords

/ALL

Specifies that all logical names in the specified logical name table are to be deleted. If no logical name table is specified, all process logical name table entries are deleted.

If you specify /ALL, you cannot enter a logical-name parameter.

/EXECUTIVE_MODE

Generally used when deleting entries that were created in executive mode in the process logical name table. However, if you specify the /EXECUTIVE_MODE keyword, the DEASSIGN command will delete supervisor and user mode entries as well.

You must have SYSNAM privilege to deassign executive mode logical names. If you specify /EXECUTIVE_MODE and you do not have SYSNAM privilege, the DEASSIGN command ignores the keyword and attempts to deassign a supervisor mode logical name.

/GROUP

Indicates that the specified logical name is in the group logical name table.

The user privilege GRPNAM is required to delete entries from the group logical name table.

/JOB

Indicates that the specified logical name is in the jobwide logical name table. If you do not explicitly specify a logical name table, the default is /PROCESS.

You cannot deassign jobwide logical name entries that were made by the system when you logged in. However, if you assign new equivalence names, you can deassign the names you explicitly created.

/PROCESS

Indicates that the specified logical name is in the process logical name table. The default is /PROCESS.

You cannot deassign logical name table entries that were made by MCR, for example, SYS$INPUT and SYS$OUTPUT. However, if you assign new equivalence names for these logical names, you can deassign the names you explicitly created.

/SUPERVISOR_MODE

Generally used when deleting entries that were created in supervisor mode in the process logical name table. However, if you specify the /SUPERVISOR_MODE keyword, the DEASSIGN command will delete user mode entries as well.

/SYSTEM

Indicates that the specified logical name is in the system logical name table.

The user privilege SYSNAM is required to delete entries from the system logical name table.

/TABLE=table-name

Indicates which logical name table the logical name is to be removed from.

The /TABLE keyword can also be used to delete a logical name table from a logical name directory table. For example, to delete a process-private table, specify the following:

```
> DEASSIGN/TABLE=LNM$PROCESS_DIRECTORY table-name
```

/USER_MODE

Generally used when deleting entries that were created in user mode in the process logical name table. If you specify the /USER_MODE keyword, the DEASSIGN command can only delete user mode entries.

EXAMPLES

1

```
> SHOW LOGICAL TEST_CASES
  O "TEST_CASES" = "XXX1:[HARVEY]FILES.DAT" (LNM$PROCESS_TABLE)
> DEASSIGN   TEST_CASES
> SHOW LOGICAL TEST_CASES
  SHOW-S-NOTRAN, No translation for logical name TEST_CASES
```

The SHOW LOGICAL command displays the current equivalence name for the logical name TEST_CASES. The DEASSIGN command deassigns the equivalence name; the next SHOW LOGICAL command indicates that the name has been deassigned.

2

```
> ASN XXX1:   COPY:
> DEASSIGN COPY
```

The ASN command equates the logical name COPY with the device XXX1 and places the names in the process logical name table. The DEASSIGN command deletes the logical name. Note that a colon was specified on the logical name COPY in the ASSIGN command, but it can be omitted in the DEASSIGN command.

3

```
> DEFINE SWITCH:  TEMP
> DEASSIGN SWITCH::
```

The DEFINE command places the logical name SWITCH: in the process logical name table. Two colons are required on the DEASSIGN command to delete this logical name because the DEFINE command does not remove trailing colons from logical names.

4
```
> DEASSIGN/ALL
```

The DEASSIGN command deletes all names from the process logical name table. This command does not, however, delete the names that were placed in the process logical name table in executive mode by MCR (SYS$INPUT, SYS$OUTPUT, SYS$ERROR, SYS$DISK, and SYS$COMMAND).

5
```
> ASN DALLAS::ZZZ0:  DATA
   .
   .
   .
> DEASSIGN DATA
```

The ASN command associates the logical name DATA with the device specification ZZZ0 on remote node DALLAS. Subsequent references to the logical name DATA result in references to the disk on the remote node. The DEASSIGN command cancels the logical name assignment. Notice that no node name is necessary with the DEASSIGN command because that information is contained in the logical name DATA.

DEBUG

The DEBUG command invokes a debugger after you have interrupted a program's execution by pressing CTRL/C or CTRL/Y. If the image is native, the VAX/VMS Symbolic Debugger prompts. If it is an RSX-11 task image that was built with ODT, ODT prints a T-bit exception message.

FORMAT

DEBUG

Command Keywords	**Defaults**
None	*None*

restrictions

- If no image is currently executing, the DEBUG command performs no operation.
- If the interrupted program was not linked with the appropriate debugging aid (VAX/VMS Symbolic Debugger for native mode images or ODT for RSX-11 task images), the DEBUG command causes a software exception condition.

  If the image is a native mode image that has not declared a condition handler or if it is an RSX-11 task image that has not specified a T-bit SST routine address, this exception may cause the image to terminate.
- The only privilege you must have to debug a file is Execute privilege on that file. Two restrictions, however, are that (1) you must not have linked the file you plan to debug with the Link qualifier /NOTRACE and (2) you can only debug code in user mode. You cannot debug any code in supervisor, executive, or kernel modes.

prompt

DBG>

DESCRIPTION

When a program image is executing, it can be interrupted by <CTRL/C> or <CTRL/Y>. Following the interruption, the DEBUG command can be issued to pass control to the debugger; this function is useful, for example, when a program is in an infinite loop and you want to gain control so you can use the debugger to determine the cause of the problem.

For details on the use of the debugger for native mode images, see the *VAX/VMS Symbolic Debugger Reference Manual.*

For details on the use of ODT for RSX-11 task images, see the *IAS/RSX-11 ODT Reference Manual.*

For details on condition handling and exception conditions, see the *VAX/VMS System Services Reference Manual.*

EXAMPLES

1

```
> RUN COPLEY/NODEBUG
CTRL/Y INTERRUPT
> DEBUG
DBG>
```

In this example, pressing CTRL/Y interrupts execution of a VAX–11 native mode image and causes MCR to prompt. Typing the DEBUG command at this point causes the VAX/VMS debugger to prompt.

2

```
> RUN PEALE/NODEBUG
CTRL/Y INTERRUPT
> DEBUG
ODT:FANGLE
```

In this example, pressing CTRL/Y interrupts execution of an RSX–11 task image and causes MCR to prompt. Typing the DEBUG command at this point causes ODT to prompt. FANGLE is the process name.

DEFINE

The DEFINE command creates a logical name table entry and assigns an equivalence name string to the specified logical name.

FORMAT

DEFINE *logical-name equivalence-name[, . . .]*

Command Keywords	**Defaults**
/EXECUTIVE_MODE	*/SUPERVISOR_MODE*
/GROUP	*/PROCESS*
/JOB	*/PROCESS*
/[NO]LOG	*/NOLOG*
/NAME_ATTRIBUTES[=(option[, . . .])]	
/PROCESS	*/PROCESS*
/SUPERVISOR_MODE	*/SUPERVISOR_MODE*
/SYSTEM	*/PROCESS*
/TABLE=name	*/TABLE=LNM$PROCESS*
/TRANSLATION_ATTRIBUTES[=(option[, . . .])]	
/USER_MODE	*/SUPERVISOR_MODE*

restrictions

- Restrictions apply to the following keywords:

  /EXECUTIVE_MODE
  /GROUP
  /SYSTEM

  See the keyword descriptions for more information.
- You must specify the first four letters of the command (DEFI) to resolve ambiguity with the RMS–11 utility DEF (specified as DEF).

prompts

_Log Name: logical-name
_Equ Name: equivalence-name[, . . .]

command parameters

logical-name

Specifies the logical name string. The logical name string can contain from 1 to 255 characters. If the string contains any characters other than alphanumerics, the dollar sign, or the underscore character, enclose the string in quotation marks. (The optional colon is not considered to be part of the string.)

equivalence-name[, . . .]

Defines the equivalence name to be associated with the logical name in the specified logical name table. The equivalence name string can contain from 1 to 255 characters. If the string contains any characters other than alphanumerics, the dollar sign, or the underscore character, enclose the string in quotation marks.

The DEFINE command allows you to assign the same logical name to more than one equivalence name. For example, you can use the same equivalence name to access different directories on different disks or different files in different directories.

The use of multiple equivalence names (see Example 5 for this command description) is referred to as the use of search lists. RSX–11 tasks using FCS-11 or direct ACP QIOs cannot use search lists. An attempt to use one results in a "bad device name" error. RSX–11 tasks using RMS–11 can use search lists in the same manner as native mode images.

DESCRIPTION

The DEFINE command creates an entry in the logical name table by defining a local name to stand for an equivalence name. An equivalence name can be a device name, another logical name, or a file specification.

If you enter more than one of the keywords from the group /PROCESS, /JOB, /GROUP, or /SYSTEM, or the keywords from the group /EXECUTIVE_MODE, /SUPERVISOR_MODE, or /USER_MODE, only the last keyword entered is accepted.

The ASSIGN command performs a similar function to the DEFINE command. Use the DEASSIGN command to cancel an entry in the logical name table, whether that entry was created by the ALLOCATE, ASSIGN, DEFINE, or MOUNT command.

Note: **Avoid assigning a logical name that matches the file name of an executable image in SYS$SYSTEM:. Such an assignment would prohibit you from invoking that image.**

command keywords

/EXECUTIVE_MODE

Specifies that the logical name in the process logical name table be entered in executive mode. If you specify the /EXECUTIVE_MODE keyword and you do not have SYSNAM privilege, the DEFINE command ignores the keyword and creates a supervisor mode logical name.

/GROUP

Places the logical name/equivalence name pair in the group logical name table. Other users who have the same group number in their User Identification Codes (UICs) can access the logical name.

The user privilege GRPNAM is required to place a name in the group logical name table.

/JOB

Places the logical name in the jobwide logical name table. This process and any of its descendant subprocesses (that is, in the same job tree) can access the logical name. The /JOB keyword is synonymous with /TABLE=LNM$JOB.

/LOG
/NOLOG

Controls whether the DEFINE command displays the logical name that was created.

/NAME_ATTRIBUTES[=(option[, . . .])]

Specifies attributes for a logical name. By default, no attributes are set. You can specify the following options for attributes:

CONFINE — The name is not to be copied into a subprocess by the SPAWN command. This option is meaningful only for when you create logical names in a private table. If you specify this option for a logical name in a shareable table, it is ignored.

The CONFINE attribute is also inherited from the logical name table where the name is entered; if the logical name table is "confined," then all names in the table are "confined."

NO_ALIAS — The logical name cannot be duplicated in this table with a less privileged (outer) access mode. If another logical name with the same name and an outer access mode already exists in this table, the name is deleted.

If you specify only one option, you can omit the parentheses. Only the attributes you specify are set.

/PROCESS

Places the logical name/equivalence name pair in the process logical name table. The default is /PROCESS.

/SUPERVISOR_MODE

Specifies that the logical name in the process logical name table be entered in supervisor mode.

The default for process logical name table entries is /SUPERVISOR_MODE. The /SUPERVISOR_MODE keyword is ignored when entries are made in the group or system logical name tables.

/SYSTEM

Places the logical name/equivalence name pair in the system logical name table. All system users can access the logical name.

The user privilege SYSNAM is required to place a name in the system logical name table.

/TABLE=name

Specifies the name of the logical name table that you want the DEFINE command to access.

/TRANSLATION_ATTRIBUTES[=(option[, . . .])]

Specifies one or more attributes that modify the logical name translation process. You can specify the following options for translation attributes:

Option	Function
CONCEALED	Indicates that the equivalence string is a concealed device name. When a concealed device name is defined, the system displays the logical name, rather than the equivalence string, in messages referring to the device. If you specified the CONCEALED attribute, then the equivalence string must be a physical device name.

Option	Function
TERMINAL	Indicates that the equivalence string should not be translated again: logical name translation should terminate with the current equivalence string.

If you specify only one option, you can omit the parentheses. Only the attributes you specify are set.

/USER_MODE

Specifies that the logical name in the process logical name table be entered in user mode.

A user mode logical name is practical for the execution of a single image because it allows an image executing in a command procedure to redefine SYS$INPUT. User mode entries are deleted when any image executing in the process exits (that is, after any DCL command or user program executing an image completes execution) or when a STOP command is issued.

By default, process logical name table entries are made in supervisor mode. The /USER_MODE keyword is ignored when entries are made in the group or system logical name tables.

EXAMPLES

1

```
> DEFINE PROCESS_NAME LIBRA
> RUN WAKE
```

The DEFINE command places the logical name PROCESS_NAME in the process logical name table with an equivalence name of LIBRA. The program WAKE translates the logical name PROCESS_NAME to perform some special action on the process named LIBRA.

2

```
> DEFINE TEMP: XXX1:
     .
     .
     .
> DEASSIGN TEMP::
```

The DEFINE command creates an equivalence name for the logical name TEMP: and places the name in the process logical name table. The DEASSIGN command deletes the logical name. Note that two colons are required on the logical name in the DEASSIGN command.

3

```
> DEFINE PORTLAND PRTLND::YYY0:[DECNET.DEMO.COM]
```

This DEFINE command places the logical name PORTLAND in the process logical name table with an equivalence name of PRTLND::YYY0: [DECNET.DEMO.COM].

4

```
> DEFINE MYDISK XXX0:[MYDIR] YYY0:[TESTDIR]
```

This DEFINE command places the logical name MYDISK in the process logical name table with two equivalence names: XXX0:[MYDIR] and YYY0:[TESTDIR]. Using the logical name MYDISK will result in access of either the MYDIR directory when the current disk is XXX0: or the TESTDIR directory when the current disk is YYY0:, depending on the file operation.

DELETE

The DELETE command deletes one or more files from a mass storage disk volume.

FORMAT

DELETE *file-spec[, . . .]*

Command Keywords	**Defaults**
/BACKUP	*/CREATED*
/BEFORE[=time]	*/BEFORE=TODAY*
/BY_OWNER[=uic]	
/[NO]CONFIRM	*/NOCONFIRM*
/CREATED	*/CREATED*
/[NO]ERASE	*/NOERASE*
/EXCLUDE=(filespec[, . . .])	
/EXPIRED	*/CREATED*
/[NO]LOG	*/NOLOG*
/MODIFIED	*/CREATED*
/SINCE[=time]	*/SINCE=TODAY*

restrictions

RSX-11 MCR does not support this command.

prompt

_File: file-spec[, . . .]

command parameter

file-spec[, . . .]

Specifies the name of each file to be deleted. If you specify more than one file specification, separate each with a comma (,) or a plus sign (+).

The first file specification must contain an explicit or default directory specification plus a file name, file type, and version number. (You can specify any of these fields as wildcard characters.) Subsequent file specifications can omit any of the fields except the version numbers; the first file specification provides defaults for additional file specifications.

A semicolon followed by no file version number, a version number of 0, or one or more blanks in the version number of a file specification results in the deletion of the latest version of the file.

Before you use negative version numbers, make sure you understand the differences between RSX-11 and VAX/VMS negative version numbers, which are described in Section 2.

DESCRIPTION

You specify /ENTRY to delete a batch or print queue entry. You can specify any of the remaining keywords to control the deletion of files. The /CREATED, /MODIFIED, and /EXPIRED keywords determine whether files are to be deleted based on their creation date, on their most recently modified date, or on their planned expiration date. The /BEFORE and /SINCE keywords are used with the selection date keywords to establish the time span to be considered by the DELETE command.

command keywords

/BACKUP

Selects files according to the dates of their most recent backup. This keyword is relevant only when used with the /BEFORE or /SINCE keywords. Use of the /BACKUP keyword is incompatible with /CREATED, /MODIFIED, and /EXPIRED. /CREATED is the default.

/BEFORE[=time]

Chooses only those specific files that have a selection date before the specified date. The selection date can be specified as an absolute date and time, a delta time, a combination absolute and delta time, TODAY, or YESTERDAY. Section 2 explains how to specify an absolute time or combination absolute and delta time.

If you do not specify /BEFORE, the command deletes all specified files regardless of date (unless /SINCE is specified; see the description of the keyword).

If you specify /BEFORE but do not supply a selection date, the DELETE command uses TODAY by default. That is, the command deletes all files created or modified yesterday or before.

/BY_OWNER[=uic]

Selects one or more files only if their owner User Identification Code (UIC) matches the specified owner UIC.

The UIC can be specified either as an eight-digit octal number or with the asterisk wildcard character.

If the /BY_OWNER keyword is specified without a UIC, the UIC of the current process is assumed.

/CONFIRM
/NOCONFIRM

Controls whether a request is issued before each individual DELETE operation to confirm that the operation should be performed on that file.

When the system issues the prompt, you can issue any of the following responses:

- Affirmative responses

  Yes
  True
  1

- Negative responses

  No
  False
  0
  <RET>

- Stop processing responses

  Quit
  <CTRL/Z>

- Continue processing without further prompts response

  ALL

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE). If you type a response other than one of those in the list, the prompt will be reissued.

/CREATED

Selects specific file(s) for deletion based on their creation date. This keyword is meaningful only when /BEFORE or /SINCE is specified. Use of this keyword is incompatible with /MODIFIED, /EXPIRED, and /BACKUP. /CREATED is the default.

The default selection date is both the created date and the most recently modified date.

/ERASE
/NOERASE

Requests that the disk blocks containing the specified file be zeroed as the file is deleted.

/EXCLUDE=(file-spec[, . . .])

Excludes any files that match the listed file specification(s). The file specification can contain a directory specification. If you specify two or more files, separate the file specifications with commas.

You can include wildcard characters in the file specifications. However, you cannot include either the device or directory field in the file specification(s) you supply with the /EXCLUDE keyword.

/EXPIRED

Selects specific files according to their planned expiration date. This keyword is meaningful only when /BEFORE or /SINCE is specified. Use of this keyword is incompatible with /CREATED, /MODIFIED, and /BACKUP.

The default selection date is both the created date and the most recently modified date.

/LOG
/NOLOG

Determines whether the file specification of each file is to be displayed when the file is deleted.

By default, the DELETE command does not display the names of files after it deletes them.

/MODIFIED

Selects specific files according to the most recent date on which they were modified. This keyword is meaningful only when /BEFORE or /SINCE is specified. Use of this keyword is incompatible with /CREATED, /EXPIRED, and /BACKUP. /CREATED is the default.

/SINCE[=time]

Chooses only those specific files that have a selection date after the specified date. The selection date can be specified as an absolute time, a delta time, a combination absolute and delta time, TODAY, or YESTERDAY. Section 2 explains how to specify an absolute time or a combination absolute and delta time.

If you do not specify /SINCE, the command deletes the specified files regardless of date (unless /BEFORE is specified; see the description of that keyword).

If you specify /SINCE but do not supply a selection date, the DELETE command uses TODAY by default; that is, the command deletes all specified files with a selection date of today.

EXAMPLE

```
> DELETE /SINCE=(11:30) INPUT.DAT;*
```

This command deletes all versions of the file INPUT.DAT created or modified after 11:30 today.

DELETE/ENTRY

The DELETE/ENTRY command deletes one or more entries from a print or batch job queue. The /ENTRY keyword is required.

FORMAT

DELETE/ENTRY=*(entry-number[, . . .])*
queue-name[:]

Command Keywords	**Defaults**
None	*None*

restrictions

If you have GROUP privilege, the job(s) to be deleted must have been queued by the current process or by any process in the same group as the current process. Otherwise, WORLD or OPER privilege is required to delete a job that is not your own or in your group.

prompts

None.

command parameters

entry-number[, . . .]
Specifies the entry number of the job to be deleted from the queue. The /ENTRY keyword requires at least one entry-number parameter to specify the job number(s) of one or more jobs to be deleted from a print or batch job queue. If you specify more than one entry number, separate the numbers with commas and enclose the list in parentheses.

queue-name[:]
Specifies the name of the queue in which the job(s) exist.

DESCRIPTION

The DELETE/ENTRY command deletes one or more jobs from a queue. If you specify more than one entry number, all jobs must be located in the same queue.

You can delete jobs that have not yet begun processing or files that are currently executing or printing.

EXAMPLES

1

```
> PRINT/HOLD  ALPHA.TXT
 Job ALPHA (queue SYS$BATCH, entry 110) holding
  .
  .
  .
> DELETE/ENTRY=110  SYS$PRINT
```

The PRINT command queues a copy of the file ALPHA.TXT in a hold status to defer its printing until a later time. The system displays the job number 110 and the name of the queue in which the file was entered. Later, the DELETE/ENTRY command requests that the entry be deleted from the queue SYS$PRINT.

2

```
> SUBMIT/HOLD/PARAMETERS=SCANLINE  DOFOR
Job DOFOR (queue SYS$BATCH, entry 203) holding
> SUBMIT/AFTER=18:00  WEATHER
Job (queue SYS$BATCH, entry 210) holding until 8-AUG-1985 18:00
  .
  .
  .
> DELETE/ENTRY=(203,210) SYS$BATCH
```

The SUBMIT commands spool the command procedures DOFOR.COM and WEATHER.COM for processing as batch jobs. DOFOR.COM is queued in a hold status; WEATHER.COM is queued for execution after 6:00 PM. Later, the DELETE/ENTRY command requests that both these entries be deleted from the queue SYS$BATCH.

DEPOSIT

The DEPOSIT command replaces the contents of a specific location in virtual memory.

FORMAT

DEPOSIT *location=data[, . . .]*

Command Keywords	**Defaults**
/ASCII	*None*
/BYTE	
/DECIMAL	
/HEXADECIMAL	
/LONGWORD	
/OCTAL	
/WORD	

restrictions

- To change the contents of a specific location in virtual memory, you must have user mode read and write access to that location in virtual memory.
- RSX-11 MCR does not support the DEPOSIT command.

prompts

None.

command parameters

location

The virtual address of a location whose contents are to be changed. The specified location must be within the virtual address space of the task image currently running in the process and it must be read/write for user access mode.

The DEPOSIT and EXAMINE commands set a pointer to the byte following the last byte modified. You can refer to this pointer as "." for the first location in a subsequent EXAMINE command or as the deposit location in a subsequent DEPOSIT command.

data[, . . .]

Data to be deposited into the specified location. If you specify a list of data, the data is deposited into consecutive locations beginning at the specified location. By default, data is assumed to be in hexadecimal format.

The DEPOSIT command converts the specified data to binary and writes it into the specified virtual memory location.

DESCRIPTION

The DEPOSIT command converts the specified value to binary before placing it in virtual memory.

The initial default radix for the DEPOSIT command is hexadecimal. The default applies to both the interpretation of numeric literals in the command line and to the data. If you use a radix keyword to modify the command, that radix becomes the default for subsequent DEPOSIT and EXAMINE commands.

The initial default length unit for the DEPOSIT command is a longword. The EXAMINE command displays data one longword at a time with blanks between the longwords. If you use a /BYTE or /WORD keyword to modify the command, that unit becomes the default for subsequent DEPOSIT and EXAMINE commands.

RSX-11 task images start at virtual address 0 under VAX-11 RSX. This is the same starting address that they have under RSX-11.

If the address specified can be read but not written, the current contents of the location are not altered but are displayed. If the address specified cannot be read or written, asterisks (****) are displayed.

When used with the EXAMINE command, the DEPOSIT command can aid in the interactive debugging of programs. It is not necessary to taskbuild with a debugger to use these commands. Also, the combined use of the EXAMINE and DEPOSIT commands provides a functionality that is similar to the RSX-11 MCR command OPEN.

command keywords

/ASCII

Indicates that the data specified is in ASCII format.

When ASCII is specified or is the default, hexadecimal is the default radix for numeric literals. The location specified is interpreted as a hexadecimal address.

/BYTE

Specifies that data is to be deposited one byte at a time. If the data specified is longer than a byte, an error message is displayed.

/DECIMAL

Indicates that the default radix for numeric literals and displayed values is decimal.

/HEXADECIMAL

Indicates that the default radix for numeric literals and displayed values is hexadecimal. This is the initial default of the command; refer to the /DECIMAL description.

/LONGWORD

Specifies that data is to be deposited one longword at a time. This is the initial default of the command; refer to the /HEXADECIMAL description.

/OCTAL

Indicates that the default radix for numeric literals and displayed values is octal. The DEPOSIT command converts the data to its binary equivalent before placing it in virtual memory.

/WORD

Specifies that data is to be deposited one word at a time.

EXAMPLE

```
> RUN   RSXPROG
      .
      .
      .
 CTRL/Y INTERRUPT
> EXAMINE/OCT/WORD  2332
  000004DA:      007402
> DEPOSIT . = 777
  000004DA:      000777
> CONTINUE
```

Pressing CTRL/Y interrupts execution of the RSX–11 task image RSXPROG. The EXAMINE command establishes the radix as octal and the length unit as a word and displays the content of location 4DA (16). The DEPOSIT command then deposits the octal value 777 in location 4DA.

DIFFERENCES

The DIFFERENCES command compares the contents of two disk files and creates a listing of those records that do not match.

FORMAT

DIFFERENCES *master-file-spec [revision-file-spec]*

Command Keywords	**Defaults**
/COMMENT_DELIMITER[=(delimiter[, . . .])]	
/IGNORE=(option[, . . .])	
/MATCH=size	*/MATCH=3*
/MAXIMUM_DIFFERENCES=n	
/MERGED[=n]	*/MERGED=1*
/MODE=(radix[, . . .])	*/MODE=(ASCII)*
/[NO]NUMBER	*/NUMBER*
/OUTPUT[=file-spec]	*See text*
/PARALLEL[=n]	
/SEPARATED[=(input-file[, . . .])]	*See text*
/SLP	
/WIDTH=n	*See text*
/WINDOW=size	*See text*

Positional Keyword	**Default**
/CHANGE_BAR[=(format[, . . .])]	*See text*

restrictions

Restrictions apply to the use of the keyword /SLP. See the keyword description for more information.

prompts

_File 1:	master-file-spec
_File 2:	revision-file-spec

command parameters

master-file-spec

Specifies the name of the primary input file to be compared.

The file specification must include a file name and a file type. No wildcard characters are allowed in the file specification.

revision-file-spec

Specifies the name of the secondary input file to be compared. Any nonspecified fields default to the corresponding fields of the primary input file specification.

If you do not specify a secondary input file, the DIFFERENCES command uses the next lower version of the primary input file.

No wildcard characters are allowed in the file specification.

DESCRIPTION

Use the DIFFERENCES command to find out whether two files are identical and, if not, how they differ. DIFFERENCES compares the two specified files on a record-by-record basis and produces an output file that lists the differences, if any.

The keywords for the DIFFERENCES command can be categorized according to their functions, as follows:

- Keywords that request DIFFERENCES to ignore data in each record:

  /COMMENT_DELIMITERS
  /IGNORE

  These keywords allow you to define characters that denote comments or to designate characters or classes of characters to ignore when comparing files—for example, extra blank lines or extra spaces within lines.

  By default, DIFFERENCES compares every character in each record.

- Keywords that control the format of the information contained in the list of differences:

  /CHANGE_BAR
  /IGNORE
  /MERGED
  /MODE
  /PARALLEL
  /SEPARATED
  /SLP
  /WIDTH

  By default, DIFFERENCES merges differences it finds in the files being compared. It lists each record in the file that has no match in the other input file and then the first record that it finds that does have a match.

  By default, DIFFERENCES supplies a line number with each listed record. It lists records in the output file with all ignored characters deleted.

  You can specify combinations of keywords to request an output listing from DIFFERENCES that includes the comparison in more than one format. Note that SLP output is incompatible with all other types of output and PARALLEL output can only be generated in ASCII mode.

- Keywords that control the extent of the comparison:

  /MATCH
  /MAXIMUM_DIFFERENCES
  /WINDOW

  By default, DIFFERENCES reads every record in the master input file and looks for a master record in the revision input file. A search for a match between the two input files continues until either a match is found or the ends of the two files are reached. Sections of the two files are considered a match only if three sequential records are found to be identical in each file.

  By default, DIFFERENCES output is written to the current SYS$OUTPUT device. Use the /OUTPUT keyword to request DIFFERENCES to write the output to an alternate file or device.

DIFFERENCES terminates with an exit status. The following severity levels indicate the result of the execution:

Severity Level	Result
SUCCESS	Files are identical
INFORMATIONAL	Files are different
WARNING	User-specified maximum number of differences has been exceeded
ERROR	Insufficient virtual memory to complete comparison

All non-SUCCESS severities indicate that the two input files are different.

command keywords

/COMMENT_DELIMITER[=(delimiter[, . . .])]

Specifies one or more comment delimiters. This keyword is used with or without the /IGNORE=COMMENTS keyword to indicate which comments are to be ignored.

You can specify a comment delimiter either by listing the character itself or by using one of the following options:

Option	Character
COLON	Colon (:)
COMMA	Comma (,)
EXCLAMATION	Exclamation point (!)
FORM_FEED	Form feed
LEFT	Left bracket ([)
RIGHT	Right bracket (])
SEMI_COLON	Semicolon (;)
SLASH	Slash (/)
SPACE	Space
TAB	Horizontal tab

You can specify up to 32 comment characters. If you specify two or more characters or options, separate them with commas and enclose the list in parentheses.

When you specify an option, you must type at least the first two letters of the option because single letters are treated as delimiters. Lowercase letters are automatically changed to uppercase unless they are enclosed in quotation marks. If both the uppercase and lowercase forms of a letter are to be used as delimiters, the letter must be specified twice, once in uppercase and once in lowercase. Special characters (for example, "!" and ",") must be enclosed in quotation marks.

If you do not include either a comment character or an option with the /COMMAND_DELIMITER keyword, DIFFERENCES assumes a default comment character based on the file type. For some file types (COB and

FOR), the default comment characters are considered valid delimiters only if they appear in the first column of a line.

File Type	Default Comment Character
B2S, B32, BAS, BLI	!
CBL, CMD	! and ;
COB	* or / in first column
COM, COR	!
FOR	! anywhere; C,D,c,d in first column
HLP	!
MAC, MAR	;
R32, REQ	!

Multicharacter comment delimiters are not allowed.

/IGNORE=(option[, . . .])

Specifies that one or more special options, strings, or records be ignored during the comparison. /IGNORE also controls whether the comparison records are output to the listing file exactly as they appeared in the input file or as edited records. You can request DIFFERENCES to ignore the following:

Option	Function
BLANK_LINES	Blank lines between data lines.
COMMENTS	Data that follows a comment delimiter. (Use the /COMMENT_DELIMITER keyword to designate one or more nondefault comment delimiters.)
FORM_FEEDS	Form-feed characters.
HEADER[=n]	First n records beginning with a record whose first character is a form feed. (The first record is not ignored if the only character it contains is a form feed.) The default value of n is 2.
TRAILING_SPACES	Space and tab characters at the end of a line of data.
SPACING	Multiple spaces or tabs within lines of data. (Each set of contiguous spaces or tabs is replaced by a single space.)

Each data line is checked for COMMENTS, FORM_FEEDS, HEADER, and SPACING before it is tested for TRAILING_SPACES and BLANK_LINES. Therefore, if you direct DIFFERENCES to ignore COMMENTS, TRAILING_SPACES, and BLANK_LINES, it will totally ignore a record that contains several spaces or blank lines followed by a comment.

By default, the DIFFERENCES command compares every character in each file and reports all differences. Also by default, DIFFERENCES lists records in the output file with all ignored characters deleted.

Although output records can contain any characters that appear in the input files, you can choose to have output records formatted as follows:

Character	Formatted Output
Tab	1-8 spaces
RET	<CR>
Line feed	<LF>
Vertical Tab	<VT>
Form feed	<FF>
Other nonprinting characters	(period)

If you specify /PARALLEL, output formatting is always performed.

You can choose how you want the records to appear by specifying one of the following options with the /IGNORE keyword:

Option	Function
EDITED	Output records are listed with ignored strings deleted
EXACT	Output records are listed exactly as they appear in the input file
PRETTY	Output records are formatted

/MATCH=size

Specifies the number of records that constitutes a match.

By default, when DIFFERENCES finds unmatched records, it assumes that the files once again match if it finds three sequential records that match. Use the /MATCH keyword to override the default match size of 3.

You can increase the /MATCH value if you feel that DIFFERENCES is incorrectly matching sections of the master and revision input files after it has detected a difference.

/MAXIMUM_DIFFERENCES=n

Specifies that DIFFERENCES is to terminate after a specified number of unmatched records has been found.

The number of unmatched records is determined by finding the maximum number of difference records for each difference section and adding them together.

If DIFFERENCES reaches the maximum number of differences that you have specified, it will output only those records that were detected before the maximum was reached. Also, it will output at most one listing format and return a warning message.

By default, there is no maximum number of differences. All records in the specified input files are compared.

/MERGED[=n]

Requests that the output file contain a merged list of differences with at least one matched record listed after each group of unmatched records. The value n is a decimal number less than or equal to the size of a match (see /MATCH=size). It indicates the number of matched records to list after each list of unmatched records. The default value of n is 1.

By default, DIFFERENCES produces a merged listing with one matched record listed after each set of unmatched records.

Use the /MERGED keyword to override the default value of n or to include a merged listing with other types of output.

/MODE=(radix[, . . .])

Specifies the format of the output listing. You can request that the output be formatted in one or more radix modes by specifying the following options:

ASCII
HEXADECIMAL
OCTAL

Enclose the option(s) in parentheses. You can truncate any of these options to one or more characters.

By default, DIFFERENCES writes the output file in ASCII. If you specify more than one radix, the output listing contains the file comparison in each specified radix. When you specify two or more radix modes, separate them with commas.

If you specify the /PARALLEL or /SLP keyword, the /MODE keyword is ignored for that listing form.

/NUMBER
/NONUMBER

Controls whether line numbers are to be generated for records in the listing. By default, line numbers are included in the listing.

/OUTPUT[=file-spec]

Defines an output file to receive the output difference list. If you omit the /OUTPUT keyword, the outpuf is written to the current SYS$OUTPUT device. If you use the /OUTPUT keyword without a file specification, the output is directed to a file with the same name as the master file with the file type DIF.

When you specify /OUTPUT, you can control the defaults applied to the output file specification. The default output file type is DIF.

No wildcard characters are allowed in the file specification.

/PARALLEL[=n]

Requests that the output file contain a parallel list of differences. The value n is a decimal number less than or equal to the size of the match (see /MATCH=n). This number indicates the number of matched records to output after each list of unmatched records.

By default, DIFFERENCES does not list records after each list of unmatched records. Also by default, DIFFERENCES creates only a list of merged differences.

/SEPARATED[=(input-file[, . . .])]

Requests that the output file contain sequential lists of unmatched records from the specified input files. You can designate the desired input files by specifying the following options:

MASTER
REVISION

If you specify both input files, separate the names or options with a comma and enclose the list in parentheses.

If you specify the /SEPARATED keyword without designating any input files, DIFFERENCES will generate a separate listing for each input file.

By default, DIFFERENCES creates only a merged list of differences.

/SLP

Requests that DIFFERENCES produce an output file suitable for input to the SLP editor. If you use the /SLP keyword, you cannot specify any of the following output file keywords: /MERGED, /PARALLEL, /SEPARATED, /CHANGE_BAR, /MODE=hexadecimal, /MODE=octal, /LINE_WIDTH, /IGNORE=pretty, /IGNORE=edited, or /NUMBER.

Use the output file produced by the /SLP keyword as input to SLP to update the master input file, that is, to make the master input file match the revision input file.

When you specify /SLP and you do not specify /OUTPUT, DIFFERENCES writes the output file to a file with the same file name as the master input file with the file type DIF.

/WIDTH=n

Specifies the width of lines in the output listing.

By default, output is 132 characters wide unless it is directed to the terminal. In that case, the output line width is controlled by the terminal line width. Use the SET TERMINAL command to change the terminal line width.

/WINDOW=size

Controls the number of records to search before listing a record as unmatched and continuing with the next record in the master input file.

The window size is the minimum size of a differences section that will cause DIFFERENCES to lose synchronization between the two input files.

By default, DIFFERENCES searches to the ends of both input files before listing a record as unmatched.

positional keyword

/CHANGE_BAR[=(format[, . . .])]

Requests that the output contain a listing of the associated file(s) with a change bar character next to the lines in the file that do not match. You can specify any of the following format options:

Option	Function
c	A change bar character
NUMBER	Line numbers
NONUMBER	No line numbers

In ASCII output mode, the change bar character parameter specifies a one-character code that will appear in the left margin next to records that do not match. By default, an exclamation point is used as the change bar character.

In HEXADECIMAL and OCTAL output mode, the change bar character is ignored. Instead, the string "***CHANGE***" appears in the record header.

You can control whether the change bar listing includes line numbers by specifying either NUMBER or NONUMBER. If neither is specified, the default is controlled by the /[NO]NUMBER command keyword.

To specify both a change bar character and either NUMBER or NONUMBER, separate the options with a comma and enclose the list in parentheses, for example, /CHANGE_BAR=($,NUMBER).

EXAMPLES

1

```
> DIFFERENCES EXAMPLE.TXT
************
File XX1:[GEORGE.TEXT]EXAMPLE.TXT;2
    1  DEMONSTRATION
    2  OF V4.0 DIFFERENCES
    3  UTILITY
******
File XX1:[GEORGE.TEXT]EXAMPLE.TXT;1
    1  DEMONSTRATION
    2  OF VMS DIFFERENCES
    3  UTILITY
************

Number of difference sections found: 1
Number of difference records found: 2

DIFFERENCES /IGNORE=()/MERGED=1-
    DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2
    DISK1:[GEORGE.TEXT]EXAMPLE.TXT;1
```

The DIFFERENCES command compares the contents of the two most recent versions of the file EXAMPLE.TXT in the current default directory. DIFFERENCES compares every character in every record and displays the results at the terminal.

2

```
> DIFFERENCES/PARALLEL/WIDTH=80/COMMENT_DELIMITER="V" EXAMPLE.TXT
-------------------------------------------------------------------------------
File DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2  |  File DISK1:[GEORGE.TEXT]EXAMPLE.TXT;1
------------------- 1 ------------------------------------ 1 ------------------
DEMONSTRATION                       |  <LF>DEMONSTRATION
-------------------------------------------------------------------------------

Number of difference sections found: 1
Number of difference records found: 1

DIFFERENCES/IGNORE=(COMMENTS)/COMMENT_DELIMITER=("V")/WIDTH=80/PARALLEL-
    DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2-
    DISK1:[GEORGE.TEXT]EXAMPLE.TXT;1
```

The DIFFERENCES command compares the same files as in Example 1, but ignores all comments following the first "V" encountered by DIFFERENCES. The command also specifies that an 80-column parallel list of differences be displayed.

3

```
> DIFFERENCES/WIDTH=80/MODE=(HEX,ASCII) EXAMPLE.TXT/CHANGE_BAR
************
File DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2
    1 ! DEMONSTRATION
    2 ! OF V4.0 DIFFERENCES
    3   UTILITY
************

************
File DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2
RECORD NUMBER 1 (00000001) LENGTH 14 (0000000E)  ***CHANGE***

     204E 4F495441 5254534E 4F4D4544 DEMONSTRATION .. 000000
RECORD NUMBER 2 (00000002) LENGTH 19 (00000013)  ***CHANGE***

 4E455245 46464944 20302E33 5620464F OF V3.0 DIFFEREN 000000
                              534543 CES............. 000010
RECORD NUMBER 3 (00000003) LENGTH 7 (00000007)

                     595449 4C495455 UTILITY......... 000000
************

Number of difference sections found: 1
Number of difference records found: 2

DIFFERENCES /WIDTH=80/MODE=(HEX,ASCII)
      DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2/CHANGE_BAR-
      DISK1:[GEORGE.TEXT]EXAMPLE.TXT;1
```

The DIFFERENCES command compares the same files as in Example 1, but lists the differences in both hexadecimal and ASCII formats. The command also specifies that default change bars be used in the output. The default change bar character for the hexadecimal output is ***CHANGE***. For the ASCII output, the default change bar character is the exclamation point.

4

```
> DIFFERENCES BOSTON::DISK2:TEST.DAT OMAHA::DISK1:[PGM]TEST.DAT
```

The DIFFERENCES command compares two remote files and displays any differences found. The first file is TEST.DAT on remote node BOSTON. The second file, also named TEST.DAT, is on remote node OMAHA.

DIRECTORY

The DIRECTORY command provides a list of files or information about a file or group of files.

FORMAT

DIRECTORY *[file-spec[, . . .]]*

Command Keywords	**Defaults**
/ACL	
/BACKUP	*/CREATED*
/BEFORE[=time]	*/BEFORE=TODAY*
/BRIEF	*/BRIEF*
/BY_OWNER[=uic]	
/COLUMNS=n	*/COLUMNS=4*
/CREATED	*/CREATED*
/[NO]DATE[=option]	*/NODATE*
/EXCLUDE=(file-spec[, . . .])	
/EXPIRED	*/CREATED*
/FILE_ID	
/FULL	
/GRAND_TOTAL	
/[NO]HEADING	*/HEADING*
/MODIFIED	*/CREATED*
/OUTPUT=file-spec	*/OUTPUT=SYS$OUTPUT*
/[NO]OWNER	*/NOOWNER*
/PRINTER	*See text*
/[NO]PROTECTION	*/NOPROTECTION*
/SECURITY	
/SELECT=(option[, . . .])	*See text*
/SINCE[=time]	*/SINCE=TODAY*
/[NO]SIZE[=option]	*/NOSIZE*
/TOTAL	*/BRIEF*
/[NO]TRAILING	*/TRAILING*
/VERSIONS=n	
/WIDTH=(option[, . . .])	*See text*

restrictions

RSX-11 MCR does not support the DIRECTORY command.

prompts

None.

command parameter

file-spec[, . . .]

Specifies one or more files to be listed. The syntax of a file specification determines what file(s) will be listed, as follows:

- If you do not enter a file specification, the DIRECTORY command lists all versions of the files in your current default directory.
- If you specify only a device name, the DIRECTORY command uses your default directory specification.

- Whenever the file specification does not include a file name and file type, all versions of all files in the specified directory are listed.
- If a file specification contains a file name and/or file type and no version number, the DIRECTORY command lists all versions.
- If a file specification contains only a file name, the DIRECTORY command assumes all file types and versions.

  If you specify more than one file, separate the file specifications with either commas (,) or plus signs (+). You can use wildcard characters for the file name, file type, or version number fields of a file specification to list all files that satisfy the components you specify.

DESCRIPTION

The output of the DIRECTORY command depends on certain formatting keywords and their defaults. These keywords are: /COLUMNS, /DATE, /FULL, /OWNER, /PROTECTION, and /SIZE. However, Files–11 Structure Level 2 files usually are listed in alphabetical and then alphanumeric order, with the highest-numbered versions first. Files–11 Structure Level 1 files are listed randomly. The page width is adjusted automatically to the number of columns requested.

In studying the keywords and the capabilities they offer, watch for keywords that work together as well as keywords that override other keywords. For example, if you want the full format, you cannot expect that much information in one column. Thus, if you specify both /COLUMNS and /FULL, the number of columns you request is ignored.

If PIP format directories are desired, you can use PIP instead of the DIRECTORY command.

command keywords

/ACL

Controls whether the Access Control List (ACL) for each file is displayed. By default, DIRECTORY does not display the ACL for each file. If you specify both the /ACL and the /COLUMNS keywords, the /COLUMNS keyword is ignored.

/BACKUP

Selects files according to the dates of their most recent backup. This keyword is relevant only when used with /BEFORE or /SINCE. Use of /BACKUP is incompatible with /CREATED, /MODIFIED, and /EXPIRED. /CREATED is the default.

/BEFORE[=time]

Specifies that only those files dated earlier than a particular time be printed. You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values.

This keyword is normally used in conjunction with one of the following keywords: /CREATED, /EXPIRED, or /MODIFIED. If you omit the /BEFORE keyword, you obtain all the files created, regardless of date. However, if you specify /BEFORE without a date or time, the default provides the files created prior to today.

/BRIEF

Includes only the file name, type, and version number of each file to be listed. The default output format is /BRIEF. However, the /BRIEF keyword is overridden, whether specified explicitly or by default, whenever any of the following formatting keywords are specified in the command: /SIZE, /DATE, /OWNER, /PROTECTION, or /FULL.

The brief format lists the Files–11 Structure Level 2 files in alphabetical order from left to right on each line, in descending version number order. Files–11 Structure Level 1 files are listed randomly.

/BY_OWNER[=uic]

Selects one or more files only if their owner User Identification Code (UIC) matches the specified owner UIC.

The UIC can be specified either as an eight-digit octal number or with the asterisk wildcard character.

If /BY_OWNER is specified without a UIC, the UIC of the current process is assumed.

/COLUMNS=n

Lists the files using the specified number of columns on each line of the display. This keyword is used in conjunction with the /BRIEF keyword (either explicitly or by default). By default, the number of columns in the brief format is four; however, you may request the brief format with as many columns as you desire. When other formatting keywords are specified in the command, they override the /COLUMNS keyword.

/CREATED

Selects the files according to their date of creation. This keyword is relevant only when used with the /BEFORE or /SINCE keywords, and should not be used with the /EXPIRED or /MODIFIED keywords. By default, when files are selected according to a particular date and time, the creation date is always used.

/DATE[=option]
/NODATE

Includes the creation, expiration, or date last written for each file listed. If you omit this keyword, the default is /NODATE. However, if you specify /DATE without an option, the creation date is provided.

You may specify one of the following options with the /DATE keyword:

Option	Function
ALL	Lists all three file dates in order, left to right: CREATED, MODIFIED, and EXPIRED
BACKUP	Lists the date of the last backup with each file
CREATED	Lists the creation date with each file
EXPIRED	Lists the expiration date with each file
MODIFIED	Lists the last date the file was written

/EXCLUDE=(file-spec[, . . .])

Excludes the listed file specification(s) from the directory search. Wildcard characters may be used in the file specification(s). At least one file specification is required for this keyword, but the file specification must not include a device or directory specification. Multiple file specifications are separated by commas and are enclosed in parentheses.

/EXPIRED

Selects files according to the planned expiration date for each file. This keyword is relevant only with the /BEFORE or /SINCE keywords, and should not be used with the /CREATED or /MODIFIED keywords.

/FILE_ID

Includes the file identification number for each specified file. The default is /NOFILE_ID, except in the /FULL display.

/FULL

Lists the following items for each file:

- File name
- File type
- Version number
- Number of blocks used
- Number of blocks allocated
- Date of creation
- Date of last backup
- Date last modified
- Date of expiration
- File owner's UIC
- File protection
- File identification number (FID)
- File organization
- Other file attributes
- Record attributes
- Record format

The /FULL keyword overrides the default brief listing format.

/GRAND_TOTAL

Inhibits the listing of all individual file information and all trailing line information, except for a one-line summary of the number of specified directories and files.

If you specify the /SIZE keyword, the grand total summary will include the total number of blocks allocated by all specified files.

/HEADING
/NOHEADING

Controls whether heading lines consisting of a device description and directory specification are printed. The default output format provides this heading.

/MODIFIED

Selects files according to the last date the file was modified. This keyword is relevant only with the /BEFORE or /SINCE keywords, and should not be used with the /CREATED or /EXPIRED keywords.

/OUTPUT=file-spec

Requests that the DIRECTORY command output be written to the file specified rather than to the current SYS$OUTPUT device. If you specify the /OUTPUT keyword without a file specification, the output is directed to SYS$OUTPUT. If you omit the file type in the file specification, the default file type is LIS.

Wildcard characters are not allowed in the file specification.

/OWNER
/NOOWNER

Controls whether the owner of the file is listed. By default, the owner is not listed.

/PRINTER

Queues the command output for printing under the name given by the /OUTPUT keyword. If you specify /PRINTER without the /OUTPUT keyword, the output is directed to a file named DIRECTORY.LIS, which is spooled for printing automatically and then deleted.

/PROTECTION
/NOPROTECTION

Controls whether the file protection for each file is listed. The default is /NOPROTECTION, which does not list the file protection.

/SECURITY

Controls whether information about file security is displayed. The /SECURITY keyword provides the same information as the three keywords /ACL, /OWNER, and /PROTECTION together.

/SELECT

Allows you to control the files selected based on the /SELECT options. If you specify only one option, you can omit the parentheses. The options are:

Option	Function
MAXIMUM=n	Selects files that have fewer blocks than the specified parameter. The default is 1073741823.
MINIMUM=n	Selects files that have more blocks than the specified parameter. The default is 0.

By default, file selection is based on other criteria.

/SINCE[=time]

Specifies that only those files dated after a specified time be printed. Observe the syntax rules for date and time values specified in Section 2.

This keyword is normally used in conjunction with one of the following keywords: /CREATED, /EXPIRED, or /MODIFIED. If you omit the /SINCE keyword, you will obtain all the files created, regardless of date. However, if you specify /SINCE without a time or date, you will obtain all files created since today began.

/SIZE[=option]
/NOSIZE

Provides the file size in blocks used and/or allocated for each file listed, according to the option you specify. If you omit this keyword, the default is /NOSIZE. However, if you specify /SIZE without an option, the listing provides (by default) the file size in blocks used. The options you can specify are the following:

Option	Function
ALL	Lists both the file size in blocks used and allocated
ALLOCATION	Lists the file size in blocks allocated
USED	Lists the file size in blocks used

/TOTAL

Inhibits the listing of all individual file information and prints only the trailing lines as described under the /TRAILING keyword.

By default, the output format is /BRIEF, which gives this total, but also lists all the file names, file types, and their version numbers.

/TRAILING
/NOTRAILING

Controls whether trailing lines that summarize the following information are output:

- Number of files listed
- Total number of blocks used per directory
- Total number of blocks allocated
- Total number of directories and total blocks used and/or allocated in all directories (only if more than one directory is listed)

By default, the output format includes most of this summary information. The /SIZE and /FULL keywords determine more precisely what summary information is included. If you omit /SIZE or /FULL, only the number of files is printed and possibly the total number of directories, if applicable. If you specify /SIZE, the number of blocks is also printed, according to the size option selected (used and/or allocated). If you specify /FULL, the number of files and the number of blocks used and allocated are all given.

/VERSION=n

Causes the latest n versions of each of the files selected to be listed. If you omit the /VERSION keyword, by default the listing includes all versions of each file.

/WIDTH=(option[, . . .])

Allows you to change the format of the directory display for four elements: file name field width, entire display width, owner field width, file size field width. If you specify only one option, you can omit the parentheses.

Option	Function
DISPLAY=n	Determines the total width for the DIRECTORY display. The value of n can range from 1 through 255. The default for n is 0, which means that DIRECTORY sets its display width to the terminal width. If the total width of the display exceeds the terminal width, the information will be truncated.
FILENAME=n	Determines the width of the file name field. The default value for n is 19. If you have requested another piece of information to be displayed along with the file name in each column, file names that exceed the n parameter will cause the line to wrap, after the file name field. (See /COLUMNS.)
OWNER=n	Determines the width of the owner field. The default value for n is 20. If the owner's UIC exceeds the length of the owner field, the information will be truncated.
SIZE=n	Determines the width of the size field. The default value for n is 6. If file size exceeds the length of the size field, the information will be truncated.

EXAMPLES

1
```
> DIRECTORY
```

The DIRECTORY command lists all versions of all files in the current default disk and directory in the brief format. The heading identifies the disk and directory, and the trailing line gives the total number of files.

2
```
> DIRECTORY/VERSIONS=1/COLUMN=1 AVERAGE.*
```

The DIRECTORY command lists only the highest versions of all files named AVERAGE in the current default directory. The format is brief; it is restricted to just one column. Heading and trailing lines are provided.

3
```
> DIRECTORY BLOCK%%%
```

The DIRECTORY command locates all versions and types of files in the default device and directory whose names begin with the letters BLOCK and end with any three additional characters. The output format is brief; it has four columns, with heading and trailing lines.

4
```
> DIRECTORY/TOTAL/SIZE=ALL
```

The DIRECTORY command outputs only a header and a trailing line that identify the total number of files and the blocks used and allocated for all versions of all files in the default disk and directory.

5

```
> DIRECTORY/EXCLUDE=(AVERAGE.DAT;*,AVERAGE.EXE;*) [* . . . ]AVERAGE
```

The DIRECTORY command locates all versions and types of files named AVERAGE in all directories and subdirectories on the default disk. From this list, all versions of all files named AVERAGE.DAT and AVERAGE.EXE are excluded prior to listing and totalling.

6

```
> DIRECTORY-
>_/MODIFIED/SINCE=09-DEC-1984:01:30/SIZE=ALL/OWNER-
>_/PROTECTION/OUTPUT=UPDATE/PRINTER [A*]
```

The DIRECTORY command locates all files that have been modified since 1:30 AM on December 9, 1984, and that reside on the default disk in directories whose names begin with the letter A. It formats the output to include all versions, the size used and allocated, the date last modified, the owner, and the protection codes. The output is directed to a file named UPDATE.LIS, which is spooled automatically and deleted when done.

DMO

The DMO command releases a volume that was previously specified in a MOUNT command. It functions identically to the DCL command DISMOUNT.

FORMAT

DMO *device-name[:]*

Command Keywords	**Defaults**
/ABORT	
/CLUSTER	
/UNIT	*See text*
/[NO]UNLOAD	*/UNLOAD*

restrictions

- If the MOUNT command was used with the /SHARE keyword, it is not actually dismounted until all the users who mounted it have dismounted it.
- Dismounting a volume causes the logical name assigned to the volume by the previous MOUNT command to be removed from the appropriate logical name table. The logical name is either the one specified in the MOUNT command or the default name of DISK$volume-label or TAPE$volume-label.
- If a volume is mounted /SYSTEM or /GROUP, a DMO command causes the volume to be dismounted even if others are currently using it. However, dismounting the volume also results in removing the name from the system or group logical name table, which requires privilege.
- If the device has been allocated with an ALLOCATE command, it remains allocated after the volume is dismounted.
- If the volume has been mounted using the /NOSHARE keyword, the system automatically allocates the device when the volume is mounted and deallocates it when the volume is dismounted.
- RSX–11 MCR supports the DMO command, but the keywords are different.
- The /NOUNLOAD keyword provides the same functionality as the /LOCK=NOUNLOAD keyword in RSX–11 MCR for this command.

prompt

_Device: device-name

command parameter

device-name[:]

Specifies the name of the device to be dismounted. The device name can be a physical device name or a logical name.

If you specify a colon following the device name, the command interpreter strips the colon.

command keywords

/ABORT

Specifies that the volume is to be dismounted, regardless of who mounted it. If the volume was mounted with the /SHARE keyword, the /ABORT keyword causes the volume to be dismounted for all users who mounted it.

You must have the user privileges GRPNAM and SYSNAM to dismount group and system volumes (respectively). To use the /ABORT keyword with a volume that is mounted neither group nor system, you must be the owner of the volume or have the privilege VOLPRO.

/CLUSTER

Specifies that after the DISMOUNT command successfully dismounts the volume on the local node, the volume is to be dismounted on every other node in the existing VAXcluster (that is, the volume is dismounted cluster-wide). If the system is not a member of a VAXcluster, the /CLUSTER keyword has no effect.

/UNIT

Specifies, for disk volume sets, that only the volume on the specified device is dismounted. By default, the DMO command dismounts all volumes in a volume set.

It is recommended that you not dismount the root volume of a volume set, since the Master File Directory (MFD) for the volume set is on the root volume. It may be impossible to access files on a volume set if the MFD is not accessible.

/UNLOAD
/NOUNLOAD

Indicates that the device is to be physically unloaded. /UNLOAD is equivalent to pressing the button on the drive to unload the volume (making it not ready). /UNLOAD is the default action; specify /NOUNLOAD if you want the unit to remain loaded (ready).

EXAMPLE

```
> MOUNT/SHARE DBA3: PROJ123 DISK
    .
    .
    .
> DMO DISK
```

The MOUNT command requests access to the volume labeled PROJ123 on DBA3 and places the logical name DISK in the current process's logical name table. Because the volume was requested using the /SHARE keyword, the volume may or may not already be mounted by other users.

The DMO command releases access to the volume PROJ123 for the current process and removes the name DISK from the process logical name table. The actual dismounting does not occur until all accessors have issued DMO commands.

DUMP

The DUMP command displays or prints the contents of files or volumes in ASCII, hexadecimal, decimal, or octal representation.

FORMAT

DUMP *file-spec[, . . .]*

Command Keywords	**Defaults**
/ALLOCATED	
/BLOCKS[=(option[, . . .])]	*See text*
/BYTE	*/LONGWORD*
/DECIMAL	*/HEXADECIMAL*
/FILE_HEADER	
/[NO]FORMATTED	*/FORMATTED*
/HEADER	
/HEXADECIMAL	*/HEXADECIMAL*
/LONGWORD	*/LONGWORD*
/NUMBER[=n]	*See text*
/OCTAL	*/HEXADECIMAL*
/OUTPUT[=file-spec]	*/OUTPUT=SYS$OUTPUT*
/PRINTER	
/RECORDS[=(option[, . . .])]	*See text*
/WORD	*/LONGWORD*

restrictions

- You can use DUMP only on files or volumes for which you have read privilege.
- Restrictions apply to the following keywords:

  /ALLOCATED
  /BLOCKS
  /BYTE
  /DECIMAL
  /HEADER
  /HEXADECIMAL
  /LONGWORD
  /OCTAL
  /OUTPUT
  /PRINTER
  /RECORDS
  /WORD

  See the keyword descriptions for more information.

prompt

_File:	file-spec

command parameter

file-spec

Specifies the files or the volume to be displayed.

If the specified device is not a disk, tape, or network device, or if the device is mounted with the /FOREIGN keyword, the file specification must contain only the device name.

If the specified device is either a network device, a disk, or tape device that is mounted without the /FOREIGN keyword, the file specification can contain wildcards.

DESCRIPTION

By default, the DUMP command formats output in ASCII and in hexadecimal longwords. You can specify another format for the dump by using a radix keyword (/OCTAL, /DECIMAL, or /HEXADECIMAL) and/or a length keyword (/BYTE, /WORD, or /LONGWORD).

Dumping Files:

If the input medium is either a network device or a disk or tape device that is mounted without the /FOREIGN keyword, the DUMP command operates on files. You can dump files in records or blocks. Wildcard specifications can be used to select a group of files for processing.

Dumping Volumes:

If the input medium is not a disk or tape device or is mounted with the /FOREIGN keyword, the DUMP command operates on the input device as a non-file-structured medium. Disk devices are dumped in 512-byte logical blocks. Other devices are dumped in physical blocks. No repositioning of the input medium occurs; consecutive blocks on a tape can be dumped by several DUMP commands.

Reading Dumps:

The ASCII representation is read left to right. The hexadecimal, decimal, or octal representation is read right to left.

Specifying Numeric Keyword Values:

The values for the /BLOCKS, /RECORDS, and /NUMBER keywords can be specified either as decimal numbers or with a leading %X, %O, or %D to signify hexadecimal, octal, or decimal numbers, respectively. For example, the following are all valid ways to specify the decimal value 24:

```
24
%X18
%O30
%D24
```

command keywords

/ALLOCATED

Specifies that the dump include all blocks allocated to the file. By default, the dump does not include blocks following the end-of-file block.

You can specify /ALLOCATED if the input is a disk that is mounted without the /FOREIGN keyword. If you specify /ALLOCATED, you cannot specify /RECORDS.

/BLOCKS[=(option[, . . .])]

Specifies that the input medium be dumped one block at a time. This is the default for all devices except network devices. You cannot specify /BLOCKS for network devices.

You can use one or more of the following options to select a range of blocks to be dumped:

Option	Function
START:n	Specifies the number of the first block to be dumped. By default, the dump begins with the first block of the file or device.
END:n	Specifies the number of the last block to be dumped. By default, the dump ends with the last block of the file or device. If the input is a disk file, the /ALLOCATED keyword determines whether the last block is the end of file block or the last allocated block.
COUNT:n	Specifies the number of blocks to be dumped. This option provides an alternative way to specify the last block to be dumped.

If you specify two options, separate them with a comma and enclose the list in parentheses. You cannot specify both END and COUNT.

Blocks are usually numbered beginning with 1. However, for a disk device that is mounted with the /FOREIGN keyword, blocks are numbered beginning with 0.

If you specify /BLOCKS, you cannot specify /RECORDS.

/BYTE

Specifies that the dump be formatted in bytes. The default format is composed of longwords.

If you specify /BYTE, you cannot specify /WORD or /LONGWORD.

/DECIMAL

Specifies that the dump be formatted in decimal. The default format is in hexadecimal representation.

If you specify /DECIMAL, you cannot specify /HEXADECIMAL or /OCTAL.

/FILE_HEADER

Specifies that data blocks that are valid Files-11 file headers be printed in an interpreted representation. All other data blocks are printed in the selected radix and length.

/FORMATTED
/NOFORMATTED

Specifies whether the file header is displayed in a formatted or unformatted representation. This keyword is meaningful only when the /HEADER keyword is specified. If you specify /FORMATTED, the file header is printed in an interpreted format. When you specify /NOFORMATTED, the file header is printed in the selected radix and length.

The default is /FORMATTED.

/HEADER

Specifies that the output include the file header as well as the access control list. You can use the /FORMATTED keyword to control the format of the display. You can display the file header without displaying file contents by also specifying /BLOCKS=COUNT=0.

You can use the /FILE_HEADER keyword with /HEADER to have Files–11 file headers printed in an interpreted representation.

You can use the /HEADER keyword if the input medium is a disk or tape mounted without the /FOREIGN keyword.

By default, the file header is not displayed.

/HEXADECIMAL

Specifies that the dump be formatted in hexadecimal. The default format is in hexadecimal representation.

If you specify /HEXADECIMAL, you cannot specify /DECIMAL or /OCTAL.

/LONGWORD

Specifies that the dump be formatted in longwords. /LONGWORD is the default format.

If you specify /LONGWORD, you cannot specify /BYTE or /WORD.

/NUMBER[=n]

Specifies how byte offsets are assigned to the lines of output. If you specify /NUMBER, the byte offsets increase continuously through the dump, beginning with the specified value. If you omit the value, the initial byte offset is zero.

By default, the byte offset is reset to zero at the beginning of each block or record.

/OCTAL

Specifies that the dump be formatted in octal. The default format is in hexadecimal representation.

If you specify /OCTAL, you cannot specify /DECIMAL or /HEXADECIMAL.

/OUTPUT[=file-spec]

Specifies that the DUMP output be written to the specified file. By default, the DUMP command writes output to SYS$OUTPUT. If you specify /OUTPUT without a file specification, the DUMP command writes output to a file with the same file name as the input file and the file type DMP.

No wildcard characters are allowed in the file specification.

If you specify /OUTPUT, you cannot specify /PRINTER.

/PRINTER

Specifies that output be queued to the system printer. By default, the DUMP command writes output to SYS$OUTPUT. If you specify /PRINTER, the DUMP command writes output to a file with the same file name as the input file and a file type DMP, and it queues the file to SYS$PRINT.

If you specify /PRINTER, you cannot specify /OUTPUT.

/RECORDS[=(option[, . . .])]

Specifies that a file be dumped one record at a time. By default, input is dumped one block at a time for all devices except network devices. The default for network devices is /RECORDS.

You can specify one or more of the following options to select a range of records to be dumped:

Option	Function
START:n	Specifies the number of the first record to be dumped. By default, the dump begins with the first record of the file.
END:n	Specifies the number of the last record to be dumped. By default, the dump ends with the last record of the file.
COUNT:n	Specifies the number of records to be dumped. This option provides an alternative way to specify the last record to be dumped.

If you specify two options, separate them with a comma and enclose the list in parentheses. You cannot specify both END and COUNT.

Records are numbered beginning with 1.

You can use the /RECORDS keyword if the input is a network device or a disk or tape mounted without the /FOREIGN keyword. If you specify /RECORDS, you cannot specify /ALLOCATED or /BLOCKS.

/WORD

Specifies that the dump be formatted in words. The default format is composed of longwords.

If you specify /WORD, you cannot specify /BYTE or /LONGWORD.

EXAMPLES

1
```
> DUMP ORION.DAT
Dump of file DISKO:[NORMAN]ORION.DAT;1 on 27-DEC-1984 15:43:26.08
File ID (3134,818,2)   End of file block 1 / Allocated 3

Virtual block number 1 (00000001), 512 (0200) bytes

 706D6173 20612073 69207369 68540033 3.This is a samp 000000
 73752065 62206F74 20656C69 6620656C le file to be us 000010
 61786520 504D5544 2061206E 69206465 ed in a DUMP exa 000020
 00000000 00000000 0000002E 656C706D mple............ 000030
 00000000 00000000 00000000 00000000 ................ 000040
 00000000 00000000 00000000 00000000 ................ 000050
 00000000 00000000 00000000 00000000 ................ 000060
          .
          .
          .
 00000000 00000000 00000000 00000000 ................ 0001E0
 00000000 00000000 00000000 00000000 ................ 0001F0
```

The DUMP command displays the contents of ORION.DAT in hexadecimal longword format, beginning with the first block in the file.

2

```
> DUMP ORION.DAT/OCTAL/BYTE

Dump of file DISKO:[NORMAN]ORION.DAT;1 on 27-DEC-1984 15:45:33.58
File ID (74931,2,1)   End of file block 1 / Allocated 3

Virtual block number 1 (00000001), 512 (0200) bytes

 151 040 163 151 150 124 000 063 3.This i 000000
 160 155 141 163 040 141 040 163 s a samp 000010
 040 145 154 151 146 040 145 154 le file  000020
 163 165 040 145 142 040 157 164 to be us 000030
 040 141 040 156 151 040 144 145 ed in a  000040
 141 170 145 040 120 115 125 104 DUMP exa 000050
 377 377 000 056 145 154 160 155 mple.... 000060
 000 000 000 000 000 000 000 000 ........ 000070
 000 000 000 000 000 000 000 000 ........ 000100
 000 000 000 000 000 000 000 000 ........ 000110

            .
            .
            .

 000 000 000 000 000 000 000 000 ........ 000760
 000 000 000 000 000 000 000 000 ........ 000770
```

The DUMP command displays the image of the file ORION.DAT, formatted in octal bytes, beginning with the first block.

EDT

The EDT command invokes the EDT interactive text editor.

FORMAT

EDT *file-spec*

Command Keywords	**Defaults**
/[NO]COMMAND[=command-file]	*/COMMAND=SYS$LIBRARY:EDTSYS.EDT*
/[NO]CREATE	*/CREATE*
/[NO]JOURNAL[=journal-file]	*/JOURNAL=input-filename.JOU*
/[NO]OUTPUT[=output-file]	*/OUTPUT=input-filename.typ;n+1*
/[NO]READ_ONLY	*/NOREAD_ONLY*
/[NO]RECOVER	*/NORECOVER*

restrictions

None.

prompt

_File: file-spec

command parameter

file-spec

Specifies the file to be created or edited using the EDT editor. If the file does not exist, it is created.

The EDT editor does not provide a default file type when creating files; if you do not include a file type, it is null. The file must be a disk file on a Files–11 formatted volume.

No wildcard characters are allowed in the file specification.

DESCRIPTION

The EDT editor creates or edits text files. You can use EDT to enter or edit text in three modes: keypad, line, or nokeypad. Keypad editing, which is screen-oriented, is available on VT52- and VT100-type terminals. Using keypad editing, you can see several lines of text at once and move the cursor throughout the text in any direction. Line editing operates on all terminals. In fact, if you have a hardcopy terminal, line editing is the only way you can use EDT. You may prefer line editing if you are accustomed to editing by numbered lines. Nokeypad editing is available on VT52- and VT100-type terminals. You may want to try nokeypad editing if you plan to redefine keys often or if you are accustomed to a screen-oriented editor.

When you invoke EDT, you are put into line mode. If you are editing an existing file, EDT prints the line number and text for the first line of the file. If you are creating a new file, EDT prints the following message:

```
Input file does not exist
[EOB]
```

In either case, EDT then prints its prompt, which is the asterisk (*).

command keywords

/COMMAND[=file-spec] /NOCOMMAND

Determines whether EDT uses a start-up command file or not. The keyword /COMMAND should be followed by an equal sign (=) and the specification of the command file. The default file type for command files is EDT.

```
> EDT/COMMAND=XEDTINI.EDT MEMO.DAT
```

If you do not include the /COMMAND=command file keyword, EDT processes the systemwide start-up command file SYS$LIBRARY:EDTSYS.EDT. If this file does not exist, EDT looks for the file named EDTINI.EDT in your default directory. If none of these files exists, EDT begins your editing session in the default state.

To prevent EDT from processing either the systemwide start-up command file or the EDTINI.EDT file in your default directory, use the /NOCOMMAND keyword.

```
> EDT/NOCOMMAND MEMO.DAT
```

No wildcard characters are allowed in the file specification.

/CREATE /NOCREATE

Controls whether EDT creates a new file when the specified input file is not found. Normally, EDT creates a new file to match the input file specification if it cannot find that file name in the specified directory. When you use /NOCREATE in the EDT command line and type a specification for a file that does not exist, EDT prints an error message and returns you to the DCL command level.

```
> EDT/NOCREATE NEWFILE.DAT
Input file does not exist
>
```

/JOURNAL[=journal-file] /NOJOURNAL

Determines whether EDT keeps a journal file during your editing session. The default file name for the journal file is the same as the input file name. The default file type is JOU. The /JOURNAL keyword enables you to use a different file specification for the journal file.

```
> EDT/JOURNAL=SAVE MEMO.DAT
```

If you are editing a file from another directory and want the journal file to be located in that directory, you must use the /JOURNAL keyword with a file specification that includes the directory name. Otherwise, EDT creates the journal file in the default directory.

The directory that is to contain the journal file should not be write-protected.

To prevent EDT from keeping a record of your editing session, use the /NOJOURNAL keyword in the EDT command line.

```
> EDT/NOJOURNAL MEMO.DAT
```

Once you have created a journal file, use the /RECOVER keyword to have EDT process the commands contained in the journal file.

No wildcard characters are allowed in the file specification.

/OUTPUT=output-file
/NOOUTPUT

Determines whether EDT creates an output file at the end of your editing session. An equal sign (=) separates the /OUTPUT keyword from the output file specification. The default file specification for both the input file and the output file is the same. Use the /OUTPUT keyword to give the output file a different file specification from the input file.

```
> EDT/OUTPUT=OUTMEM.DAT MEMO.DAT
```

You can include directory information as part of your output file specification to send output to another directory.

```
> EDT/OUTPUT=[BARRETT.MAIL]MEMO.DAT MEMO.DAT
```

The /NOOUTPUT keyword suppresses the creation of an output file, but not the creation of a journal file. If you are testing some edits and are not sure you want an output file, you can use /NOOUTPUT. A system interruption will not prevent you from recreating your editing session, because a journal file is still being maintained. If you decide you want to keep your editing work, you can save the edited text, even though you specified /NOOUTPUT, by using the line mode command WRITE to put the text into an external file before you end the session.

```
> EDT/NOOUTPUT MEMO.DAT
```

No wildcard characters are allowed in the file specification.

/READ_ONLY
/NOREAD_ONLY

Determines whether EDT keeps a journal file and creates an output file. With the default /NOREAD_ONLY, EDT maintains the journal file and creates an output file when it processes the line mode command EXIT. Using the /READ_ONLY keyword is like using both the /NOJOURNAL and /NOOUTPUT keywords.

```
> EDT/READ_ONLY CALENDAR.DAT
```

Use /READ_ONLY when you are searching a file. If you then want to modify the file, use WRITE to save your changes. Remember, however, that you have no journal file.

/RECOVER
/NORECOVER

Determines whether EDT reads a journal file at the start of the editing session. The default is /NORECOVER.

When you use the /RECOVER keyword, EDT reads the appropriate journal file and processes whatever commands it contains.

```
> EDT/RECOVER MEMO.DAT
```

If the journal file type is not JOU or the file name is not the same as the input file name, you must include both the /JOURNAL keyword and the /RECOVER keyword.

```
> EDT/RECOVER/JOURNAL=SAVE.XXX MEMO.DAT
```

Because /NORECOVER is the default for EDT, you do not need to specify it in a command line.

EXAMPLES

1

```
> EDT/OUTPUT=NEWFILE.TXT OLDFILE.TXT
    1          This is the first line of file text
*
```

This EDT command invokes the EDT editor for editing the file OLDFILE.TXT. EDT processes the systemwide start-up command file SYS$LIBRARY:EDTSYS.

EDT. If this file does not exist, EDT looks for the file EDTINI.EDT in your default directory. If none of these files exists and you have not specified the /COMMAND keyword, EDT begins your editing session in the default state. When the session ends, the edited file has the name NEWFILE.TXT.

2

```
> EDT/RECOVER OLDFILE.TXT
```

This EDT command invokes the EDT editor for recovering from an abnormal exit during a previous editing session. EDT opens the file OLDFILE.TXT, and then processes the journal file OLDFILE.JOU. Once the journal file has been processed, interactive editing can continue.

EXAMINE

The EXAMINE command displays the contents of virtual memory at the terminal. It is used after an RSX–11 task image or VAX/VMS native image has been interrupted by CTRL/Y. If you interrupt an image while it is executing, you can examine its locations; then you can resume image execution by issuing a CONTINUE command.

The information is displayed in the following format:

```
address:  contents
```

Information that is inaccessible at user mode is displayed as asterisks (****).

FORMAT

EXAMINE *location[:location]*

Command Keywords	**Defaults**
/ASCII	*None*
/BYTE	
/DECIMAL	
/HEXADECIMAL	
/LONGWORD	
/OCTAL	
/WORD	

restrictions

- You must have user mode read and write access to the location in virtual memory whose contents you want to examine.
- The EXAMINE command analyzes expressions arithmetically. Therefore, keywords are interpreted correctly only when they appear immediately after the command name.
- RSX–11 MCR does not support the EXAMINE command. Instead of the DEPOSIT and EXAMINE commands, RSX–11 MCR uses the OPE (Open) command.

prompts

None.

command parameter

location [:location]

Specifies the address or range of addresses in virtual memory to be displayed. If you specify a range of addresses, you must separate the two addresses with a colon (:). The second address must be larger than the first. Locations can be specified using expressions that contain the arithmetic operators +, –, *, and /.

Addresses are always displayed in hexadecimal radix.

The EXAMINE and DEPOSIT commands set a pointer to the last byte examined. You can refer to this pointer as "." when specifying the first location in a subsequent EXAMINE command or as the deposit location in a subsequent DEPOSIT command.

DESCRIPTION

When used with the DEPOSIT command, the EXAMINE command aids in the interactive debugging of programs. You do not need to taskbuild the task image with a debugger to use these commands.

The initial default radix for the EXAMINE command is hexadecimal. The default applies to both the interpretation of numeric literals in the command line and to the format of the data displayed. If you use a radix keyword to modify the command, that radix becomes the default for subsequent EXAMINE and DEPOSIT commands.

The initial default length unit for the EXAMINE command is a longword. The EXAMINE command displays data one longword at a time with blanks between longwords. If you use a /BYTE or /WORD keyword to modify the command, that unit becomes the default for subsequent EXAMINE or DEPOSIT commands.

RSX–11 task images start at virtual address 0 under VAX–11 RSX. This is the same starting address they have under RSX–11.

command keywords

/ASCII

Indicates that data at the specified location is to be displayed in ASCII representation. Binary values that do not have ASCII equivalents are displayed as periods (.).

When ASCII is specified or is the default, hexadecimal is the default radix for numeric literals that are specified on the command line.

/BYTE

Indicates that data at the specified location is to be displayed one byte at a time.

/DECIMAL

Indicates that the data is to be displayed in decimal format and changes the default radix for numeric literals.

/HEXADECIMAL

Indicates that the data is to be displayed in hexadecil format and changes the default radix for numeric literals. This is the initial default of the command; refer to the restrictions following the examples below.

/LONGWORD

Indicates that data at the specified location is to be displayed one longword at a time. This is the initial default of the command; refer to the preceding restrictions for more information.

/OCTAL

Indicates that the data is to be displayed in octal format and changes the default radix for numeric literals.

/WORD

Indicates that data at the specified location is to be displayed one word at a time.

EXAMPLES

1

```
> RUN RSXPROG
      .
      .
CTRL/Y INTERRUPT
> EXAMINE/WORD/OCT 2677
   000005BF:  007402
> CONTINUE
```

The RUN command begins executing the RSX–11 task image RSXPROG. While the task image is running, pressing CTRL/Y interrupts it. The EXAMINE command requests that the contents of the word addresses by virtual memory location 5BF (16) be displayed in octal. After the display, the CONTINUE command causes the image to resume execution.

2

```
> RUN VAXPROG
      .
      .
CTRL/Y INTERRUPT
> EXAMINE/LONG/ASC 1C00+50:1C00+7F
   00001C50:  FOOLISH CONSISTENCY IS THE HOBGOBLIN OF LITTLE M
> EXA 1C80:1C83
   00001C80:  INDS
```

While the VAX/VMS native image VAXPROG is executing, CTRL/Y interrupts it. The EXAMINE command then requests that all the data in the virtual address range specified be displayed in ASCII format. Note that the addresses must be expressed in hexadecimal and are interpreted as such.

EXECUTE PROCEDURE (@)

The EXECUTE PROCEDURE (@) requests execution of an indirect command file by the Indirect Command Processor.

FORMAT

@file-spec *[p1 [p2 [...p9]]]*

Command Keywords	**Defaults**
/[NO]CLI	*/CLI*
/LB	
/[NO]LO	*/NOLO*
/[NO]MC	*/MC*
/[NO]TR	*/NOTR*

restrictions

None.

prompts

None.

command parameters

file-spec

Specifies the command procedure to be executed or the device from which input for the preceding command is to be read.

If you do not specify a file type, Indirect uses the default file type CMD.

p1,p2,...p9

Specifies from one to nine optional parameters to pass to the indirect command file. The parameters assign numeric or character string values to the symbols, p1, p2, and so on up to p9 in the order of entry. The symbols are local to the indirect command file. Unspecified parameters are set to null strings. Separate each parameter with one or more blanks.

You can specify a numeric value for a parameter using any valid arithmetic expression. You can also specify a character string value using any alphanumeric or special characters.

DESCRIPTION

You can cause symbol substitution by enclosing a symbol in single quotation marks. Refer to Section 5 for information on symbol substitution and using indirect command files.

When an indirect command file terminates, any data files that remain open are closed by MCR.

command keywords

Refer to Section 5 for descriptions of the /[NO]CLI, /LB, /[NO]LO, /[NO]MC, and /[NO]TR keywords.

EXAMPLES

1

```
> @STDJOB.CMD PAYROL FICA
```

This command requests execution of the indirect command file PAYROL.CMD and provides two parameters, PAYROL and FICA. The indirect command file contains the commands in the following example.

2

```
.enable substitution
RUN 'P1'
PRINT 'P1'.DAT
RUN 'P2'
PRINT 'P2'.DAT
```

When the indirect command file is executed, the two parameters are substituted into the RUN and PRINT commands. The result is that the indirect command file runs PAYROL.EXE and FICA.EXE and prints the files PAYROL.DAT and FICA.DAT.

HELP

The HELP command provides information available in the system HELP files (or in the MCR HELP library if your system manager installed it while installing VAX–11 RSX). The information is displayed on the current default output device (SYS$OUTPUT). You can also display information from your own set of HELP libraries.

FORMAT

HELP *[option ...]*

Command Keywords	**Defaults**
/[NO]INSTRUCTIONS	*/INSTRUCTIONS*
/[NO]LIBLIST	*/LIBLIST*
/[NO]LIBRARY[=file-spec]	*/LIBRARY=HELPLIB*
/OUTPUT[=file-spec]	
/[NO]PAGE	*/PAGE*
/[NO]PROMPT	*/PROMPT*
/[NO]USERLIBRARY[=(table[,...])]	*/USERLIBRARY=ALL*

restrictions

None.

prompt

See the description of the /PROMPT keyword.

command parameters

option ...

Specifies one or more options that indicate the information you want. Information is arranged in a hierarchical manner. The levels are:

1. None: Describes the HELP command and lists options that you can specify to obtain information about other topics that are documented in the root library. Each item in the list is an option in the first level of the hierarchy.
2. Topicname: Describes a topic documented in either the root library or one of the other default libraries that are enabled, and lists options for additional information available on this topic.
3. Topicname followed by a specific item: Provides descriptions of the specified subtopics for this topic.
4. @file-spec followed by any of the above: Specifies a HELP library to replace the current root library and then proceeds as above. The file specification must take the same form as the file specification included with the /LIBRARY command keyword. However, if the specified library is an enabled user-defined default library, the file specification can be abbreviated to any unique substring of that default library's logical name translation.

If you use an asterisk (*) in place of any option, the HELP command displays all information available at that level.

If you use an ellipsis after any option, HELP displays all the information on the specified topic and all subtopics of that topic.

You can specify percent signs (%) and asterisks (*) in the option as wildcard characters.

DESCRIPTION

To use the HELP facility in its simplest form, issue the HELP command from your terminal. HELP displays a list of topics at your terminal and the prompt "Topic?" . If you want to see information on one of the topics, type the topic name after the prompt. The system will display information on that topic.

If the topic has subtopics, HELP will prompt you with Subtopic?. If you want information on one of the subtopics, type the name after the prompt. If you want information on another subtopic or topic, press RETURN. The system will first prompt with "Subtopic?" , and another press of RETURN will bring the "Topic?" prompt. Or, if you are finished with the HELP facility and want to exit, you can press RETURN again to return to the MCR command level.

command keywords

/INSTRUCTIONS
/NOINSTRUCTIONS

Controls whether the HELP command displays information on how to use the HELP facility. By default, the HELP command display includes a description of the facility and the format, along with the list of topics. If you specify /NOINSTRUCTIONS, only the list of topics is displayed.

/LIBLIST
/NOLIBLIST

Controls whether a list of all available default HELP libraries is output at the topic level. The default is /LIBLIST.

/LIBRARY[=file-spec]
/NOLIBRARY

Controls whether an alternate HELP library is used in place of the default system library, SYS$HELP:HELPLIB.HLB or SYS$HELP:MCRHELPLIB.HLB. The specified library is used as the main HELP library and is searched for HELP information before any user-defined default HELP libraries are checked.

If you omit the device and directory specification, the default is SYS$HELP, the logical name of the location of the system HELP libraries. The default file type is HLB.

The /NOLIBRARY keyword is used to exclude the default HELP library from the library search order.

/OUTPUT[=file-spec]

Controls whether HELP information is output to a specified file instead of to the current SYS$OUTPUT device. If the file specification does not include a file name or type, HELP assumes the default HELP.LIS. If you omit the device and directory specification, the default directory is your current default directory.

By default, output is written to the current SYS$OUTPUT device.

/PAGE /NOPAGE

Controls whether page breaks are generated when the terminal screen is full.

The default is /PAGE.

/PROMPT /NOPROMPT

Controls whether HELP initiates an interactive session once the initial HELP command is specified. By default, HELP initiates an interactive session.

If you specify /PROMPT, one of four different prompts is displayed, requesting you to specify a particular HELP topic or subtopic. Each prompt represents a different level in the hierarchy of HELP information. The four prompt levels are:

1. Topic?—The root library is the main library. The appearance of this prompt means that you are not currently examining HELP for a particular topic.
2. [library-spec] Topic?—The root library is a library other than the main library. The appearance of this prompt means that you are not currently examining HELP for a particular topic.
3. [option ...]subtopic?—The root library is the main library. The appearance of this prompt means that you are currently examining HELP for a particular topic (and subtopics).
4. A combination of 2 and 3.

When you encounter one of these prompts, you can respond by entering any one of the following responses:

Response	Action in the Current Prompt Environment
option[,...]	(1,2) Searches all enabled libraries for these options. (3,4) Searches additional HELP libraries for the current topic (and subtopic) for these options.
@file-spec option[,...]	(1,2) Same as option [,...], except that the library specified by @file-spec is now the root library. If the specified library does not exist, treats @file-spec as a normal option. (3,4) Same as option [,...]; treats @file-spec as a normal option.
?	(1,2) Displays a list of topics available in the root library. (3,4) Displays the list of subtopics of the current topic (and subtopics) for which HELP exists.
<RET>	(1) Exits from HELP. (2) Changes root library to main library. (3,4) Prompts for a topic or subtopic at the next higher level.
<CTRL/Z>	(1,2,3,4) Exits from HELP.

/USERLIBRARY[=(table[,...])] /NOUSERLIBRARY

Controls whether HELP searches one or more user-specified root libraries after it has searched the root HELP library.

When you specify the /USERLIBRARY keyword, HELP searches the process, group, and system logical name tables for logical names of the form HLP$LIBRARY, HLP$LIBRARY_1, HLP$LIBRARY_2, and so on, to find the file specifications of any user-defined libraries. When HELP searches for information on a given topic, it proceeds in the following search order: root library, main library (if not the same as the root library), process libraries, group libraries, system libraries, and root library. Note that the root library is searched a second time if the search up to that points fails. In this way, the context is returned to the root library from which the search was initiated.

If you specify more than one table for HELP to search, separate them with commas and enclose the list in parentheses. You can specify any of the following tables:

Table	Purpose
ALL	HELP searches the process, group, and system logical name tables for user-defined library definitions.
NONE	HELP does not search any of the logical name tables; this specification is the same as /NOUSERLIBRARY.
PROCESS	HELP searches the process logical name table for user-defined library definitions.
GROUP	HELP searches the group logical name table for user-defined library definitions.
SYSTEM	HELP searches the system logical name table for user-defined library definitions.

By default, if you omit the keyword, /USERLIBRARY=ALL is assumed.

EXAMPLES

1
```
> HELP
HELP

  .
  .  (HELP message text and subtopics, if any)
  .
Topic?
```

Issuing the HELP command without any keywords or parameters produces a display of the HELP topics available from the root HELP library, SYS$HELP:HELPLIB.HLB.

If you type one of the listed topics in response to the Topic? prompt, the HELP facility displays information about that topic and a list of subtopics (if there are any). If one or more subtopics exist, the HELP utility will prompt you for a subtopic.

```
Topic? APPEND
APPEND

  .
  .  (HELP message text and list of subtopics, if any)
  .
APPEND Subtopic?
```

If you type a subtopic name, HELP displays information about that subtopic:

```
APPEND Subtopic? /LOG
APPEND
  /LOG
  .
  .  (HELP message text)
  .
APPEND Subtopic?
```

If one or more sub-subtopics exist, HELP will prompt you for a sub-subtopic. Otherwise, as in Example 1, the facility will prompt you for another subtopic of the topic you are currently reading.

Typing a question mark (?) will redisplay the HELP message and options at your current level. Pressing RETURN will either (1) move you back to the previous HELP level if you are in a subtopic level or (2) terminate HELP if you are at the first level. Pressing CTRL/Z terminates HELP at any level.

2

```
> HELP/NOPROMPT APPEND/LOG
  .
  .  (APPEND/LOG HELP message)
  .
>
> HELP/NOPROMPT/PAGE COPY  *
  .
  .  (HELP messages on all first-level COPY subtopics)
  .
>
```

You can get HELP on a specific topic or subtopic, or on a range of topics or subtopics. You can also choose not to be prompted for additional topics or subtopics.

The two HELP commands request HELP on specific topics. In each case, HELP displays the HELP message you request and then returns you to MCR command level and the angle bracket prompt.

The first command requests HELP on the /LOG keyword of the APPEND command. The asterisk in the second example is a wildcard character. It signals the HELP facility to display information about all COPY subtopics, which HELP then displays in alphabetical order. The /NOPROMPT keyword suppresses prompting in both sample commands. The /PAGE keyword on the second HELP command causes output to the screen to stop after each screenful of information is displayed.

3

```
> HELP FILL
Sorry, no documentation on FILL
Additional information available:
  .
  .  (list of first-level topics )
  .
Topic? @EDTHELP FILL
FILL
  .
  .  (FILL HELP message)
  .
@EDTHELP Topic?
```

When you issue a request for HELP on a topic that is not in the default HELP library, you can instruct the HELP facility to search another HELP library for the topic. In this example, entering the command @EDTHELP FILL instructs the HELP facility to search the HELP library SYS$HELP:EDTHELP.HLB for information on FILL, an EDT editor command. HELP displays the message and prompts you for another EDT editor topic.

INITIALIZE

The INITIALIZE command formats and writes a label on a mass storage volume. The default format for disk volumes created using the MCR command INITIALIZE is Files–11 Structure Level 2. The INITIALIZE command can also initialize Files–11 Structure Level 1 volumes. The default format for magnetic tape volumes is the ANSI standard for tape labels and file structure for information interchange, Level III.

FORMAT

INITIALIZE *device-name[:] volume-label*

Command Keywords	**Defaults**
/ACCESSED=n (D)	*/ACCESSED=3*
/BADBLOCKS=(area[,...]) (D)	
/CLUSTER_SIZE=n (D)	*See text*
/DATA_CHECK[=(option[,...])] (D)	*See text*
/DENSITY=density-value (T)	
/DIRECTORIES=n (D)	*/DIRECTORIES=16*
/[NO]ERASE	*/NOERASE*
/EXTENSION=n (D)	*/EXTENSION=5*
/FILE_PROTECTION=code (D)	*See text*
/GROUP (D)	
/HEADERS=n (D)	*/HEADERS=16*
/[NO]HIGHWATER	*See text*
/INDEX=position (D)	*/INDEX=MIDDLE*
/LABEL=option (T)	*See text*
/MAXIMUM_FILES=n (D)	*See text*
/OVERRIDE=(option[,...]) (T)	
/OWNER_UIC=[uic]	
/PROTECTION=code	*See text*
/[NO]SHARE (D)	*/SHARE*
/STRUCTURE=level (D)	*/STRUCTURE=2*
/SYSTEM (D)	*See text*
/USER_NAME=string (D)	*Current user name*
/[NO]VERIFIED (D)	*See text*
/WINDOWS=n (D)	*/WINDOWS=7*

D = applicable to disk only.

T = applicable to tape only.

Keywords are categorized according to the descriptions that follow.

restrictions

None.

prompts

_Device:	device-name[:]
_Label:	volume-label

command parameters

device-name[:]

Specifies the name of the device on which the volume to be initialized is physically mounted. The device does not have to be allocated first; however, it is the recommended practice.

volume-label

Specifies the label to be written on the volume. For a disk volume, you can specify a maximum of 12 alphanumeric characters. For a tape volume, you can specify a maximum of 6 alphanumeric characters.

DESCRIPTION

Many of the INITIALIZE command keywords are used to maximize input/output efficiency. For information on these parameters, see the *VAX/VMS System Management and Operations Guide.*

You do not need any special privileges to initialize a blank disk or tape volume. If a volume has previously been written, however, your UIC must match the owner UIC on the volume, or you must have the user privilege to override volume protection. In the case of a tape that has been written, you can initialize the volume if you are allowed write access to it.

The compatible form of the INITIALIZE command between the VAX–11 RSX and RSX–11 MCR is:

```
INI device-name:volume-label
```

keywords applicable to disk and tape

/ERASE
/NOERASE

Controls the Data Security Erase (DSE) operation on the volume before initializing it. The /ERASE keyword applies to Files–11 Structure Level 2 disk and ANSI tape volumes, and is valid for tape devices that support the hardware erase function, such as TU78 and MSCP tapes.

If you specify /ERASE, a DSE operation is performed on the volume. For disk devices, the ERASE volume attribute is set. In effect, each file on the volume is erased when it is deleted.

Note that the amount of time taken by the DSE operation depends on the volume size; INITIALIZE/ERASE is always slower than INITIALIZE/NOERASE. The default is /NOERASE.

/OWNER_UIC=[g,m]

Specifies the User Identification Code (UIC) to be assigned ownership of the volume and of system files on the volume. The group (g) field of the UIC can have a value in the range 0 through 37777 (8) and the member (m) field can have a value in the range 0 through 177777 (8).

The brackets are required.

If you do not specify /OWNER_UIC, your current UIC is assigned ownership of the volume.

/PROTECTION=code

Specifies the protection to be applied to the volume. The protection determines which users can read files, write files, create directories, and delete files on the volume.

Specify the protection code according to the standard syntax rules described in Section 2. If you do not specify /PROTECTION, all categories of users are allowed all types of access. If you omit a category of user when specifying protection, that category is denied all access.

The system applies only read and write access restrictions for magnetic tapes; create and delete are meaningless. In addition, both the system and the owner are given read and write access regardless of what you specify in the protection code.

keywords applicable only to disk

/ACCESSED=n

Specifies the number of directories to be maintained in system space for ready access. The maximum value of n is 255. Operator (OPER) privilege is required to use the /ACCESSED keyword.

If you do not specify /ACCESSED, the INITIALIZE command uses a value of 3 by default.

/BADBLOCKS=(area[,...])

Specifies faulty areas on the volume. The INITIALIZE command marks the areas as allocated so that no data is written in them.

You can specify one or more faulty areas using either or both of the following formats. If you specify more than one area, separate specifications with a comma and enclose the list in parentheses.

Faulty Area	Purpose
lbn[:count]	Specifies a logical block number on the disk volume and, optionally, a count of logical blocks beginning with the allocation of the logical block specified to be marked.
sector.track.cyl[:count]	Specifies a specific sector, track, and cylinder on the disk volume and, optionally, a count of blocks beginning with the first block specified to be marked allocated.

All media supplied by DIGITAL and supported on VAX-11 RSX except floppy disks, TU58 cartridges, and RP04/5/6 disk packs are factory-formatted and contain bad block data. The Bad Block Locator Utility (BAD) or the diagnostic formatter ESRAC may be used to refresh the bad block data or construct it for the media exceptions just given. Use of the /BADBLOCKS keyword is necessary only to enter bad blocks that are not identified in the bad block data for that volume.

/CLUSTER_SIZE=n

Defines the minimum allocation unit in blocks. The maximum size you can specify for a volume is 1/100 the size of the volume. The minimum size you can specify is calculated by using the formula:

```
disk size (blocks)
------------------
  255 * 4096
```

The default cluster size for a Files–11 Structure Level 2 disk depends on the disk capacity. For disks that are 50,000 blocks or larger, the default cluster size is 3. Disks that are smaller than 50,000 blocks have a default value of 1.

Files–11 Structure Level 1 disks always have a cluster size value of 1.

/DATA_CHECK[=(option[,...])] /NODATA_CHECK

Defines a default for data check operations following all read and/or write operations on the volume. You can specify either or both of the following options:

Option	Function
READ	Performs data checks after all read operations
WRITE	Performs data checks after all write operations

If you specify /DATA_CHECK without specifying an option, the system assumes /DATA_CHECK=WRITE. By default, the system performs no data checking. You can override the checking you specify at initialization when you issue a MOUNT command for the volume.

You cannot specify /DATA_CHECK for Structure Level 1 volumes.

/DIRECTORIES=n

Specifies the number of entries to preallocate for user directories. The maximum allowable value of n is 16,000. If you do not specify /DIRECTORIES, the INITIALIZE command uses a value of 16 by default.

/EXTENSION=n

Specifies the number of blocks to use as a default extension size for all files on the volume. The default extension size is used when a file being updated increases to a size greater than its initial allocation. The maximum allowable value of n is 65,535.

If you do not specify a default extension size, the INITIALIZE command uses a value of 5.

/FILE_PROTECTION=code

Defines the default file protection to be applied to all files on the volume. Specify the code according to the standard syntax rules for protection, as described in Section 2.

This default protection is not used when the volume is being used on a VAX/VMS system. It is used on RSX–11 systems only. VAX–11 RSX uses the user default file protection specified in your user authorization file.

/GROUP

Defines a disk volume as a group volume. The owner UIC of the volume defaults to the group number of the user issuing the command and a member number of 0.

The /GROUP keyword establishes the volume protection as RWED for system, owner, and group.

/HEADERS=n

Specifies the number of file headers to be allocated initially for the index file. By default, the INITIALIZE command allocates 16 file headers.

The minimum allowable value of n is 16. The /MAXIMUM_FILES keyword specifies the maximum.

/HIGHWATER
/NOHIGHWATER

Controls highwater-marking on the specified volume. This keyword is applicable to Files–11 Structure Level 2 disk volumes only.

If you specify /HIGHWATER, INITIALIZE sets the File Highwater Mark (FHM) volume attribute. FHM is a mechanism that guarantees that a user cannot read data that he has not written.

The default is /HIGHWATER. The /NOHIGHWATER keyword disables FHM for the volume.

/INDEX=position

Requests that the index file for the volume's directory structure be placed in a specific location on the volume. You can specify one of the following options:

Option	Function
BEGINNING	Places the index file at the beginning of the volume
MIDDLE	Places the index file in the middle of the volume
END	Places the index file at the end of the volume
BLOCK:n	Places the index file at the beginning of the logical block specified

By default, the INITIALIZE command places the index file in the middle of the volume.

/MAXIMUM_FILES=n

Restricts the maximum number of files that the volume can contain that can override the default value. The default is calculated from the volume size in blocks as follows:

```
   volume-size
-------------------
(cluster factor +1) * 2)
```

The maximum size you can specify for any volume is determined by the following formula:

```
   volume-size
-------------------
(cluster factor +1)
```

The minimum value you can specify is 0. However, note that once a value for the maximum number of files is specified, the only way to reset it is to reinitialize the volume.

/SHARE
/NOSHARE

Controls whether a disk volume is shareable. The protection code for the volume defaults to all types of access for all categories of users. If you specify /NOSHARE, the protection code defaults to no access for group or world.

/STRUCTURE=level

Indicates the structure level of the volume. If you do not specify /STRUCTURE=1, the INITIALIZE command initializes the volume as a Files–11 Structure Level 2 volume by default.

If you specify /STRUCTURE=1, you cannot specify /CLUSTER_SIZE or DATA_CHECK.

Files–11 Structure Level 1 is the disk structure that is compatible with RSX–11.

/SYSTEM

Defines a disk volume as a system volume. The owner UIC of the volume defaults to [1,1] and default protection allows all types of access to the volume for all users.

No user privilege is required to use the /SYSTEM keyword. However, only users with system UICs can create directories on system volumes.

/USER_NAME=string

Specifies a user name from 1 through 12 characters to be recorded on the volume. If not specified, the INITIALIZE command uses the user name under which you logged in.

/VERIFIED
/NOVERIFIED

Indicates whether the disk has bad block data on it. The INITIALIZE command assumes that disks contain bad block data and uses the data to mark the bad blocks as allocated. Use /NOVERIFIED to request INITIALIZE to ignore bad block data on the disk.

/WINDOWS=n

Specifies the number of mapping pointers to be allocated for the file windows. When a file is opened, the file system uses the mapping pointers to access data in the file. The default number of pointers is 7; this also is the minimum allowable value of n. The maximum value of n is 80.

keywords applicable only to tape

/DENSITY=density-value

Specifies the density in bits per inch (bpi) at which the tape is to be written. You can specify a density of 800, 1600, or 6250.

If you do not specify /DENSITY for a blank tape, the system uses a default density of 1600 bpi. If you do not specify a density for a previously written tape, the system uses the density at which the tape was last written.

/LABEL=option

Defines characteristics for the magnetic tape volume label, as directed by the included option. The available options are as follows:

- OWNER_IDENTIFIER:"(14 ANSI characters)"

  Allows you to specify the owner identifier field in the VOL1 label. The field specified can accept up to 14 ANSI characters.

- VOLUME_ACCESSIBILITY:"character"

  Specifies the character to be written in the volume accessibility field of the VAX/VMS ANSI volume label VOL1 on an ANSI tape. The character may be any valid ANSI "a" character. This set of characters includes numeric characters, uppercase letters, and any one of the following nonalphanumeric characters:

  ```
  !  " % ' ( ) * + , - . / :  ;  < = > ?
  ```

The default is the space character.

If you specify any character other than a space, you must specify the /OVERRIDE=ACCESSIBILITY keyword on the INITIALIZE and MOUNT commands to access the tape.

/OVERRIDE=(option[,...])

Requests that the accessibility or the expiration date specified for the tape be overridden. One or both of the following keywords must be specified as an option:

Option	Function
ACCESSIBILITY	Indicates that the accessibility specified in nonblank header 1 and volume 1 labels of the tape is to be overridden; VAX-11 RSX never writes in these fields
EXPIRATION	Requests the INITIALIZE command to ignore the expiration date on a tape volume; the date is indicated by the expiration date of the first file on the volume

To initialize a tape that has not reached its expiration date or that has a nonblank accessibility field, you must be the owner of a tape volume (your UIC must match the UIC written on the volume) or you must have the user privilege VOLPRO to override volume protection.

If you specify both keywords, separate each by a comma and enclose them in parentheses.

EXAMPLES

1

```
> INITIALIZE/STRUCTURE=1 DBB2:MYVOL
```

This command initializes the disk volume on device DBB2 as a Structure Level 1 volume.

```
2    > @CMDFIL.CMD
         .
         .
         .
        INI DB2:MYVOL
         .
         .
         .
```

In this sequence, the indirect command file CMDFIL.CMD contains a command to initialize a volume. This command is in a format compatible with the RSX–11 MCR command INITVOLUME.

LOGIN PROCEDURE

VAX–11 RSX does not have a HELLO command. Instead, you gain the attention of the login procedure by pressing CTRL/C, CTRL/Y, or RETURN. Login prompts for your user name and password. The login procedure is equivalent to the RSX–11 MCR HELLO command in that it verifies your right to use the system.

FORMAT

< CTRL/C> or < CTRL/Y> or < RET>

Command Keywords	**Defaults**
/CLI=command-interpreter	*See text*
/[NO]COMMAND[=file-spec]	*/COMMAND*
/DISK=device-name[:]	*See text*
/TABLES=(command-table[,...])	*See text*

restrictions

None.

prompts

Username:	user-name[keyword,...]
Password:	password

command parameters

None

DESCRIPTION

When you log in with MCR, MCR uses the login file specified in your user authorization file entry or searches your default directory for a file named LOGIN.CMD. If one is present, MCR executes it as an indirect command file.

VAX–11 RSX automatically maps references to SY0 to the device associated with SYS$DISK. SYS$DISK is initially set to the default device specified in your UAF entry.

You must specify the optional keywords immediately after you enter your user name.

The login procedure performs the following functions:

- Validates your right to access the system by checking your user name and password against the entries in the system's user authorization file
- Establishes the default characteristics of your terminal session based on your user name entry in the authorization file
- Executes either the command procedure file named LOGIN.COM if one exists in your default directory or the command file, if any, defined in the user authorization file.

DIGITAL recommends that you use RETURN if you are logged in from a terminal for which autobaud is set.

command keywords

/CLI=command-interpreter

Specifies the name of an alternate command interpreter. You can specify either the DCL or MCR command interpreter. If a command interpreter is not specified, your default command interpreter (MCR) is used.

/COMMAND[=file-spec]
/NOCOMMAND

Controls whether VAX–11 RSX executes your default login file or the specified command file. /COMMAND is assumed. Unless your user authorization file specifies a different login file, the MCR command interpreter executes LOGIN.CMD. DCL executes LOGIN.COM by default. Both command interpreters search your default directory.

/DISK=device-name[:]

Specifies the name of a disk device to be associated with SYS$DISK for the terminal session. If you do not specify /DISK, the default SYS$DISK named in your authorization file is used.

/TABLES=(command-table[,...])

Specifies the name of an alternate CLI table to override the default listed in the user authorization file. If the /CLI keyword is set to DCL or MCR, the /TABLES keyword defaults to the correct value.

EXAMPLE

```
CTRL/Y INTERRUPT
Username:  BARRETT/CLI=MCR
Password:
           Welcome to VAX/VMS Version 4.00
           Last interactive login at 16-NOV-84 09:16:47.08
           Last non-interactive login at 15-NOV-84 17:32:34.27
>
```

Pressing CTRL/Y gets the attention of login, which prompts for the user name. The user name is entered, followed by keywords requesting MCR. After validating the user name, login prompts for the password.

LOGOUT

The LOGOUT command terminates a batch or an interactive terminal session.

FORMAT

LOGOUT

Command Keywords	**Defaults**
/BRIEF	*/BRIEF*
/FULL	*/BRIEF*
/[NO]HANGUP	*See text*

restrictions

RSX-11 MCR does not support the LOGOUT command.

prompts

None.

DESCRIPTION

The system displays a termination message and performs any necessary cleanup operations, such as terminating the current task image if one exists, dismounting any private volumes that remain mounted, and deallocating devices. Finally, it deletes your process and subprocesses, if any.

When you log out from an interactive terminal session, the brief form of the logout message appears. When an indirect command file (batch job) logs out, it writes the full form of the logout message to the log file.

If you turn the power off at your terminal without using the LOGOUT command from a direct line, you remain logged in.

Remember that if you have been using the SET HOST command to log in on remote processors, multiple LOGOUT commands might be necessary to end the terminal session.

The BYE command is supported by both VAX-11 RSX MCR and RSX-11 MCR.

command keywords

/BRIEF

Requests the brief form of the logout message. The command interpreter displays your user name and the date and time when you logged out. The default for an interactive session is /BRIEF.

/FULL

Requests the long form of the logout message. When you specify /FULL, the command interpreter displays a summary of accounting information for the terminal session. The default for a batch job is /FULL.

/HANGUP
/NOHANGUP

For dialup terminals, determines whether the phone will hang up whenever you log out. By default, the phone is not disconnected when you log out on a terminal that is connected to the host over a dialup line.

EXAMPLES

1

```
> LOGOUT
  HARVEY   logged out at 27-JUL-1984 17:48:56.73
```

This example shows what appears at the terminal when you end an interactive terminal session with a LOGOUT command.

2

```
FIELD   logged out at 27-JUL-1984 14:23:45.30
Accounting information:
Buffered I/O count:       31     Peak working set size: 100
Direct I/O count:          9     Peak virtual size:     300
Page faults:              66     Mounted volumes:         2
Elapsed CPU time: 0 00:00:00.33  Elapsed time:   0 00:00:13.27
```

When an indirect command file logs out or when you specify the /FULL keyword, the system generates a summary of accounting statistics as shown in this example.

MAIL

Invokes the VAX/VMS Personal Mail Utility (MAIL), which is used to send messages to other users of the system. For a complete description of the VAX/VMS Personal Mail Utility, including information about the MAIL command and its keyword, see the *VAX/VMS Utilities Reference Volume*.

FORMAT

MAIL *[file-spec] [recipient-name]*

Command Keyword	**Default**
/SUBJECT="text"	*None*
/EDIT=[(send,reply=extract,forward)	*/NOEDIT*
/SELF	*NOSELF*

MERGE

The MERGE command invokes the VAX/VMS Sort Utility to combine two through ten similarly sorted input files and create a single output file. Note that input files to be merged must be in sorted order. This description provides a functional overview of the MERGE command. For a complete functional description of the Sort Utility, including more information about the MERGE command, see the *VAX/VMS Utilities Reference Volume*.

FORMAT

MERGE *input-file-spec1,input-file-spec2[,...]*
output-file-spec

Command Keywords	**Defaults**
/[NO]CHECK_SEQUENCE	*/CHECK_SEQUENCE*
/COLLATING_SEQUENCE=sequence	*/COLLATING_SEQUENCE=ASCII*
/[NO]DUPLICATES	*/DUPLICATES*
/KEY=(field[,...])	*See text*
/SPECIFICATION[=file-spec]	*See text*
/[NO]STABLE	*/NOSTABLE*
/[NO]STATISTICS	*/NOSTATISTICS*

Input File Keyword	**Default**
/FORMAT=(file-attribute[,...])	*None*

Output File Keywords	**Defaults**
/ALLOCATION=n	*None*
/BUCKET_SIZE=n	
/CONTIGUOUS	
/FORMAT=(record-format[,...])	
/INDEXED_SEQUENTIAL	
/OVERLAY	
/RELATIVE	
/SEQUENTIAL	

restrictions

None.

prompts

_Input: input-file-spec1,input-file-spec2[,...]
_Output: output-file-spec

command parameters

input-file-spec1,input-file-spec2[,...]

Specifies the names of the sorted files whose records are to be merged. At least 2 file specifications, but not more than 10, must be specified and separated by commas. The keys must be the same in all files.

output-file-spec

Specifies the name of the merged file to be created. Only one output file specification may be specified. No wildcard characters are allowed in the file specification.

command keywords

/CHECK_SEQUENCE
/NOCHECK_SEQUENCE

Examines the input files to be merged to ensure they are in order. If a record is out of order, MERGE gives the following message at the end of the merge operation:

```
%SOR-E-BAD_ORDER, Input file [n] is out of order.
```

/COLLATING_SEQUENCE=sequence

Determines which collating sequence (ASCII, EBCDIC, or MULTINATIONAL) is used when merging records. MERGE arranges characters in ASCII sequence by default.

If the EBCDIC option is selected, the input files are merged as if the EBCDIC key characters were translated into ASCII key characters and then merged as an ASCII key. The records do not change.

If the MULTINATIONAL option is selected, the input files are merged according to the MULTINATIONAL sequence.

/DUPLICATES
/NODUPLICATES

Eliminates all but one of a record with equal keys. Because the record retained is unpredictable, you must specify which duplicate record to keep by invoking MERGE at the program level.

The /STABLE and the /NODUPLICATES keywords are mutually exclusive.

/KEY=(field[,...])

Defines a merge key, including position, size, order, and data type. If the key field starts in the first position, encompasses the entire record, has been sorted in ascending order, and contains character data, the /KEY keyword need not be specified.

The key field is defined by options that must be separated with commas and enclosed in parentheses.

<table>
<tr><th>Required Option</th><th>Function</th></tr>
<tr><td>POSITION:n</td><td>Specifies the position of the key within each record, where the first character of each record is position 1.</td></tr>
<tr><td>SIZE:n</td><td>Specifies the length of the merge key in characters, bytes, or digits, depending on the key field data type. The valid sizes, based on data types, are:
<table>
<tr><th>Data Type</th><th>Values for n</th></tr>
<tr><td>character</td><td>1–32767</td></tr>
<tr><td>binary</td><td>1, 2, 4, 8, 16</td></tr>
<tr><td>any decimal data type</td><td>1–31</td></tr>
</table>
The total of all key field sizes must be less than 32,767 bytes.</td></tr>
</table>

Elective Option	Function
NUMBER:n	Specifies the precedence of the merge key being defined, where 1 represents the primary merge key, 2 represents the secondary merge key, and so on. If this option is not specified on the first /KEY keyword, NUMBER:1 is assumed; if not specified on any subsequent /KEY keywords, the default value is the number assigned to the previous key, plus 1. The legal values are 1 through 255.
BINARY CHARACTER DECIMAL	Indicates the type of data appearing in the merge key field. If not specified, MERGE assumes that the data type is CHARACTER.
SIGNED UNSIGNED	Indicates whether the key data type is signed or unsigned binary data.
LEADING_SIGN TRAILING_SIGN	Indicates whether the sign of a decimal data type key appears at the beginning or end of the key. If the key data type is specified as DECIMAL and neither of these options is specified, the default is TRAILING_SIGN.
OVERPUNCHED_SIGN SEPARATE_SIGN	Indicates whether the sign of a decimal data type key is superimposed on the decimal value or is separated from the decimal value. If the key data type is specified as DECIMAL and neither of these options is specified, the default is OVERPUNCHED_SIGN.
ASCENDING DESCENDING	Indicates whether the key is to be merged into ascending or descending order. If neither of these options is specified, the default is ASCENDING.

/SPECIFICATION[=file-spec]

Indicates that the command and file keywords, including key field definitions, are contained in a specification file. If no file specification is included, MERGE reads the specification file from SYS$INPUT.

The format and contents of the specification file are described in detail in the *VAX/VMS Utilities Reference Volume.*

No wildcard characters are allowed in the file specification.

/STABLE /NOSTABLE

Specifies the order in which MERGE places records with equal keys. The order is determined by the order of the file specifications on the command line. If the command is MERGE/STABLE X,Y and a record from X.DAT and a record from Y.DAT have equal keys, the record from X.DAT will be written to the output file before the record from Y.DAT.

If /STABLE is not specified, the order of records with equal keys is unpredictable.

/STATISTICS /NOSTATISTICS

Specifies that a statistical summary is to be displayed after the MERGE operation has completed. By default, no statistical summary is displayed.

input file keywords

/FORMAT=(file-attribute[,...])

Specifies attributes of the input file to override the existing data that MERGE normally obtains through VAX RMS. One or both of the following options can be specified.

Option	Function
RECORD_SIZE:n	Specifies, in bytes, the length of the longest record, overriding the record size defined in the file header or label. Specifies a record size for an input file when the input file is not on disk or when the longest record size is known to be inaccurate. The longest record length that can be specified is 32,767 bytes for sequential files, 16,383 bytes for relative files, and 16,362 bytes for indexed sequential files.
FILE_SIZE:n	Defines the size of the file in blocks. Specifies a file size for a file that is not on disk. If you do not specify a file size for an input file, MERGE uses the default value of 1000 blocks. The maximum file size that can be specified is 4,294,967,295 blocks.

output file keywords

/ALLOCATION=n

Specifies the number of 512-byte blocks to be allocated for the output file. By default, MERGE allocates blocks based on the number of records sorted.

The number of blocks specified can be in the range of 1 through 4,294,967,295.

/BUCKET_SIZE=n

Specifies the number of 512-byte blocks per bucket for the output file. The maximum size that can be specified is 32 blocks.

If you do not specify a bucket size, the bucket size of the output file is the same as that of the input file.

/CONTIGUOUS

Controls whether the allocation of disk space for the output file is to be contiguous. If you specify /CONTIGUOUS, you must also specify /ALLOCATION to define the number of blocks to allocate for the output file.

By default, MERGE output is not contiguous.

/FORMAT=(record-format[,...])

Defines the output file record format. You can specify one or more of the following options:

Option	Function
CONTROLLED[:n] FIXED[:n] VARIABLE[:n]	Defines the output file record format and length, where n is the length of the longest record in the output file. If n is not specified, it defaults to the length of the longest record in the input file. The maximum record size that can be specified is 16,383 bytes (less any control bytes).
SIZE:n	Specifies the size in bytes of the fixed portion of the controlled record when CONTROLLED is specified. The maximum size of the fixed-length control field that can be specified is 255 bytes. If CONTROLLED is specified, and no size is specified, MERGE uses the default value of 2 bytes.
BLOCK_SIZE:n	Specifies the block size in bytes when the output file is directed to a magnetic tape. By default, MERGE uses the block size of the input file if the input file is a tape file. If the input file is not a tape file, MERGE uses, by default, the block size specified when the output tape volume was mounted. You can specify a block size in the range of 20 through 65,535. Note, however, that to ensure compatible interchange with most non-DIGITAL systems, the block size should be less than 2048 bytes.

If you do not specify /FORMAT to define the record format of the output file, MERGE assumes a default output format based on the process selected.

/INDEXED_SEQUENTIAL

Specifies that the output file is in indexed sequential order. If you specify /INDEXED_SEQUENTIAL, the output file must exist, its organization must be indexed, and it must be empty. Therefore, you must also specify /OVERLAY to overlay the existing file.

By default, a record or tag sort creates an output file that has the same organization as the input file. You must specify INDEXED_SEQUENTIAL to create an indexed sequential output file from a sequential relative input file.

/OVERLAY

Indicates that the existing file is to be overlaid with the sorted records of the input file.

By default, a new output file is created and does not overlay an existing file.

/RELATIVE

Specifies that the output file is in relative file organization.

By default, a record or tag sort results in an output file that has the same organization as the input file. Use the /RELATIVE keyword to create a relative output file from a sequential or indexed sequential input file.

/SEQUENTIAL

Specifies that the format of the output file has sequential organization. This is the default for an address or index sort process; for a record or tag sort process, the output file format defaults to the organization of the input file.

Specify /SEQUENTIAL to create an output file that is in a different format from the input file.

EXAMPLES

1

```
> MERGE /KEY=(POSITION:1,SIZE:80)/CHECK_SEQUENCE-
>_[CUST.WORK]CUST01.DAT,-
>_[CUST.WORK]CUST02.DAT, CUSTMERGE.DAT
```

Two identically sorted disk files in one directory are merged into another file in the current default directory. The /CHECK_SEQUENCE input file keyword is used to ensure that the files are sorted in the correct order. Note that the input files are not in the current default directory, so they have the directory included as part of their file specifications.

2

```
> MERGE /KEY=(DECIMAL,TRAILING,SEPARATE,POSITION:1,SIZE:6,)-
>_DESCENDING/KEY=(DECIMAL,LEADING,SEPARATE,POSITION:30,SIZE:2)-
>_LICENSE80.DAT,LICENSE81.DAT/CHECK_SEQUENCE LICENSE.DAT
```

Two files are merged. Both files have been sorted with the first key in descending order and the second key sorted in ascending order. All files are in the current default directory.

3

```
> MERGE/KEY=(PO:1,SI:30) TRNTO:[PGM]FILE1.SRT,FILE2.SRT-
>_MERGEFILE.DAT
```

This MERGE command merges two input files on the directory [PGM] at the remote node TRNTO into the output file, MERGEFILE.DAT, at the local node.

MOUNT

The MOUNT command makes a disk or magnetic tape volume or volume set and the data it contains available for processing by system commands or user programs. For additional information on the MOUNT command, refer to the description of the Mount Utility in the *VAX/VMS Utilities Reference Volume*.

FORMAT

MOUNT *device-name[:][,...] [volume-label[,...]] [logical-name[:]]*

Command Keywords	**Defaults**
/ACCESSED=n (D)	*See text*
/[NO]ASSIST	*/ASSIST*
/AUTOMATIC (T)	*/AUTOMATIC*
/BIND=volume-set-name (D)	
/BLOCKSIZE=n (T)	*/BLOCKSIZE=2048*
/[NO]CACHE=(option[,...]) (D)	
/COMMENT="string"	
/DATA_CHECK[=(option,[...])]	*See text*
/DENSITY=n (T)	*See text*
/EXTENSION=n (D)	
/FOREIGN	*See text*
/GROUP (D)	
/[NO]HDR3 (T)	*/HDR3*
/[NO]LABEL (T)	*/LABEL*
/[NO]MESSAGE	*/MESSAGE*
/[NO]MOUNT_VERIFICATION (D)	*/MOUNT_VERIFICATION*
/OVERRIDE=(option[,...])	
/OWNER_UIC=uic	
/PROCESSOR=option	
/PROTECTION=code	
/[NO]QUOTA (D)	*/QUOTA*
/RECORDSIZE=n (T)	*See text*
/[NO]SHARE (D)	*/NOSHARE*
/SYSTEM (D)	
/[NO]UNLOAD	*/UNLOAD*
/WINDOWS=n (D)	
/[NO]WRITE	*/WRITE*

D = applicable to disk only.

T = applicable to tape only.

Keyword descriptions are categorized according to the descriptions that follow.

restrictions

- VAX–11 RSX and RSX–11 MCR do not support the same keywords for the MOUNT command.
- You cannot allocate a device interactively and then mount the device from an indirect command procedure. However, you can allocate and mount the device either interactively or from a command procedure.

prompts

_Device:	device-name[:][,...]
_Label:	volume-label[,...]
_Log name:	logical-name[:]

command parameters

device-name[:][,...]

Specifies the physical device name or logical name of the device on which the volume is to be mounted.

When mounting a volume set, you can specify more than one device name. Separate the device names with either a comma (,) or plus sign (+). If you are mounting tape volumes, you can specify more volume labels than device names.

If you specify a colon (:) following the device name, the command strips it.

volume-label[,...]

Specifies the alphanumeric label written on the volume when it was initialized. For Files–11 Structure Levels 1 and 2 disk volumes, the volume label can have up to 12 characters; for tape volumes, it can have up to 6 characters.

If you specify more than one volume label, separate the labels with a comma (,) or plus sign (+). The volumes must be in the same volume set and the labels must be specified in the correct order.

The volume-label parameter is not required when you mount a volume with the /FOREIGN keyword, nor when you specify /OVERRIDE=IDENTIFICATION. To specify a logical name when you enter either of these keywords, type any alphanumeric characters in the volume label parameter position.

logical-name[:]

Defines a 1- to 255-character alphanumeric string that is the logical name to be associated with the device(s). The logical name subsequently can be used to refer to the volume or volume set in commands and programs.

If you do not specify a logical name, the MOUNT command assigns the default logical name DISK$volume-label for disk devices or TAPE$volume-label for tape devices.

The MOUNT command places the logical name in the process logical name table unless /GROUP or /SYSTEM is requested. In the latter cases, the name is placed in the group or system logical name tables, respectively.

If you specify multiple devices and a logical name, the logical name applies to all devices specified.

DESCRIPTION

When you issue the MOUNT command, the MOUNT command checks:

- That the device has not been allocated to another user
- That a volume is physically loaded on the device specified
- That the label on the volume matches the label specified
- For tapes, that the volume accessibility field of the VOL1 label is valid

Mounting volumes with the /SHARE, /GROUP, or /SYSTEM keyword deallocates the device.

If you mount a device in a subprocess, the device will be owned by the main process. This allows the main process and all subprocesses in your job to access the device. When the subprocess exits, the device remains mounted and owned by the main process.

VAX–11 RSX MCR syntax for the MOUNT command accepts either a colon (:) or a colon (:) and a space between the device name and the volume label.

The form of the MOUNT command that is compatible between RSX–11 and VAX–11 RSX MCR is the following:

```
MOU llnn:volume-label
```

where

llnn is an RSX–11 physical device name that has been assigned as the logical name for a VAX/VMS physical device unit

volume-label is the volume label

keywords applicable to disk and tape

/ASSIST
/NOASSIST

Controls whether to allow operator or user intervention should a MOUNT request fail.

If you specify /ASSIST or omit this keyword, the user and certain classes of operators are notified if a failure occurs during the MOUNT operation. If a failure occurs, the operator or user then has the choice of either aborting the mount or correcting the error condition and allowing the mount to continue.

The operator ASSIST messages are sent to all operators enabled to receive DEVICE and TAPE-and-DISK messages. Thus, if an operator's assistance is needed while mounting a disk device, a message is sent to both device and disk operators. See the DCL command REPLY (in the *VAX/VMS DCL Dictionary*) for more information on enabling and disabling operator terminals.

Any operator reply to a MOUNT request is written to SYS$OUTPUT to be displayed on the user's terminal or written in a batch job log.

If no operator is enabled to receive and respond to a MOUNT request, a message is displayed informing the user of the situation. If the MOUNT request originated from a batch job, the MOUNT is aborted since there is no way to correct the error condition.

/COMMENT="string"

Includes the specified string with the operator request if the mount operation requires operator assistance. It is used with the /ASSIST keyword. The text string is output to the log file and the current SYS$OUTPUT device. Unless you enclose the string in quotation marks ("), characters are converted to uppercase and space and tab characters are removed.

/DATA_CHECK[=(options,...)]

Specifies that all read and write operations to the volume are to be followed by a write-check operation. The write-check compares the data on disk with that in memory to ensure that the transfer was successful. You can specify either or both of the following options:

Option	Function
READ	Performs checks following all read operations
WRITE	Performs checks following all write operations

If you specify /DATA_CHECK without including an option, the default of /DATA_CHECK=WRITE is used.

/FOREIGN

Indicates that the volume is not in the standard ANSI format used by VAX/VMS. If you do not specify /FOREIGN, the volume is assumed to be in Files–11 format.

If you specify /FOREIGN, the program that reads the volume must be able to process the labels on the volume, if any. For example, you must mount a DOS-formatted tape with the /FOREIGN keyword and use FLX to transfer it.

The default protection applied to foreign volumes is read and write for system and owner. If you specify /GROUP in addition, group members are given read and write access. If you specify /SYSTEM or /SHARE, group and world are both given read and write access. Any user category that has read and write access to a foreign volume also is allowed logical I/O and physical I/O access to the volume.

To protect a foreign volume, you must specify the /PROTECTION keyword. If the volume is currently in Files–11 format, you must have override volume protection (VOLPRO) privilege to mount it as foreign or you must be the volume owner (same UIC).

/MESSAGE
/NOMESSAGE

Controls whether MOUNT request messages are output to your current SYS$OUTPUT device. Note that if you specify /NOMESSAGE during an operator-assisted mount, messages are not output to SYS$OUTPUT, but the operator sees them.

/OVERRIDE=(option[,...])

Inhibits one or more of the following protection checks that the MOUNT command performs:

Option	Function
ACCESSIBILITY	(For tapes only.) Allows you to override a nonblank VOL1 or HDR1 accessibility field. You must have the user privilege to override volume protection (VOLPRO) or be the owner of the volume.
EXPIRATION	(For tapes only.) Allows you to write a tape that has not yet reached its expiration date. You must have the user privilege to override volume protection (VOLPRO) or your UIC must match the UIC written on the volume.
IDENTIFICATION	Allows you to mount a volume when you do not know what the volume label is. If you specify /OVERRIDE=IDENTIFICATION, you can specify anything for the volume-label parameter or you can omit it; the MOUNT command ignores whatever you enter. The volume must be mounted /NOSHARE (either explicitly or by default). This functionality parallels that of the RSX–11 MCR keyword /OVR.
LOCK	Directs MOUNT not to write-lock the volume as a consequence of certain errors encountered while mounting it. Use this option when you are mounting a damaged volume to be repaired using the VAX/VMS Verify Utility (under DCL). VOLPRO privilege or ownership of the volume is required to use this option.
SETID	(For tapes that do not conform to ANSI standards.) Allows you to inhibit checks of the file set identifier when you switch reels in a multivolume tape set.

If you specify more than one option, separate them with commas and enclose the list in parentheses.

If you have VOLPRO privilege and you specify /OVERRIDE=(ACCESSIBILITY, EXPIRATION) along with the /FOREIGN keyword, the tape will not be read.

/OWNER_UIC=[uic]

Requests that the specified UIC be assigned ownership of the volume while it is mounted, thereby overriding the ownership recorded on the volume. If you are mounting a device /FOREIGN, /OWNER_UIC requests an owner UIC other than your current UIC.

The brackets are required.

You must have the privilege to override volume protection to use the /OWNER_UIC keyword or you must be the owner of the volume.

/PROCESSOR=option

Allows you to control whether the volume has its own Ancillary Control Process (ACP) process and which ACP image the process executes. The following options can be specified:

Option	Function
UNIQUE	Creates a new process to contain a copy of the default ACP image for the specified device type or controller
SAME:device	Requests the same ACP process that is currently being used by the specified device
file-spec	Creates a new process to contain the specified ACP image (for example, a modified ACP)

Operator privilege is required to use the /PROCESSOR keyword.

/PROTECTION=code

Specifies the protection code to be assigned to the volume. All access to any file on the volume must pass both the volume protection and the protection applied to the individual file.

You should specify the code according to the standard syntax rules for specifying protection described in Section 2.

If you do not specify protection, it defaults to that assigned when the volume was initialized.

The protection specified by this keyword takes precedence over the protection implied by other keywords, for example, /SYSTEM and /GROUP. The only exception is /NOWRITE.

/UNLOAD
/NOUNLOAD

Controls whether or not the disk or tape volume(s) specified in the MOUNT command is unloaded when it is dismounted. The default is /UNLOAD.

/WRITE
/NOWRITE

Control whether the volume can be written. By default, a volume is considered read/write. Specify /NOWRITE to provide read-only access to protect files.

/NOWRITE is equivalent to write-locking a drive.

keywords applicable only to disk

/ACCESSED=n

Specifies the approximate number of directories that will be in use concurrently on the volume. A value from 0 to 255 may be specified to override the default value that was specified when the volume was initialized. The value n is a decimal number.

Operator privilege is required to use /ACCESSED.

/BIND=volume-set-name

Creates or adds to a volume set that consists of one or more disks.

The volume-set-name specifies a 1- to 12-character name that identifies the volume set.

When a volume set is created, the volumes are specified on a volume-label list. Each volume is assigned a relative volume number based on its position in the label list. The first volume specified is called the root volume of the volume set.

When adding one or more volumes to a set, the first volume label specified must be the root volume unless the root volume already is online.

/CACHE=(option[,...]) /NOCACHE

Controls whether disk-caching limits established at system generation time will be disabled or overridden. One or more of the following limits may be altered by use of the appropriate option:

Option	Function
[NO]EXTENT[=n]	Enables or disables extent caching. You must have the operator (OPER) privilege and you must specify n, the number of entries in the extent cache. Note that NOEXTENT and EXTENT=0 both disable extent caching.
[NO]FILE_ID=n]	Enables or disables file-identification caching denoted by n, the number of entries given to be a value greater than 1. Note that both FILE_ID=1 and NOFILE_ID disable file-identification caching. Operator (OPER) privilege is required.
LIMIT=n	Specifies the maximum amount of free space in the extent cache in one-thousandths of the currently available free space on the disk.
[NO]QUOTA[=n]	Enables or disables the quota cache. Operator (OPER) privilege is required and you must specify n, the number of entries in the cache. Usually, n is set to the maximum number of active users expected on the disk when quotas are enabled. Both NOQUOTA and QUOTA=0 disable quota file caching.
WRITETHROUGH	Disables write-back caching, which writes only the file headers of files open for write when other files are closed. In comparison, if you specify the WRITETHROUGH option, file headers are written to the disk on every file-header operation.

When specifying more than one option, separate each by a comma and enclose the list in parentheses.

/NOCACHE disables all disk caching on a volume. Specifying /NOCACHE is equivalent to specifying /CACHE=(NOEXTENT,NOFILE_ID, NOQUOTA,WRITETHROUGH).

/EXTENSION=n

Specifies the number of blocks by which the file is extended. This number can be overridden by a command or program request. The minimum value of n is 0, the maximum is 65535. The value n is interpreted as a decimal number.

If you do not specify /EXTENSION, the extension value defaults to that specified when the volume was initialized.

/GROUP

Makes the volume available to any user with the same UIC group number as the user issuing the MOUNT command. These users are not required to issue a MOUNT command to gain access to the volume. However, volume protection is applied.

The logical name for the device is placed in the group logical name table. You must have the appropriate privilege to place a name in the group logical name table.

A volume mounted /GROUP remains mounted until it is explicitly dismounted. It is not automatically dismounted when its accessors log out of the system.

If you issue a MOUNT command for a volume that is already mounted /GROUP, you must use the /SHARE keyword. Any additional keywords are ignored.

/MOUNT_VERIFICATION /NOMOUNT_VERIFICATION

Controls whether the device is a candidate for mount verification.

/QUOTA /NOQUOTA

Controls whether disk quotas will be enforced on this disk volume. The /QUOTA keyword is the default, which enforces quotas for each user. /NOQUOTA inhibits quota checking. You must either own the volume or have the VOLPRO privilege to use this keyword.

/SHARE /NOSHARE

Indicates whether the volume is shareable. If the volume has already been mounted /SHARE by another user and you request that it be mounted with the /SHARE keyword, any other keywords you specify are ignored.

A volume mounted /SHARE is automatically dismounted when all its accessors log out of the system.

By default, the MOUNT command assumes that a device is not shareable and allocates it.

If you have previously allocated the device and then specify the /SHARE keyword, the device is deallocated.

/SYSTEM

Makes the volume available to all users of the system as long as the UIC-based volume protection allows them access. These users are not required to issue a MOUNT command to gain access to the volume.

The logical name for the device is placed in the system logical name table. You must have the appropriate privilege to place a name in the system logical name table.

A volume mounted /SYSTEM remains mounted until it is explicitly dismounted. It is not automatically dismounted when its accessors log out of the system.

If you issue a MOUNT command for a volume that is already mounted /SYSTEM, you must use the /SHARE keyword. Any additional keywords are ignored.

/WINDOWS=n

Specifies the number of retrieval pointers to be used in mapping data when the volume files are opened. If /WINDOWS is not specified, the number of pointers defaults to the value specified when the volume was initialized. The value n is interpreted as a decimal number.

Operator privilege is required to use the /WINDOW keyword.

keywords applicable only to tape

/AUTOMATIC
/NOAUTOMATIC

Determines whether MOUNT enables or disables automatic volume switching and labeling for magnetic tape.

If you specify /NOAUTOMATIC, the MTAACP requires operator intervention to switch to the next drive during end-of-tape processing. It requires that the operator specify a label for each new reel added to a volume set.

/BLOCK=n
/NOBLOCK

Sets the default block size to a specific number of bytes per block. If you specify /NOBLOCK, each block will contain one record.

By default, records are written to tape volumes in 2048-byte blocks unless /FOREIGN and /NOLABEL are specified. In that case, the default block size is 512 bytes.

/DENSITY=n

Specifies the density (in bpi) at which the tape is written for either foreign or unlabeled tapes. You can specify a density of 800, 1600, or 6250. To specify a specific density, either the /FOREIGN or /NOLABEL keyword must be specified and the first operation performed on the tape must be a write.

If you specify /LABEL or if the first operation on the tape is a read, the tape is read or written at the density at which the first records on the tape are recorded.

/HDR3
/NOHDR3

Controls whether ANSI HDR3 labels are written on magnetic tapes. ANSI HDR3 labels are written on tapes by default. For tapes that will be used on systems that do not tolerate HDR3 labels, specify /NOHDR3.

/LABEL
/NOLABEL

Indicates whether the tape contains standard labels.

If you mount a tape specifying the /NOLABEL keyword, an end-of-file condition is returned when a tape mark is encountered while reading the tape.

The default protection for unlabeled tapes is all access to the system and owner and no access to the group and world.

/RECORDSIZE=n

Specifies the number of bytes in each record. This keyword is normally used with the /FOREIGN or /BLOCKSIZE keywords to read and write fixed-length records on a block-structured device. In this case, the record size must be less than or equal to the default block size. The block size can range from 20 through 65,532 with VAX RMS or 18 through 65,534 without VAX RMS.

Use the /RECORDSIZE keyword whenever you are mounting tapes without HDR2 labels (such as RTM tapes). This provides VAX RMS with default values for both the maximum record size and the length of the largest record.

EXAMPLES

1

```
> ASN  DBA1:=DK0:
> @RODIN
  MOUNT DK0:MYVOL
 % MOUNT-I-MOUNTED, MYVOL mounted on _DBA1:
      .
      .
      .
```

The ASN command defines DBA1 as the equivalent name for the logical name DK0. When the indirect command file RODIN.CMD executes, its command to mount the volume labeled MYVOL on DK0 becomes, in effect, a request to mount MYVOL on DBA1.

2

```
> MOUNT DBA1 MYVOL DISK
%  MOUNT-I-MOUNTED, MYVOL mounted on _DBA1:
```

This command mounts the volume MYVOL on disk device DBA1 and assigns a logical name of DISK.

PRINT

The PRINT command queues one or more files for printing on either a system-defined printer or the device you specify. The system considers a file or files queued by a PRINT command to be a job. It assigns a unique job identification to each job in the system.

FORMAT

PRINT *file-spec[, . . .]*

Command Keywords	**Defaults**
/AFTER=time	
/CHARACTERISTICS=(c[, . . .])	*See text*
/DEVICE=queue-name[:]	*/DEVICE=SYS$PRINT*
/FORM=type	*/FORM=0*
/[NO]HOLD	*/NOHOLD*
/[NO]IDENTIFY	*/IDENTIFY*
/JOB_COUNT=n	*/JOB_COUNT=1*
/[NO]LOWERCASE	*/NOLOWERCASE*
/NAME=job-name	*See text*
/NOTE=string	
/[NO]NOTIFY	*/NONOTIFY*
/OPERATOR=string	
/PARAMETERS=(p[, . . .])	
/PRIORITY=n	*See text*
/QUEUE=queue-name[:]	*/QUEUE=SYS$PRINT*
/REMOTE	
/[NO]RESTART	*/RESTART*
/USER=username	*See text*

Positional Keywords	**Defaults**
/[NO]BURST[=option]	*See text*
/COPIES=n	*/COPIES=1*
/[NO]DELETE	*/NODELETE*
/[NO]FEED	*/FEED*
/[NO]FLAG[=option]	*See text*
/[NO]HEADER	*/NOHEADER*
/PAGES=([lowlim,]uplim)	*See text*
/[NO]PASSALL	*/NOPASSALL*
/SETUP=module[, . . .]	*See text*
/[NO]SPACE	*/NOSPACE*
/[NO]TRAILER[=option]	*See text*

restrictions

- If you include a node name in your file specification, you must use the /REMOTE keyword. If you specify /REMOTE, you cannot use any other keywords.
- Use of the PRINT command requires operator (OPER) privilege and execute (E) or write (W) access to the queue.

- Use of the /USER keyword requires CMKRNL privilege and read access to the system authorization file.

prompt

_File: file-spec[, . . .]

command parameter

file-spec[, . . .]

Provides the file specification of one or more files to be printed. If you specify multiple files, separate them with either a comma (,) or a plus sign (+).

You can use wildcard characters (*) in place of the directory, file name, type, or version fields of the file specification.

If you omit a file type, the PRINT command uses LIS by default.

DESCRIPTION

The PRINT command places the specified file(s) in a printer queue for printing. The files queued by a PRINT command are considered a job. The system assigns a unique job identification number to each job in the printer queue and displays this entry number when the PRINT command completes execution. (Completion of a PRINT command means that the job has been queued to the appropriate printer queue. It does not mean that the actual printing process is finished.)

Once a print job has been queued, the version of the file submitted is printed, even if a newer version of the file is created before the print job runs.

Printer queues are identified by name. If you do not specify a queue name with the /DEVICE or /QUEUE keywords, the system queues the job to SYS$PRINT. The PRINT command, by default, displays the name of the queue on which it entered the job.

You can delete a queue entry using the DELETE command with the /ENTRY keyword.

You can control the attributes of a print queue using the SET QUEUE command.

RSX–11M optionally supports the PRINT utility.

command keywords

/AFTER=time

Requests that the job be printed after a specific time of day. The time can be specified as an absolute time, delta time, or combination absolute and delta time (see Section 2).

For example, /AFT=17 places a job in a hold status until 5:00 PM. If it is after 5:00, the job is printed immediately.

/CHARACTERISTICS=(c[, . . .])

Specifies one or more characteristics desired for printing the file(s). If you specify only one characteristic, you can omit the parentheses. Codes for characteristics can be either keywords or digits and are defined on installation. (See the system manager for the list of characteristics available on your system.) Characteristics can refer to such things as ink color.

A print job can execute on a printer queue only if each characteristic specified with the PRINT command is also specified for that particular printer queue. If you specify a characteristic that is not specified for that particular printer queue, the job is placed in a hold status. (For your job to print, the system manager should stop the queue, physically change the characteristic(s) of the printer, and restart the queue, specifying the new values listed in the /CHARACTERISTICS keyword.)

Specification of a characteristic for a printer queue does not prevent jobs that do not specify that characteristic from being executed.

Use the SHOW QUEUE/CHARACTERISTICS command to find out the characteristics of the printer queue.

/DEVICE=queue-name[:]

Requests that the files specified be queued to a particular device. If you do not specify /DEVICE, files are queued to the current default printer. The default device is SYS$PRINT. The /DEVICE keyword is synonymous with the /QUEUE keyword.

/FORM=type

Specifies the form type required for the files to be printed. You can specify the form type using a numeric value or an alphanumeric code. Codes for form types are defined when the system is installed. You can use the SHOW QUEUE/FORM command to find out the types of forms available for your system.

If you specify a form type different from that of the queue, your job is placed in a hold status until the form type of the queue is set equal to the form type of the job. (For your job to print, the system manager should stop the queue, physically change the form type(s) of the printer, and restart the queue specifying the new value listed in the /FORM keyword.)

/HOLD
/NOHOLD

Controls whether the file is available for printing immediately. The file cannot be released for printing until you issue the SET QUEUE command to release it.

/IDENTIFY
/NOIDENTIFY

Controls whether the PRINT command displays a message indicating the job number of the print job and the name of the queue to which it is entered.

By default, the PRINT command displays this information when a job is queued successfully. The display is made on the current input device.

/JOB_COUNT=n

Requests that the entire job be printed n times. The value of n is a decimal number ranging from 1 through 255. If you omit this keyword, one copy of the job is printed.

/LOWERCASE
/NOLOWERCASE

Indicates whether the specified file(s) contain(s) lowercase alphabetic letters and must be printed on a printer that has uppercase and lowercase capabilities.

The default is /NOLOWERCASE, which means that files can be printed on printers supporting only uppercase letters.

/NAME=job-name

Defines a one- to eight-character alphanumeric string to identify the job. The name string is displayed in response to the SHOW QUEUE command.

If /NAME is not specified, the name string defaults to the file name of the first, or only, file in the job.

/NOTE=string

Allows you to specify a message to appear on the flag page for the job. The string can contain up to 255 characters.

/NOTIFY
/NONOTIFY

Controls whether a message is broadcast to any terminal you are logged in to, notifying you when your print job has been completed or aborted.

/OPERATOR=string

Allows you to specify a message to be sent to the operator. The string can contain up to 255 characters.

As soon as the system processes the PRINT/OPERATOR command, the message is transmitted to the operator account.

/PARAMETERS=(p[, . . .])

Specifies from one to eight optional parameters to be passed to the job.

If you specify only one parameter, you can omit the parentheses. The commas delimit individual parameters. To specify a parameter that contains any special characters or delimiters, enclose the parameter in quotation marks.

Each parameter can have as many as 255 characters.

/PRIORITY=n

Specifies the priority of the print job. The value of n must be in the range of 0 through 255; 0 is the lowest priority and 255 is the highest.

By default, jobs are assigned the same priority as your process. You can assign lower priorities to your nonessential jobs so that jobs you want sooner will be processed first.

You must use the user privilege OPER to set a priority value that is higher than your process's priority.

/QUEUE=queue-name[:]

Requests that the file(s) specified be queued to a particular device. The default is SYS$PRINT.

This keyword performs the same function as /DEVICE.

/REMOTE

Indicates that the specified files be printed on a remote node. Because the file to be printed must exist on the remote node, the file specification must contain the name of the remote node. If this keyword is specified, the file is queued for printing to SYS$PRINT of the node on which the file resides. See also the appropriate DECnet documentation.

If you specify /REMOTE, you cannot specify any other keywords.

/RESTART
/NORESTART

Specifies whether the job can restart after a crash or a STOP/REQUEUE command. The default is /RESTART.

/USER=username

Allows you to submit a print job for another user. The print job will run exactly as if that user had submitted it. The job runs under that user's privileges, UIC, and process priority. Accounting information is logged to that user's account. By default, the user identification comes from the requesting process.

The user name can be any name that is valid on your system.

To use this keyword, you must have the user privilege CMKRNL and read access to the system authorization file.

positional keywords

/BURST[=option]
/NOBURST

Indicates whether a burst page is to precede the file. A burst page is printed before a flag page and contains the same information. However, it is printed over the perforation between pages to make the separation of files easier.

When you specify /BURST, you need not specify /FLAG. A flag page will automatically follow the burst page.

The /BURST keyword can take either of two options: ALL or ONE. The ALL option indicates that each file to be printed will be preceded by a burst page and a flag page. The ONE option indicates that a burst page will apply only to the first copy of the first file in the job.

You can use /BURST without an option to indicate that a burst page should be inserted before an individual file. Place /BURST directly before the file specification to which it applies. For example, PRINT A.DAT, B.DAT, C.DAT, /BURST MESSAGE.X prints a burst page only before the file MESSAGE.X, assuming that /NOBURST is the current default for the print queue.

Use /[NO]BURST to override the installation-defined defaults that have been set for the printer queue you are using.

/COPIES=n

Specifies the number of copies of the file to be printed. The value of nn is a decimal number in the range of 1 through 255. The default number of copies is 1.

If you specify /COPIES as a command keyword, each file named in the command is printed the specified number of times. If you specify /COPIES as a file keyword, only that file that precedes the keyword is printed the specified number of times.

/DELETE
/NODELETE

Controls whether the file is to be deleted after printing.

If you specify /DELETE after a file specification, only that file is deleted after it is printed. If you specify /DELETE after the PRINT command, all files specified are deleted. Files are not deleted after printing by default.

Note that the protection code must allow delete access to the current UIC.

/FEED
/NOFEED

Controls whether the PRINT command automatically inserts form feeds when it prints files that do not have carriage control characters. By default, the PRINT command inserts a form feed when the forms are within four lines of the end of the form. On standard 66-line forms, a form feed occurs after printing 62 lines.

/FLAG[=option]
/NOFLAG

Indicates whether a flag page is to be printed preceding the file. If you specify /FLAG with a file specification, a separate flag page is printed preceding the associated file. The flag page lists the file name.

If you specify /FLAG with the command verb, a flag page is printed for each file in the job. The flag page lists the job name.

The /FLAG keyword can take either of two options: ALL or ONE. The ALL option indicates that each file to be printed will be preceded by a flag page. The ONE option indicates that a flag page will apply only to the first copy of the first file in the job.

You can use /FLAG without an option to indicate that a flag page should be inserted before an individual file. Place /FLAG directly before the file specification to which it applies.

Use /[NO]FLAG to override the installation-defined defaults that have been set for the printer queue you are using.

When you specify /BURST, you need not specify /FLAG. A flag page automatically follows the burst page.

/HEADER
/NOHEADER

Controls whether the name of the file is printed at the top of each printed page.

/PAGES=([lowlim,]uplim)

Specifies the number of pages to print for the specified job. You can use /PAGES to print portions of long files. By default, all pages of the file are printed.

The lowlim specifier refers to the first page in the group of pages that you want printed for that file. If you omit the l, the printing will start on the first page of the file.

The uplim specifier refers to the last page of the file that you want printed. When you want to print to the end of the file, but do not know how many pages that will be, you can use "" as the u. You can omit the parentheses if you are including only a specific value for u. For example, /PAGES=10 prints the first 10 pages of the file; /PAGES=(5,10) prints pages 5 through 10; /PAGES=(5,"") starts printing at page 5 in the file and continues until the end of the file is reached.

/PASSALL
/NOPASSALL

Specifies that the output symbiont print the file in PASSALL mode. By default, the output symbiont does not use PASSALL mode.

If /PASSALL is placed between the PRINT command and any file specifications, all files will be printed in PASSALL mode. To specify /PASSALL with only some files in the job, place the keyword after each file that you want printed in PASSALL mode.

/SETUP=module[, . . .]

Calls for the specified module(s) to be extracted from the device control library and copied to the printer before a file is printed. By default, no device control modules are printed.

/SPACE
/NOSPACE

Specifies whether output is to be double-spaced. By default, output is single-spaced.

/TRAILER[=option]
/NOTRAILER

Controls whether a trailer page is printed at the end of a file. The trailer page displays the job entry number, as well as information about the user submitting the job and the file(s) being printed.

The /TRAILER keyword can take either of two options: ALL or ONE. The ALL option indicates that each file to be printed will be followed by a trailer page. The ONE option indicates that a trailer page will apply only to the last copy of the last file in the job.

To have the /TRAILER keyword apply to individual files in a multifile job, place the keyword immediately after each file that you want to have a trailer page.

Use /[NO]TRAILER to override the installation-defined defaults that have been set for the printer queue you are using.

EXAMPLES

1

```
> PRINT/REMOTE BOSTON::WORK$:[SMITH.MEMO]JUNE10.MEM
```

The PRINT command, which is entered on a node other than BOSTON, queues the file JUNE10.MEM that resides on the BOSTON node. The file is entered on the printer queue at node BOSTON.

2

```
> COPY REPORT.MEM BOSTON::
> PRINT/REMOTE BOSTON::REPORT.MEM
```

The two commands are entered at a node other than BOSTON. The COPY command copies the file REPORT.MEM from the current node to the BOSTON node. The PRINT command queues the file REPORT.MEM located on the BOSTON node for printing at the BOSTON node.

PURGE

The PURGE command deletes all but the highest-numbered version or versions of a specified file or files.

FORMAT

PURGE *[file-spec[, . . .]]*

Command Keywords	**Defaults**
/BACKUP	*/CREATED*
/BEFORE[=time]	*/BEFORE=TODAY*
/BY_OWNER[=uic]	
/[NO]CONFIRM	*/NOCONFIRM*
/CREATED	*/CREATED*
/[NO]ERASE	*/NOERASE*
/EXCLUDE=(file-spec[, . . .])	
/EXPIRED	*/CREATED*
/KEEP=n	*/KEEP=1*
/[NO]LOG	*/NOLOG*
/MODIFIED	*/CREATED*
/SINCE[=time]	*/SINCE=TODAY*

restrictions

RSX–11 MCR does not support the PURGE command.

prompts

None.

command parameter

file-spec[, . . .]

Provides the file specification of the files to be purged. If you specify more than one file, separate each with either a comma (,) or a plus sign (+). Do not specify a version number.

You can use wildcards in place of the file name or file type fields.

If you do not provide the file name and file type, the PURGE command purges all files in the directory.

DESCRIPTION

You use the PURGE command to delete earlier versions of files. By default, PURGE keeps only the highest-numbered version of a file. If you do not include a file specification with the PURGE command, all files in the current directory are affected by the purge.

PURGE differs from the operation of PIP /PU in that the /KEEP keyword specifies the actual number of files to be kept rather than the highest version numbers. The difference occurs only when versions of a file have been deleted. If the highest version number is 15 and version number 13 has already been deleted, PURGE/KEEP=3 deletes all versions except 12, 14, and 15. Using PIP name/PU:3 keeps only versions 14 and 15, however, because PIP calculates 15-3=12 and keeps only version numbers greater than 12. In short, PIP calculates and PURGE counts.

command keywords

/BACKUP

Selects files according to the dates of their most recent backup. This keyword is relevant only when used with the /BEFORE or /SINCE keyword. Use of the /BACKUP keyword is incompatible with /CREATED, /MODIFIED, and /EXPIRED. /CREATED is the default.

/BEFORE[=time]

Selects only those files that are dated before the specified time.

You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values. You can also use the options TODAY, TOMORROW, and YESTERDAY. If no time is specified, TODAY is assumed.

/BY_OWNER[=uic]

Selects one or more files only if their owner User Identification Code (UIC) matches the specified owner UIC.

The UIC can be specified either as an eight-digit octal number or with the asterisk wildcard character (*).

If the /BY_OWNER keyword is specified without a UIC, the UIC of the current process is assumed.

/CONFIRM
/NOCONFIRM

Controls whether a request is issued before each individual PURGE operation to confirm that the operation should be performed on that file.

When the system issues the prompt, you can issue any of the following responses:

- Affirmative responses

  Yes
  True
  1

- Negative responses

  No
  False
  0
  <RET>

- Stop processing responses

  Quit
  <CTRL/Z>

- Continue processing without further prompts response

  All

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE). If you type a response other than one of those in the list, the prompt will be reissued.

/CREATED

Selects files based on their dates of creation. This keyword is relevant only when used with the /BEFORE or /SINCE keyword. Use of the /CREATED keyword is incompatible with /MODIFIED, /EXPIRED, and /BACKUP. /CREATED is the default.

/ERASE
/NOERASE

Writes a block of zeros for each block in each file that is purged.

/EXCLUDE=(file-spec[, . . .])

Excludes any files that match the listed file specification(s) from the PURGE operation. The file specification can contain a directory specification. If you specify two or more files, separate the file specifications with commas.

Wildcard characters are allowed in the file specifications. However, you cannot include either the device or directory field in the file specification(s) you supply with the /EXCLUDE keyword.

/EXPIRED

Selects files according to the dates on which they will expire. This keyword is relevant only when used with the /BEFORE or /SINCE keyword. Use of the /EXPIRED keyword is incompatible with /CREATED, /MODIFIED, and /BACKUP. /CREATED is the default.

/KEEP=n

Specifies the maximum number of versions of the specified files to be retained. If /KEEP is not specified, the command deletes all but the highest-numbered version from the directory.

/LOG
/NOLOG

Specifies that the names of files be displayed as they are deleted. /NOLOG is the default; file names are not displayed as they are purged.

/MODIFIED

Selects files according to the dates on which they were last modified. This keyword is relevant only when used with the /BEFORE or /SINCE keyword. Use of the /MODIFIED keyword is incompatible with /CREATED, /EXPIRED, and /BACKUP. /CREATED is the default.

/SINCE[=time]

Selects only those files that are dated after the specified time.

You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values. You can also use the options TODAY, TOMORROW, and YESTERDAY. If no time is specified, TODAY is assumed.

EXAMPLES

1

```
> PURGE/KEEP=2 *.DAT
```

This command purges all but the two highest-numbered versions of the files in the default directory with a file type of DAT.

2

```
> PURGE [PAYROLL.MSG]
```

This command purges all but the highest-numbered version of the files in subdirectory [PAYROLL.MSG].

RENAME

The RENAME command changes the directory specification, file name, file type, or version number of an existing file or disk directory.

FORMAT

RENAME *input-file-spec[, . . .] output-file-spec*

Command Keywords	**Defaults**
/BACKUP	*/CREATED*
/BEFORE[=time]	*/BEFORE=TODAY*
/BY_OWNER[=uic]	
/[NO]CONFIRM	*/NOCONFIRM*
/CREATED	*/CREATED*
/EXCLUDE=(file-spec[, . . .])	
/EXPIRED	*/CREATED*
/[NO]LOG	*/NOLOG*
/MODIFIED	*/CREATED*
/[NO]NEW_VERSION	*/NEW_VERSION*
/SINCE[=time]	*/SINCE=TODAY*

restrictions

RSX-11 MCR does not support the RENAME command.

prompts

_From:	input-file-spec[, . . .]
_To:	output-file-spec

command parameters

input-file-spec[, . . .]

Specifies the names of one or more files whose specifications are to be changed.

You can use wildcard characters in the directory specification, and also in the file name, file type, or version number fields of the file specification. Through the use of wildcard characters, all files whose specifications satisfy the fields that are specified are renamed.

output-file-spec

Provides the new file specification to be applied to the input file. The RENAME command uses the device, directory, file name, and file type of the input file specification to provide defaults for nonspecified fields in the output file.

You can specify wildcard characters in place of the directory specification, file name, or version number of the output-file-spec parameter; the RENAME command uses the corresponding field in the input file specification to name the output file. Wildcard characters in corresponding fields of the input and output file specification result in multiple rename operations.

The RENAME command supplies output file version numbers according to the first of the following descriptions that applies:

- If the output file specification contains an explicit version number, RENAME uses that version number.
- If the input file specification contains a wildcard in the version number field, RENAME uses the version number of each input file for the name of the output file.
- If no file currently exists with the same file name and file type as that specified for the output file, RENAME assigns a file version number of 1.
- If a file currently exists with the same file name and file type as that specified for the output file name, RENAME assigns the output file a version number one greater than the highest existing version number unless the /NONEW_VERSION keyword is specified.

command keywords

/BACKUP

Selects files according to the dates of their most recent backup. The keyword is relevant only when used with the /BEFORE or /SINCE keyword. Use of /BACKUP is incompatible with /CREATED, /MODIFIED, and /EXPIRED. /CREATED is the default.

/BEFORE[=time]

Selects only those files that are dated before the specified time.

You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values. You can also use the options TODAY, TOMORROW, and YESTERDAY. If no time is specified, TODAY is assumed.

/BY_OWNER[=uic]

Selects one or more files only if their owner User Identification Code (UIC) matches the specified owner UIC.

The UIC can be specified either as an eight-digit octal number or with the asterisk wildcard character.

If /BY_OWNER is specified without a UIC, the UIC of the current process is assumed.

/CONFIRM
/NOCONFIRM

Controls whether RENAME displays the file specification of each file before renaming and asks you to confirm whether or not you want to rename the file. If you specify /CONFIRM, you must respond to a prompt by entering a Y (YES) or a T (TRUE) before the RENAME command renames the file. If you enter anything else, such as N or NO, the file is not renamed.

When the system issues the prompt, you can issue any of the following responses:

- Affirmative responses

  Yes
  True
  1

- Negative responses

  No
  False
  0
  <RET>

- Stop processing responses

  Quit
  <CTRL/Z>

- Continue processing without further prompts response

  All

You can use any combination of upper and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE). If you type a response other than one of those in the list, the prompt will be reissued.

/CREATED

Selects files based on their dates of creation. This keyword is relevant only when used with /BEFORE or /SINCE. Use of /CREATED is incompatible with /MODIFIED, /EXPIRED, and /BACKUP. /CREATED is the default.

/EXCLUDE=(file-spec[, . . .])

Excludes any files that matched the listed file specification(s) from the RENAME operation. The file specification can contain a directory specification. If you specify two or more files, separate the file specifications with commas.

Wildcard characters are allowed in the file specifications. However, you cannot include either the device or directory field in the file specification(s) you supply with the /EXCLUDE keyword.

/EXPIRED

Selects files according to the dates on which they will expire. This keyword is relevant only when used with /BEFORE or /SINCE. Use of /EXPIRED is incompatible with /CREATED, /MODIFIED, and /BACKUP. /CREATED is the default.

/LOG
/NOLOG

Controls whether RENAME displays the file specification of each file it renames.

/MODIFIED

Selects files according to the dates on which they were last modified. This keyword is relevant only when used with /BEFORE or /SINCE. Use of /MODIFIED is incompatible with /CREATED, /EXPIRED, and /BACKUP. /CREATED is the default.

/NEW_VERSION
/NONEW_VERSION

Controls whether RENAME automatically assigns a new version number to the output file if a file with the same name and file type already exists. Assigning a new version number is the default action.

By default, RENAME automatically assigns a new version number to the file if a file with the same name and file type exists.

/SINCE[=time]

Selects only those files that are dated after the specified time.

You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values. You can also use the options TODAY, TOMORROW, and YESTERDAY. If no time is specified, TODAY is assumed.

EXAMPLES

1

```
> RENAME AVERAGE.OBJ OLDAVER
```

The RENAME command changes the file name of the highest existing version of the file AVERAGE.OBJ to OLDAVER.OBJ. If no file named OLDAVER.OBJ exists, the new file is given a version number of 1.

2

```
> RENAME *.TXT;* *.OLD;*
```

In all versions of all files with the file type TXT, this command changes the file type from TXT to OLD. The file names and version numbers are not changed.

3

```
> RENAME  [MALCOLM.TESTFILES]SAVE.DAT  []TEST
```

The RENAME command renames the file SAVE.DAT in the directory MALCOLM.TESTFILES to TEST.DAT. The new file is catalogued in the current default directory.

4

```
> RENAME/LOG
_From: DATA.*.INFO.*
_To:  NEW
%RENAME-I-RENAMED, _DISK0:[SYSTEM]DATA.AAA;1 renamed to _DISK0:[SYSTEM]NEW.AAA;1
%RENAME-I-RENAMED, _DISK0:[SYSTEM]DATA.BBB;1 renamed to _DISK0:[SYSTEM]NEW.BBB;1
%RENAME-I-RENAMED, _DISK0:[SYSTEM]DATA.CCC;1 renamed to _DISK0:[SYSTEM]NEW.CCC;1
%RENAME-I-RENAMED, _DISK0:[SYSTEM]INFO.001;1 renamed to _DISK0:[SYSTEM]NEW.001;1
%RENAME-I-RENAMED, _DISK0:[SYSTEM]INFO.002;1 renamed to _DISK0:[SYSTEM]NEW.002;1
%RENAME-I-RENAMED, _DISK0:[SYSTEM]INFO.003;1 renamed to _DISK0:[SYSTEM]NEW.003;1
>
```

The RENAME command illustrates wildcard characters in the input file names and the use of temporary default file types and version numbers on the output files. The result is the renaming of all six files displayed by the /LOG keyword, three with the file name of DATA and three with the file name of INFO.

RESUME

The RESUME command resumes execution of an indirect command procedure that was interrupted when a .PAUSE directive (see Section 5) was issued by the indirect command file.

FORMAT

RESUME

Command Keywords	**Defaults**
None	*None*

restrictions

- You cannot use a VAX/VMS utility in prompting mode when an indirect command file pauses.
- You cannot use the EDT editor in a prompting mode when an indirect command file pauses.

DESCRIPTION

You can use compatibility mode tasks such as PIP or EDI in prompting mode when an indirect command file pauses. You can also issue any MCR commands.

EXAMPLE

```
> @MYFILE
     .
     .
     .
IND---Pausing. To continue type "RES"
> RESUME
IND---Continuing
     .
     .
     .
> @<EOF>
```

In this example, the user enters the command @MYFILE. Indirect pauses during execution of the command file. To resume execution, the user enters the command RESUME.

RUN (IMAGE)

The RUN (Image) command places a compatibility mode or native mode image into execution in the process.

FORMAT

RUN *file-spec*

Command Keyword	**Default**
/[NO]DEBUG	*None*

restrictions

- If you install an image with elevated privileges, you must specify the image name in the command line as either RUN image-name or RUN image-name.EXE for it to run with its amplified privileges.
- If you specify a version number (RUN image-name.EXE;1) or simply a semicolon (RUN image-name;), the image will run with only current process privileges. When you explicitly specify a version number, the image activator does not search its list of special images that have been installed with privileges.

prompts

None.

command parameter

file-spec

Specifies that an executable image be executed. If you do not specify a file type, the RUN command uses the default file type of EXE.

No wildcard characters are allowed in the file specification.

DESCRIPTION

You can truncate the RUN command to a single letter, R.

The form of the RUN command that is compatible between RSX-11 and VAX-11 RSX MCR is:

```
RUN file-spec
```

command keyword

/DEBUG
/NODEBUG

Controls, for native VAX/VMS images only, whether the image is to be run with the debugger. If the native image was linked with the /DEBUG keyword and you do not want the debugger to prompt, use the /NODEBUG keyword. If the image was linked with /TRACEBACK, traceback reporting is performed when an error occurs.

If the native image was not linked with the debugger, you can specify /DEBUG to request the debugger at execution time. However, if /NOTRACEBACK was specified when the image was linked, /DEBUG is invalid.

EXAMPLES

1

```
> RUN LIBRA
```

The image LIBRA.EXE starts executing in the process.

2

```
> MAC TEST,TEST/-SP=TEST
> TKB TEST/DA,TEST/-SP=TEST
> RUN TEST
ODT:XXXXX
_
```

An RSX–11 MACRO–11 task image named TEST is assembled, built, and run with ODT.

RUN (PROCESS)

The RUN command requests execution of a native or RSX–11 task image. You can request an image for execution in your process (the requesting process), or you can use the RUN command to create a subprocess or detached process to execute the image. By default, the image executes in the requesting process.

FORMAT

RUN *file-spec*

Command Keywords	**Defaults**
/[NO]ACCOUNTING	*/ACCOUNTING*
/AST_LIMIT=quota	*See text*
/[NO]AUTHORIZE	*/AUTHORIZE*
/BUFFER_LIMIT=quota	*See text*
/DELAY=delta-time	
/[NO]DETACHED	
/[NO]DUMP	*/NODUMP*
/ENQUEUE_LIMIT=quota	*See text*
/ERROR=file-spec	
/EXTENT=quota	*See text*
/FILE_LIMIT=quota	*See text*
/INPUT=file-spec	
/INTERVAL=delta-time	
/IO_BUFFERED=quota	*See text*
/IO_DIRECT=quota	*See text*
/JOB_TABLE_QUOTA=quota	*See text*
/MAILBOX=unit	
/MAXIMUM_WORKING_SET=quota	*See text*
/OUTPUT=file-spec	
/PAGE_FILE=quota	*See text*
/PRIORITY=n	*See text*
/PRIVILEGES=(privilege[, . . .])	*/PRIVILEGES=SAME*
/PROCESS_NAME=process-name	
/QUEUE_LIMIT=quota	*See text*
/[NO]RESOURCE_WAIT	*/RESOURCE_WAIT*
/SCHEDULE=absolute-time	
/[NO]SERVICE_FAILURE	*/NOSERVICE_FAILURE*
/SUBPROCESS_LIMIT=quota	*See text*
/[NO]SWAPPING	*/SWAPPING*
/TIME_LIMIT=limit	*See text*
/UIC=uic	
/WORKING_SET=default	*See text*

restrictions

Note: **These restrictions contain important information. You should read them if you intend to use any RUN command keywords. See also the description that follows these restrictions.**

- For RSX–11 task images issuing the Send Data and Receive Data directives, the mailboxes used in emulating these directives are not created until just prior to actual image execution. Therefore, using either the

/INTERVAL or /DELAY keywords does not create the mailboxes when the command is issued; you cannot assume that the mailboxes exist until task-image execution begins.

- The system defines minimum values for each specifiable quota. If you specify either a quota below the minimum or a deductible quota that reduces your quota below the minimum, the RUN command cannot create the process. To determine your current quotas, issue the SHOW PROCESS/QUOTAS command.
- The following keywords schedule execution of the task image:

  /DELAY
  /INTERVAL
  /SCHEDULE

  If you specify any of these keywords, the RUN command creates the process, places it in a state of hibernation, and schedules a wake-up request for the appropriate time. The process cannot execute the task image until it is wakened. For RSX–11 task images, /INTERVAL has the same effect as /DELAY.
- Restrictions apply to the following keywords:

  /[NO]ACCOUNTING
  /[NO]AUTHORIZE
  /PRIORITY
  /PRIVILEGES
  /[NO]SWAPPING
  /UIC

  See the keyword descriptions for more information.

prompt

_File: file-spec

command parameter

file-spec

Provides the file specification of the native or RSX–11 task image to be executed. If the file specification does not include a file type, the RUN command uses EXE by default.

DESCRIPTION

When you specify keywords with the RUN command, the command creates a separate process to execute a specific image. The command displays the created process's identification on SYS$OUTPUT. By default, RUN creates a subprocess with the same UIC, privileges, and priority as the requesting process and deducts resource quotas from the requesting process for the subprocess. If you specify /DETACHED, the command creates a detached process in which to execute the image.

Input, Output, and Error Streams

Use the following keywords to assign equivalent names for the logical names SYS$INPUT (TI), SYS$OUTPUT (TI), and SYS$ERROR (CL):

/INPUT
/OUTPUT
/ERROR

Defining Process Attributes

Use the following keywords to override the default attributes for a process:

/ACCOUNTING
/PRIORITY
/PRIVILEGES
/PROCESS_NAME
/SERVICE_FAILURE
/SWAPPING

Assigning Resource Quotas

You can specify limits that affect performance but are not deducted from your process's resources. The nondeductible quotas are:

/AST_LIMIT
/EXTENT
/IO_BUFFERED
/IO_DIRECT
/MAXIMUM_WORKING_SET
/WORKING_SET

When you issue a RUN command and a detached process is created, pooled quotas are established, restricting the amount of various system resources available to the process and its descendant process(es). The following resource quotas are pooled when you create a detached process (that is, when the values you specify are deducted from your current quotas and are given to the detached process):

/BUFFER_LIMIT
/ENQUEUE_LIMIT
/FILE_LIMIT
/PAGE FILE
/QUEUE_LIMIT
/SUBPROCESS_LIMIT

The quota amounts are returned to you when the detached process is deleted.

The /UIC keyword requests the RUN command to create a detached process; you must have the required privilege to create a detached process. Refer to the appropriate VAX/VMS documentation for more information on user privileges.

If you run an RSX-11 task image that has a task name starting with an alphabetic character in the image label block and specify /PROCESS, the task name becomes the process name; that is, it overrides the name specified by /PROCESS.

Hibernation and Schedule Wakeups

Use the Following keywords to schedule execution of the task image:

/DELAY
/INTERVAL
/SCHEDULE

For an RSX-11 task image, the /INTERVAL keyword has the same effect as /DELAY; that is, the process executes at the designated time and, when the task image exits, the process is deleted. The task image executes only once.

For a native image, /INTERVAL causes the image to hibernate and be executed again after the specified interval if the image issues the VAX/VMS MACRO instruction, RET (Return from Procedure). If the image issues an Exit system service, it exits.

Issue a STOP command to terminate execution of the task image in the process and, if the process is not your current process, it is deleted.

Issue a CANCEL command to cancel wake-up requests that are scheduled for the process but are not yet delivered.

command keywords

/ACCOUNTING /NOACCOUNTING

Controls whether accounting records are to be logged for the created process. By default, all processes are logged in the system accounting file.

You must have the user privilege ACNT to disable accounting.

/AST_LIMIT=quota

Specifies the maximum number of Asynchronous System Traps (ASTs) that the created process can have outstanding.

If you do not specify an AST limit quota, the default value established at system generation time is used; the minimum required for any process to execute is 2. A value of 10 is typical.

This quota is nondeductible.

/AUTHORIZE /NOAUTHORIZE

Controls whether the login image searches the UAF to validate a detached process when the image to be executed is the system login image (LOGINOUT.EXE).

By default, the login image checks the UAF whenever a detached process is created. You can specify /NOAUTHORIZE to create a detached process running under the control of the command interpreter. The process permanent files specified by the /INPUT and /OUTPUT keywords are made available to the command interpreter for input and output.

The user privilege DETACH is required to create a detached process. Any nonspecified attributes of the created process default to the same as those of the current process.

/BUFFER_LIMIT=quota

Specifies in bytes the maximum amount of memory that the created process can use for buffered I/O operations or temporary mailboxes. Mailboxes are used to emulate RSX-11 Send Data and Receive Data directives; see the preceding restrictions.

If you do not specify a buffered I/O limit quota, the default value at system generation time is used. The minimum amount required for any process to execute is 1024 bytes. This quota is pooled; see the preceding restrictions.

/DELAY=delta-time

Requests that the created process be placed in hibernation and awakened after a specified time interval has elapsed.

If you specify /INTERVAL with /DELAY, the first wake-up request is scheduled for the delay time specified and all subsequent wake-up requests are scheduled according to the specified time interval.

/DETACHED
/NODETACHED

Controls whether a detached process or subprocess is created to execute the task image. If you do not specify /DETACHED, a subprocess is created by default.

/DUMP
/NODUMP

If a task image terminates due to an unhandled condition, the /DUMP keyword causes the contents of the address space to be written to a file SYS$LOGIN:IMAGEDUMP.DMP.

/ENQUEUE_LIMIT=quota

Specifies the maximum number of lock requests that a process can queue at any one time.

If you do not specify an enqueue limit quota, the default value established at system generation is used. The minimum required for any process to operate is 2. A value of 6 is typical.

/ERROR=file-spec

Defines an alphanumeric-character equivalence-name string, 1 to 63 characters long, for the logical device name SYS$ERROR. The logical name and equivalence name are placed in the process logical name table for the created process.

/EXTENT=quota

Limits the maximum size to which a task image can expand its working set size (with the Adjust Working Set Limit ($ADJWSL) system service.

If you do not specify an extent quota, the system uses the default value established at system generation time. The minimum value required for any process to execute is 10. A value in the range 400 to 2000 is typical.

The extent quota is nondeductible.

/FILE_LIMIT=quota

Specifies the maximum number of files that a process can have open at one time.

If you do not specify an open-file quota for a created process, the system uses the default value established at system generation time. The minimum amount required for a process to execute is 2. A value of 20 is typical.

This quota is pooled.

/INPUT=file-spec

Defines an alphanumeric-character equivalence-name string, 1 to 63 characters long, for the logical device name SYS$INPUT. The logical name and equivalence name are placed in the process logical name table for the created process.

/INTERVAL=delta-time

Requests that the created process be placed in hibernation and awakened at regularly scheduled intervals as specified by the value of delta time.

If you specify /DELAY or /SCHEDULE with /INTERVAL, the first wake-up request occurs at the delay time or scheduled time specified. All subsequent wake-up requests occur at the specified interval time. If neither /DELAY nor /SCHEDULE is specified, the first wake-up request occurs immediately.

If the image to be executed is an RSX–11 image, the /INTERVAL keyword has the effect of the /DELAY keyword. Only one wake-up occurs.

/IO_BUFFERED=quota

Specifies the maximum number of system-buffered I/O operations the created process can have outstanding at a time.

If you do not specify a buffered I/O quota, the default value established at system generation time is used. The minimum required for any process to execute is 2. A value of 6 is typical.

This quota is not deductible.

/IO_DIRECT=quota

Specifies the maximum number of outstanding direct I/O operations permitted at the time for the created process.

If you do not specify a direct I/O quota, the default value established at system generation time is used. The minimum value of n required for any process is 2. A value of 6 is typical.

This quota is not deductible.

/JOB_TABLE_QUOTA=quota

Allows you to specify a quota for a detached process's job-wide logical name table.

A value of zero has a special meaning. It means that the table, for all practical purposes, has infinite quota because its quota is pooled with that of its parent table, the system directory table.

Note that the /JOB_TABLE_QUOTA keyword only has meaning for detached processes. If the /JOB_TABLE_QUOTA keyword is specified in RUN command which results in the creation of subprocess, it will be ignored.

/MAILBOX=unit

Specifies the unit number of a mailbox to receive a termination message when the created process is deleted. If no mailbox is specified, the creating process does not receive notification when the process is deleted.

You must have the user privilege CMKRNL to use this keyword.

/MAXIMUM_WORKING_SET=quota

Specifies the maximum increase in working set size that the image to be executed in the process can have.

If you do not specify a working set quota, the system uses the default value established at system generation time. The minimum working set size required for a process to execute is 10 pages.

This quota is not deductible.

/OUTPUT=file-spec

Defines an alphanumeric-character equivalence-name string, 1 to 63 characters long, for the logical device name SYS$OUTPUT. The logical name and equivalence name are placed in the process logical name table for the created process.

/PAGE_FILE=quota

Specifies the maximum number of pages that can be allocated in the paging file for the process.

If you do not specify a paging file quota, the system uses the default value established at system generation time. The minimum number of pages required for a process to execute is one 256-page block.

This quota is pooled.

/PRIORITY=n

Specifies the base priority at which the created process is to execute. VAX/VMS priorities range from 0 through 31 (10). Priorities 0 through 15 are normal process priorities. Priorities 16 through 31 are real-time priorities.

You must have the ALTPRI privilege to set the created process's base priority to a value higher than your process's base priority. If you specify no priority or a higher priority than allowed, the system uses your process's priority by default.

/PRIVILEGE=(privilege[, . . .])

Defines privileges for the created process. You must have the SETPRV user privilege to give a process you create any privileges that you do not have.

If you specify multiple privileges, each must be separated by a comma and the list must be enclosed in parentheses.

Any of the privileges can be preceded by NO to deny the process that privilege (for example, NOSWAP_MODE).

Rather than specifying individual privileges, you can specify either ALLPRIV or SAME. ALLPRIV indicates that the created process is to have all privileges. SAME indicates that the created process is to have the same privileges as your process. If you specify /PRIVILEGE=NOSAME, the created process has no privileges.

If you do not specify /PRIVILEGE, the created process has the same privileges as your process (that is, the default is PRIVILEGE=SAME).

/PROCESS=process-name

Defines an alphanumeric character name, 1 to 15 characters long, for the created process. The process name is implicitly qualified by the group number of the process's UIC (that is, process names must be unique only within the group).

If you do not specify a process name, the process will not have a name by default.

/QUEUE_LIMIT=quota

Specifies the maximum number of timer-queue entries that the created process can have outstanding at any one time. This maximum includes timer requests and scheduled wake-up requests. Timer requests are used to emulate RSX–11 Mark Time directives.

If you do not specify a timer-queue entry quota, the system uses the default value established at system generation time. A process does not require any timer-queue limit to execute. A value of 8 is typical.

This quota is pooled.

/RESOURCE_WAIT
/NORESOURCE_WAIT

Enables or disables resource wait mode for the created process. By default, if a system resource is required for a process to execute a particular function and the resource is not available, the system places the process in a wait state until the resource becomes available.

If you specify /NORESOURCE_WAIT, the process receives an error status code when the resource is not available.

/RESOURCE_WAIT is the default.

/SCHEDULE=absolute-time

Requests that the created process be placed in hibernation and later awakened at the time of day specified by the value of absolute time. The rules for entering time values are described in Section 2.

/SERVICE_FAILURE
/NOSERVICE_FAILURE

Enables or disables an exception mode for a system service failure. By default, if an error occurs when a process calls a system service either directly or as a result of RSX–11 directive emulation, a status code indicating the error is returned.

If you specify /SERVICE_FAILURE and an error occurs during the processing of a system service request, the process receives an exception condition. Native images can declare condition handlers for such exception conditions.

/SUBPROCESS_LIMIT=quota

Specifies the maximum number of subprocesses that the created process is allowed to create.

If you do not specify a subprocess limit, the system uses the default value established at system generation time. A process does not require any subprocess quota to execute. A value of 8 is typical.

This quota is pooled.

/SWAPPING
/NOSWAPPING

Enables or disables swapping for the created process. By default, a process is swapped from the balance set in physical memory to allow other processes to execute. Swapping is similar in function to RSX–11 checkpointing.

You must have the user privilege PSWAPM to specify /NOSWAPPING for a process you create. If you specify /NOSWAPPING, the process is not swapped out of the balance set when it is in a wait state.

/TIME_LIMIT=limit

Specifies the maximum amount of CPU time allocated to the created process, in delta time, where the resolution is to 10 ms. When the time expires, the process is deleted. The default value is established at system generation time. A CPU time limit of 0 indicates that CPU time is not restricted; 0 is a typical value.

If you restrict CPU time for a process, specify the time limit according to the rules for specifying delta time values as given in Section 2.

This quota is deductible.

/UIC=uic

Specifies that the created process is to be a detached process. The [g,m] format defines the User Identification Code (UIC) for the created process. The values of g and m are numeric strings that represent the group and member numbers of the process, respectively. The number is in the range of 0 through 377 (8).

You must have the user privilege CMKRNL to use this keyword.

/WORKING_SET=default

Specifies the default working set size for the created process; that is, the number of pages in the working set for the task image to be executed.

If you do not specify a default working set size, the system uses the default value established at system generation time. The minimum number of pages required for a process to execute is 10 pages. The value specified cannot be greater than the working set quota specified by the /MAXIMUM_WORKING_SET keyword.

This quota is not deductible.

EXAMPLES

1
```
> RUN RSXIMAGE.TSK
```

This command activates the RSX-11 task image RSXIMAGE in the requesting process; no subprocess or detached process is created. The system does not prompt for subsequent commands until the task image terminates.

2
```
> RUN VAXIMAGE /UIC=[122,20]
```

This command creates a detached process that runs under UIC [122,20] and has the same privileges and resource quotas as the requesting process.

3

```
> RUN FIRST.TSK /PROCESS=PAY1 /DELAY=::10
> RUN SECOND.TSK /PROCESS=PAY2
```

The commands above create two subprocesses named PAY1 and PAY2, each of which is to execute an RSX–11 task image. PAY1 hibernates as a result of the /DELAY keyword on its RUN command; PAY2 is activated immediately. The two processes cooperate. SECOND, the task image executing in PAY2, awakens PAY1 by issuing a directive.

The process requesting that the two task images be run remains free to accept subsequent user commands.

Both subprocesses have the same privileges as their requesting process. Resource quotas are subtracted from the quotas of the requesting process.

RUNOFF

The RUNOFF command invokes the DIGITAL Standard Runoff (DSR) text formatter to format one or more ASCII files. This description provides a functional overview of the RUNOFF command, emphasizing MCR syntax and grammar. For a complete functional description of the DSR formatter, including more information about the RUNOFF command, see the *VAX DIGITAL Standard Runoff (DSR) Reference Manual*. For information about the commands RUNOFF/CONTENTS and RUNOFF/INDEX, see the separate descriptions of those commands.

FORMAT

RUNOFF *file-spec[, . . .]*

Command Keywords	**Defaults**
/FORM_SIZE=n	*/FORM_SIZE=66*
/[NO]LOG	*/NOLOG*
/[NO]PAUSE	*/NOPAUSE*
/[NO]SIMULATE	*/NOSIMULATE*

Positional Keywords	**Defaults**
/BACKSPACE	
/[NO]BOLD[=n]	*/BOLD=1*
/[NO]CHANGE_BARS[="c"]	
/CONTENTS	*See RUNOFF/CONTENTS command*
/[NO]DEBUG[=(option[, . . .])]	*/NODEBUG*
/DEVICE=(option[, . . .])	*See text*
/[NO]DOWN[=n]	*/NODOWN*
/INDEX	*See RUNOFF/INDEX command*
/[NO]INTERMEDIATE[=file-spec]	*/NOINTERMEDIATE*
/MESSAGES=(option[, . . .])	*/MESSAGES=(OUTPUT,USER)*
/[NO]OUTPUT[=file-spec]	*See text*
/PAGES=string	*All pages output*
/[NO]RIGHT[=n]	*/NORIGHT*
/SEPARATE_UNDERLINE[="c"]	
/[NO]SEQUENCE	*/NOSEQUENCE*
/[NO]UNDERLINE_CHARACTER[="c"]	*/UNDERLINE_CHARACTER="_"*
/VARIANT=string	

restrictions

None.

prompt

`_(r)File: file-spec[, . . .]

command parameter

file-spec[, . . .]

Specifies one or more ASCII files (containing text and DSR commands) to be formatted by the RUNOFF command. If you omit the file type, DSR assumes a file type of RNO. Separate multiple files with commas.

DSR produces an output file having the same file name as the related input file. The output file type depends on the input file type. For a list of input file types and the associated output file types, see the *VAX DIGITAL Standard Runoff (DSR) Reference Manual*. The default output file type is MEM.

Wildcard characters are not allowed in the file specification.

DESCRIPTION

The RUNOFF command allows you to:

- Adjust the amount of text on a page
- Control the position of text on a page
- Control underlining, overwriting, and boldfacing of text
- Override some DSR commands and flags in your input file
- Process all or part of your input file
- Create an intermediate file for indexes or tables of contents

command keywords

/FORM_SIZE=n

Specifies the maximum number of lines that can be output on a page, including all header and trailing information. The default is 66 lines per page. You can specify a different number of lines by supplying a value for the variable n.

/LOG
/NOLOG

Controls whether DSR displays the following information at the terminal when processing is completed:

- DSR version number
- Number of diagnostic messages reported
- Number of output pages generated
- Output file specification

If there are errors in processing, DSR outputs a message to the terminal even if the /NOLOG keyword is specified.

The default is /NOLOG.

/PAUSE
/NOPAUSE

Controls whether DSR pauses after printing each page of output. The /PAUSE keyword allows you to insert single sheets of paper or reproduction masters into hardcopy output devices.

When output is halted, the terminal bell rings to remind the operator to insert a new form. Processing resumes after the operator presses the space bar.

Do not use this keyword in a batch job.

The default is /NOPAUSE.

/SIMULATE
/NOSIMULATE

Controls whether DSR uses line feeds or form feeds to advance to the top of each page.

You can use /SIMULATE with devices that have no formfeed capability. When you specify /SIMULATE, the generation of form feeds is disabled, and DSR prints enough blank lines to cause a skip to the top of each new page. It also causes a pause before the first page of output. To continue after the pauses, press the space bar.

The default, /NOSIMULATE, uses form feeds.

positional keywords

/BACKSPACE

Controls whether DSR uses the ASCII backspace character to perform character-by-character overprinting. The default is for DSR to perform line-by-line overprinting.

/BOLD[=n]
/NOBOLD

Controls whether flagged characters or strings of characters are boldfaced by overprinting the text. You can specify the number of times DSR overprints flagged text by stating a value for the variable n.

If you specify /BOLD without specifying a number, the flagged text is overprinted once (/BOLD=1). If you specify /BOLD=0 or /NOBOLD, the text is not overprinted.

The default is /BOLD=1.

/CHANGE_BARS[="c"]
/NOCHANGE_BARS

Controls whether DSR outputs change bars in the formatted file. The default change bar character is the vertical bar (|). The change bars appear three spaces to the left of the lines of text that you have marked for change bars. See the .BEGIN BAR and .END BAR commands in the *VAX DIGITAL Standard Runoff (DSR) Reference Manual.*

You can replace the default change bar character by supplying a substitute character for the /CHANGE_BARS[="c"] keyword. This replacement character follows the VMS rules for quoted strings. If you put quotation marks around the character, DSR will output the character exactly as entered ("a" -> a). If you do not put quotation marks around the character, DCL will change a lowercase alphabetic character to uppercase (a -> A).

The /CHANGE_BARS keyword without a value uses the default change bar character (|). The /NOCHANGE_BARS keyword overrides any change bar commands in the input file and disables the output of change bars.

/DEBUG[=(option[, . . .])

/NODEBUG

Controls whether DSR outputs the information specified by the /DEBUG options. You can use one or more of the following command options:

Option	Function
CONDITIONALS	Causes DSR to ignore all conditional processing commands (.IF, .IFNOT, .ELSE, .ENDIF) in the input file. DSR includes both "true" and "false" conditional information in the output file along with formatted text. See the *VAX DIGITAL Standard Runoff (DSR) Reference Manual* for further details on the .IF, .IFNOT, .ELSE, .ENDIF, and .VARIABLE commands and the /VARIANT keyword.
FILES	Causes DSR to output all .REQUIRE commands as well as the text of the required files.
INDEX	Causes DSR to output the indexing commands, .INDEX and .ENTRY, in addition to the text to which they refer.
CONTENTS	Causes DSR to output all .SEND TOC commands along with formatted text.
ALL	Specifies all four options (CONDITIONALS, FILES, INDEX, and CONTENTS).

If you specify more than one option, separate them with commas and enclose the list in parentheses. If you specify /DEBUG without specifying any options, ALL is assumed.

The default is /NODEBUG.

/DEVICE=(option[, . . .])

Controls whether DSR generates an output file that is suitable for printing on an LN01 laser printer. You can choose options to indicate different hardware, emphasis for flagged characters, and page orientation from the following list:

Hardware Options	Function
LN01	Produces an output file that is suitable for printing on an LN01 laser printer. The file name is the same as the input file name and the default file type is LNI.
LN01E	Produces an output file that is suitable for printing on an LN01 laser printer using the standard European paper size (4A). The file name is the same as the input file name and the default file type is LNI.

You must choose one of the hardware options.

Emphasis Options	Function
ITALIC	This option loads the italic and bold-italic fonts. The specific font face used depends on whether you have flagged your emphasized characters for underlining or for bolding and underlining. See the *VAX DIGITAL Standard Runoff (DSR) Reference Manual* for complete tables showing which type faces are loaded with which options. ITALIC is the default.
UNDERLINE	No italic or bold-italic fonts are loaded. The characters you have flagged for emphasis are underlined. The LN01 allows only 16 underlined segments per line. One segment is considered one unbroken sequence of underlining. Therefore, if you want to underline individual words and not the spaces between them, you will only be able to underline 16 words (segments) per line. DSR will not report an error if the user exceeds this hardware limit.

Orientation Options	Function
PORTRAIT	This option causes the appropriate portrait font to be loaded for portrait mode. The characters will be printed so that the long side of the paper is at the side rather than the top of the paper. Margins and page size are set to print on a page 8 1/2" by 11" in portrait mode. See the *VAX DIGITAL Standard Runoff (DSR) Reference Manual* for tables giving the page size and margins for European paper. PORTRAIT is the default.
LANDSCAPE	This is the default when the options LN01 or LN01E are used. The page will be printed with the long side of the paper at the top of the page. This option allows you to override the PORTRAIT option.

/DOWN[=n]
 /NODOWN

Controls whether DSR inserts a specified number of blank lines at the top of each page. These blank lines precede any header information. The number of blank lines you specify (n) is deducted from the page size. Therefore the number of text lines that can be output to a page is reduced by the value of n.

For example, if you specify /DOWN=10 with a .PAGE SIZE of 58 lines, the first 10 lines on the page will be blank and there will be a maximum of 48 lines of text and header information.

If you specify the /DOWN keyword without a value, five blank lines are inserted. If you specify /DOWN=0 or omit the keyword, no blank lines are inserted, except those associated with the print device or header layout.

/INTERMEDIATE[=file-spec]
/NOINTERMEDIATE

Controls whether DSR generates an intermediate output file that can be used as input to the DSR table of contents utility and the DSR indexing utility. See RUNOFF/CONTENTS and RUNOFF/INDEX for further information on producing tables of contents and indexes.

If you specify /INTERMEDIATE, DSR creates an output file that has the same file name as the input file and a file type of BRN. You can rename the output file by supplying a file specification that is different from the default values.

/NOINTERMEDIATE is the default.

/MESSAGES=(option[, . . .])

Specifies the destination of all DSR error messages. You can indicate a specific destination by using one or both of the following options:

Option	Function
OUTPUT	Messages are sent to the output MEM file
USER	Messages are displayed on the terminal (SYS$ERROR)

If you specify both options, separate them with commas and enclose the list in parentheses.

The default, /MESSAGES=(OUTPUT,USER), sends messages to the output MEM file and displays them on the terminal.

/OUTPUT[=file-spec]
/NOOUTPUT

Specifies where the output from DSR should go. If you specify /OUTPUT without a file specification, or if you omit the keyword, the output file name is the same as the input file name. The output file type depends on the input file type. The default input file type is RNO and the default output file type is MEM. (See the *VAX DIGITAL Standard Runoff (DSR) Reference Manual* for a list of input file types and the associated output file types.)

You can change the name of the output file by supplying a file specification for the value file-spec.

/OUTPUT=TT: causes output to be sent to the terminal rather than to a disk file. You can use the value TT: with hard copy terminal devices of the "daisy wheel" type to obtain hard copies of your formatted files.

The /NOOUTPUT keyword instructs DSR not to create an output file. You can use the /NOOUTPUT keyword with the /INTERMEDIATE keyword. The combination of these two keywords produces only an intermediate BRN file and does not produce a formatted output file.

You can also use /NOOUTPUT to check an input file for errors without using system resources to generate a formatted output file.

/PAGES=string

Instructs DSR to output only the pages within a specified range. You indicate a range of page numbers by specifying starting and ending page numbers that are separated by a colon (/PAGES=20:40).

When specifying more than one range of page numbers, separate each range with a comma and enclose the list in quotation marks. The maximum number of ranges you can specify is five (/PAGES="2-9:2-12,4-1:4-10,5-9:5-10,A-1:A-3

Index-1:Index-5").

If you specify only a starting page number, output begins at the specified page and continues to the end of the file. To output a single page, the start range and end range must be the same (/PAGES=5:5).

For an entire appendix, only a letter is required (for example, /PAGES="A"). For an entire index, only the word "INDEX" is required (/PAGES="INDEX"). You can refer to a specific appendix or index pages with a numeric suffix such as INDEX-10.

Note that the /PAGES keyword does not recognize any display modes. You must specify the default form of page numbers (5-15) rather than any special form you may have specified with the .DISPLAY NUMBER command (V-15). For details on .DISPLAY NUMBER, see the *VAX DIGITAL Standard Runoff (DSR) Reference Manual.*

The default is for DSR to output all pages.

/RIGHT[=n]
/NORIGHT

Causes the text on each page (including header information) to be shifted to the right by the number (n) of columns specified. A period is needed to force decimal numbers. Note that these columns are not deducted from the page width specified in the input file.

If you specify /RIGHT without specifying a number, the text is shifted to the right five spaces. If you specify a value of zero or omit the keyword, no shift occurs.

/SEPARATE_UNDERLINE[="c"]

Causes flagged text to be underlined by hyphens on the next line rather than by underscores on the same line. You can specify a character to replace the hyphen (the default character for separate underlining). You must specify the replacement character as either a character enclosed in quotation marks or as an octal, decimal, or hexadecimal value for the desired character.

Do not use this keyword with the /NONSPACING_UNDERLINE or the /[NO]UNDERLINE_CHARACTER keywords. The functions of /SEPARATE_UNDERLINE and these keywords are mutually exclusive.

/SEQUENCE
/NOSEQUENCE

Controls whether DSR generates input file sequence numbers in the output file.

For editors that generate line numbers in the input file, the /SEQUENCE keyword causes similar numbering to appear in the output file. The line numbers appear in the left margin at the beginning of each line of output.

If the text editor does not generate sequential numbers in the input file, sequential numbers are still generated in the output file, but without leading zeros.

The default is /NOSEQUENCE.

/UNDERLINE_CHARACTER[="c"]
/NOUNDERLINE_CHARACTER

Causes flagged text to be underlined before a new line is processed. The default underline character is an underscore (_). You can specify a character to replace the default underline character. The replacement character may be specified as either a character enclosed in quotation marks or an octal, decimal, or hexadecimal value of the desired character.

Do not use this keyword with the /SEPARATE_UNDERLINE or /NONSPACING_UNDERLINE keywords. The functions of /[NO]UNDERLINE_CHARACTER and these keywords are mutually exclusive.

/VARIANT=string

Controls the processing of the conditional commands (.IF, .IFNOT, .ELSE, and .ENDIF) by specifying the name(s) of the segments to be processed. See the *VAX DIGITAL Standard Runoff (DSR) Reference Manual* for descriptions of the conditional commands.

If you specify multiple names in a string, you must separate them by commas and enclose the string in quotation marks.

EXAMPLES

1

```
> RUNOFF CHAPT1
```

The RUNOFF command takes the file CHAPT1.RNO as input and writes formatted output to the file CHAPT1.MEM.

2

```
> RUNOFF CHAPT1/RIGHT=10,CHAPT2
```

This RUNOFF command produces a CHAPT1.MEM file with margins 10 spaces to the right of the margins specified in the input file CHAPT1.RNO. It also generates a CHAPT2.MEM file whose margins are not affected by the /RIGHT=10 keyword.

3

```
> RUNOFF/OUTPUT=SYS$OUTPUT: TEXT.DAT,INTRO
```

This command sends output to the terminal rather than to a disk file. The keyword applies to both the input files, TEXT.DAT and INTRO.RNO.

RUNOFF/CONTENTS

The RUNOFF/CONTENTS command invokes the DIGITAL Standard Runoff (DSR) table of contents utility to create an RNT file that can be processed by DSR to make a table of contents. This description provides a functional overview of the RUNOFF/CONTENTS command, emphasizing MCR syntax and grammar. For a complete functional description of the DSR table of contents utility, see the *VAX DIGITAL Standard Runoff (DSR) Reference Manual.*

FORMAT

RUNOFF/CONTENTS *file-spec[, . . .] or*
file-spec[+ . . .]

Command Keywords	**Defaults**
/[NO]BOLD	*/NOBOLD*
/DEEPEST_HEADER=n	*/DEEPEST_HEADER=6*
/[NO]IDENTIFICATION	*/NOIDENTIFICATION*
/[NO]INDENT	*/NOINDENT*
/[NO]LOG	*/NOLOG*
/[NO]OUTPUT[=file-spec]	*See text*
/PAGE_NUMBERS=(option[, . . .])	*/PAGE_NUMBERS=(NORUNNING,LEVEL=6)*
[NO]REQUIRE=file-spec	*/NOREQUIRE*
/[NO]SECTION_NUMBERS	*/SECTION_NUMBERS*
/[NO]UNDERLINE	*/NOUNDERLINE*

restrictions

None.

prompt

_(c)File: file-spec[, . . .]

command parameter

file-spec[, . . .] or file-spec[+ . . .]

Specifies one or more intermediate binary files (BRN) that contain information (chapter titles, header levels, sections, and so on) for making a table of contents. To create a BRN file, use the RUNOFF command with the /INTERMEDIATE keyword. See the RUNOFF command for details on the /INTERMEDIATE keyword.

If you omit the input file type, the DSR table of contents utility uses a default file type of BRN.

For single input files, the DSR table of contents utility produces an output file with the same file name as the input file. The output file type is RNT.

If you separate multiple input files by commas, the table of contents utility produces a separate RNT file for each input file. If you separate multiple input files with a plus sign (+), the table of contents utility produces a single RNT file that contains table of contents information for all the input files. The default output file name is the same as the first input file name, and the default file type is RNT.

Wildcard characters are not allowed in the file specification.

DESCRIPTION

The RUNOFF/CONTENTS command allows you to create an RNT file that can be processed by DSR to produce a table of contents. The command keywords allow you to specify the following characteristics for table of contents entries:

- Bolding or underlining of entries
- Deepest header level to be included
- Running page numbers or chapter-oriented page numbers
- Section numbers output or not output

command keywords

/BOLD
/NOBOLD

Controls whether the bolding specified in chapter and header titles in the input file appears in the table of contents.

If you specify /BOLD, the text flagged for bolding in the body of the document is marked for overprinting in the finished table of contents.

If you specify /NOBOLD, the text flagged for bolding in the document is not overprinted in the table of contents.

The default is /NOBOLD.

/DEEPEST_HEADER=n

Controls how many levels of headers are output in the table of contents. You can specify any number of header levels to be displayed by changing the value of n.

The default is /DEEPEST_HEADER=6.

/IDENTIFICATION
/NOIDENTIFICATION

Reports the current version number of the DSR table of contents utility.

The default is /NOIDENTIFICATION.

/INDENT
/NOINDENT

Controls how many spaces header levels after level 1 are indented in the table of contents.

If you do not use this keyword or if you specify /NOINDENT, all header levels after header level 1 will be indented two spaces.

If you specify /INDENT, each header level after header level 1 will be indented two spaces beyond the preceding header level.

The default is /NOINDENT.

/LOG
/NOLOG

Controls whether the DSR table of contents utility displays the following information at the terminal:

- The name of each input file as it is being processed
- The name of each input file after it is processed
- The name of the output file that is created

If there are any errors in processing, the DSR table of contents utility will send messages to the terminal even if /NOLOG is in effect.

The default is /NOLOG.

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the DSR table of contents utility sends the output.

If you specify the /OUTPUT keyword without a file specification or if you omit the keyword entirely, the DSR table of contents utility creates a file with the same file name as the input file. The default file type is RNT.

To change the file specification from the default values, supply a new file specification.

The /NOOUTPUT keyword suppresses the creation of an output file. You can use /NOOUTPUT to check an input file for errors without using system resources to generate an output file.

/PAGE_NUMBERS=(option[, . . .])

Controls whether the page numbers in the table of contents are running page numbers or chapter-oriented page numbers, and how many levels of headers have page references listed in the table of contents. To specify these options, select from the following list:

Option	Function
LEVEL=n	Specifies that header levels up to and including header level n have page numbers listed in the table of contents. The default is to display page numbers for six levels of headers.
NORUNNING	Specifies chapter-oriented page numbers (such as 1–3, 10–42). You can specify chapter-oriented numbers for the table of contents even if the document does not have chapter-oriented numbers. NORUNNING is the default.
RUNNING	Specifies running page numbers. You can specify running page numbers for the table of contents even if the document does not have running page numbers.

If you supply more than one option, separate them with commas and enclose the list in parentheses.

/REQUIRE=file-spec
/NOREQUIRE

Allows you to change or delete the heading on the first page of a table of contents. The default heading is the word CONTENTS, centered on the page, followed by one blank line. You can substitute another word as a heading or not have any heading.

To change the heading, do one of the following:

1 If you do not want any heading, merely specify a null file as the file-spec for /REQUIRE.

```
> RUNOFF/CONTENTS/REQUIRE=nl:
```

2 If you want to use a different heading, create or edit a file that specifies the heading that you want. Use the file that you create as the file-spec for /REQUIRE.

When you use the /REQUIRE keyword the default heading for the first page of the contents is not generated. The file that you are "requiring must provide the heading. The file can contain both DSR commands that change the format of the first page and the text that you want to appear at the top of the page. Or the file can contain only DSR commands to format the first page of the contents. For example, you can put the command .FIGURE 10 in the file. This command generates 10 blank lines at the top of the first page of the table of contents. You can use these blank lines for later pasteup.

/SECTION_NUMBERS
/NOSECTION_NUMBERS

Controls whether the DSR table of contents utility displays section numbers in the table of contents. The /SECTION_NUMBERS keyword displays section numbers for all header levels in the table of contents. /NOSECTION_NUMBERS suppresses the display of section numbers for all header levels.

The default is /SECTION_NUMBERS.

/UNDERLINE
/NOUNDERLINE

Controls whether the underlining specified in chapter and header titles in the input file appears in the table of contents.

If you specify /UNDERLINE, the text flagged for underlining in the body of the document is underlined in the table of contents.

If you specify /NOUNDERLINE, the text flagged for underlining in the body of the document is not underlined in the table of contents.

The default is /NOUNDERLINE.

EXAMPLES

1

```
> RUNOFF/INTERMEDIATE CHPT1,CHPT2,CHPT3
```

Before using RUNOFF/CONTENTS, you must create a BRN file as input for the DSR table of contents utility. This command creates a separate BRN for CHPT1, CHPT2, and CHPT3.

2

```
> RUNOFF/CONTENTS CHAPT1
```

The RUNOFF/CONTENTS command takes the file CHAPT1.BRN as input and creates an RNT file that can be processed by DSR to produce a final table of contents for CHAPTER 1.

3
```
> RUNOFF/CONTENTS/INDENT/NOSECTION_NUMBERS CHPT2.BRN
```

This command takes the file CHPT2.BRN as input and creates an RNT file. The RNT file can produce a final table of contents that has each header lev after header level 1 indented two spaces beyond the preceding header level The final table of contents will not have section numbers listed. To produc the final table of contents, you must use the RNT file as input to DSR. By default, the formatted table of contents has the same name as the input file and a file type of MEC.

RUNOFF/INDEX

The RUNOFF/INDEX command invokes the DIGITAL Standard Runoff (DSR) indexing utility to create an RNX file that can be processed by DSR to create an index. This description provides a functional overview of the RUNOFF/INDEX command, emphasizing MCR syntax and grammar. For a complete functional description of the DSR indexing utility, see the *VAX DIGITAL Standard Runoff (DSR) Reference Manual.*

FORMAT

RUNOFF/INDEX *file-spec[, . . .] or file-spec[+ . . .]*

Command Keywords	**Defaults**
/[NO]IDENTIFICATION	*/NOIDENTIFICATION*
/LINES_PER_PAGE=n	*/LINES_PER_PAGE=55*
/[NO]LOG	*/NOLOG*
/[NO]OUTPUT[=file-spec]	*See text*
/[NO]PAGE_NUMBERS=option	*/PAGE_NUMBERS=NORUNNING*
/[NO]REQUIRE=file-spec	*/NOREQUIRE*
/[NO]RESERVE=n	*/NORESERVE*

restrictions

None.

prompt

_(i)File: file-spec[, . . .]

command parameter

file-spec[, . . .] or file-spec[+ . . .]

Specifies one or more intermediate binary files (BRN) that contain information (indexing commands, page number references, and so on) for making an index. To create a BRN file, use the RUNOFF command with the /INTERMEDIATE keyword. See the RUNOFF command for details on the /INTERMEDIATE keyword.

If you omit the input file type, the DSR indexing utility uses a default file type of BRN.

For single input files, the DSR indexing utility produces an output file with the same file name as the input file. The output file type is RNX.

If you separate multiple input files with commas, the indexing utility produces an RNX file for each input file. The output files have the same file name as the input files and a default file type of RNX.

If you separate multiple input files with a plus sign (+), the indexing utility produces a single RNX file that contains index information for all the input files. The default output file name is the same as the name of the first input file, and the default file type is RNX.

Wildcard characters are not allowed in the file specification.

DESCRIPTION

The RUNOFF/INDEX command allows you to create an RNX file that can be processed by DSR to produce an index for one or more input files. The formatted index is a two-column index with balanced columns on each page. This index can be used for draft copies or for final production. The command keywords allow you to specify the following characteristics for index entries:

- Running page numbers or chapter-oriented page numbers
- The number of lines of index entries per page
- Special text and heading on the first page of the index

command keywords

/IDENTIFICATION
/NOIDENTIFICATION

Reports the current version number of the DSR indexing utility.

The default is /NOIDENTIFICATION.

/LINES_PER_PAGE=n

Specifies the number of lines of index entries on each page of the finished index. This number does not include the number of lines required for headings and footings.

The default is 55 lines. This value is designed to work properly in the default formatting environment of DSR. You must calculate the value n if you change the default environment in any of the following ways:

- If you use subtitles in the document that requires the RNX file
- If you make the page length for the document anything other than 58 lines per page
- If you use any .LAYOUT other than zero (0)

To calculate the correct value for /LINES_PER_PAGE, use the following formula:

```
/LINES_PER_PAGE=n
n = .PAGE SIZE (the first parameter is length value)
    minus 4 if subtitles are used; minus 3 if no subtitles
    minus the number of lines reserved for .LAYOUT 1,
    .LAYOUT 2, or .LAYOUT 3.
```

/LOG
/NOLOG

Controls whether the DSR indexing utility displays the following information at the terminal when processing is completed:

- The name of each input file as it is being processed
- The name of each input file after it is processed
- The name of the output file that is created

If there are any errors in processing, INDEX will send messages to the terminal even if /NOLOG is in effect.

The default is /NOLOG.

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the DSR indexing utility sends the output.

If you specify the /OUTPUT keyword with no value for the file specification or if you omit the keyword entirely, the output file name is the same as the input file name. The default file type is RNX.

To change the file specification from the default values, supply a new file name and file type for the value file-spec.

The /NOOUTPUT keyword instructs the DSR indexing utility not to create an output file. You can use /NOOUTPUT to check an input file for errors without generating an output file.

/PAGE_NUMBERS=option

Controls whether the page number references in the index are running page numbers or chapter-oriented page numbers. To specify the type of page numbers you want, select from the following options:

Option	Function
NORUNNING	Specifies chapter-oriented page numbers (such as 1–3, 10–42). You can specify chapter-oriented numbers for an index even if they do not appear in the document. NORUNNING is the default.
RUNNING	Specifies running page numbers (such as 1, 50, or 230). You can specify running page numbers for an index even if the document does not display running page numbers.

/REQUIRE=file-spec
/NOREQUIRE

Allows you to change the heading on the first page of an index. The default heading is the word INDEX centered on the page, followed by three blank lines.

To change the heading:

- Create a file that specifies the format and the text that you want as the heading on the first index page.
- Use the file you create as the file specification for /REQUIRE=file-spec.

When you use the /REQUIRE keyword, the default heading is not generated. You must specify a heading for the first index page in the file you create. The file you create can contain only DSR commands, or it can contain both DSR commands and text that you want to appear at the top of the first page of the index. For example, you can put the DSR command .FIGURE 10 in the file. This command generates 10 lines of white space at the top of the index. You can use these blank lines for later pasteup. See the *VAX DIGITAL Standard Runoff (DSR) Reference Manual* for a sample file that changes the index heading.

If you are adding lines of text or white space to the heading on the first page of the index, you must allow space for this addition. Use the /RESERVE=n keyword to provide the space you need. See the /RESERVE keyword for more details.

The default is /NOREQUIRE.

/RESERVE=n
/NORESERVE

Allows you to reserve space at the top of the first page of the index for text or white space that you want to include with the /REQUIRE=file-spec keyword. Determine how many lines of text or white space you are adding to the top of the first page of the index and use this number as the value n for the /RESERVE keyword.

The default is /NORESERVE.

The following examples show how to make an index for a manual. Put indexing commands in your RNO file (see the *VAX DIGITAL Standard Runoff (DSR) Reference Manual* for more information on indexing commands) then follow the steps shown in the following examples.

EXAMPLES

1
```
> RUNOFF/INTERMEDIATE CHAPT1.RNO,CHPT2.RNO
```

The /INTERMEDIATE keyword creates two intermediate binary files, CHPT1.BRN and CHPT2.BRN. These BRN files are used in the next example as input for the RUNOFF/INDEX command.

2
```
> RUNOFF/INDEX/LINES_PER_PAGE=52 CHAPT1
```

The RUNOFF/INDEX command calls the DSR indexing utility to create an RNX file that can be processed by DSR to produce a finished index. The /LINES_PER_PAGE keyword has a value of 52 rather than the default of 55. You can only have 52 lines of index entries because the layout for the book (.LAYOUT 2,3) calls for the page numbers to be centered at the bottom of the page, 3 lines below the last line of text. You must allow for these 3 lines at the bottom of each index page by calculating the value n for the /LINES_PER_PAGE=n keyword.

SEARCH

The SEARCH command searches one or more files for a specified character string or strings and lists all the lines containing occurrences of the string(s).

FORMAT

SEARCH *file-spec[, . . .] search-string[, . . .]*

Command Keywords	**Defaults**
/[NO]EXACT	*/NOEXACT*
/EXCLUDE=(file-spec[, . . .])	
/FORMAT=option	*/FORMAT=TEXT*
/[NO]HEADING	*/HEADING*
/[NO]LOG	*/NOLOG*
/MATCH=option	*/MATCH=OR*
/[NO]NUMBERS	*/NONUMBERS*
/[NO]OUTPUT[=file-spec]	*/OUTPUT=SYS$OUTPUT*
/[NO]REMAINING	*/NOREMAINING*
/[NO]WINDOW[=(n1,n2)]	*/NOWINDOW*

restrictions

None.

prompts

_Filename(s):	file-spec[, . . .]
_String(s):	search-string[, . . .]

command parameters

file-spec[, . . .]

Specifies the names of one or more files to be searched. You must specify at least one file name. If you specify two or more file names, separate them with commas.

Wildcard characters are allowed in the file specification.

search-string

Specifies one or more strings to search for in the specified files. If the search string contains any lowercase letters or nonalphanumeric characters (including spaces), enclose it in quotation marks.

You can use the /MATCH and /EXACT keywords to alter the way that SEARCH matches search strings.

DESCRIPTION

The SEARCH command enables you to search through files for specific character strings. The system displays all lines containing occurrences of the strings. You can use SEARCH keywords to tailor the search operation to your specific needs.

The SEARCH command opens the file with shared read and write access. Therefore, any file that has its attributes set to shared write will be searched even if it is currently opened by other users.

command keywords

/EXACT
/NOEXACT

Controls whether the SEARCH command matches the search string exactly or whether it treats uppercase and lowercase letters as equivalents. The default keyword, /NOEXACT, causes SEARCH to ignore case differences in letters.

Specifying the /EXACT keyword causes the system to use less CPU time. Therefore, if you are sure whether the letters are uppercase or lowercase, it is more efficient to use /EXACT.

/EXCLUDE=(file-spec[, . . .])]

Causes the SEARCH command to exclude the listed file specification(s) from the search. A file specification for this keyword must not include a device or directory specification but does allow wildcard characters.

If you use this keyword, you must follow it with at least one file specification. If you specify two or more files, separate them with commas and enclose the list in parentheses.

If you omit the /EXCLUDE keyword, SEARCH will examine all files in the input file list. Use this keyword when one or more of the file specifications in the input file list contains wildcards and you want to exclude certain files.

/FORMAT=option

Controls reformatting of records during output. You can specify one of the following formatting options:

Option	Function
TEXT	Replaces all control characters in the text with a circumflex character followed by the keyboard control character (for example, CTRL/C is replaced with ^C). The terminal formatting characters <TAB>, <CR>, <LF>, <VT>, and <FF> are passed without change. Two special characters, ESCAPE and DELETE, are converted to <ESC> and <DEL>. Substituting <ESC> avoids unexpected behavior when random escape sequences are printed at a terminal. TEXT is the default format.
PASSALL	Moves characters to the output device with no translation whatsoever. You can use /FORMAT=PASSALL whenever you do not want the SEARCH command to substitute the "^character" symbol for control characters (for example, ^G for CTRL/G). The terminal driver will not actually send eight-bit characters to the terminal unless either SET TERMINAL/PASSALL or SET TERMINAL/EIGHT_BIT is already in effect.
DUMP	DUMP format is very similar to TEXT format. However, with DUMP format, all control characters (including <TAB>, <CR>, and <LF>) are prefixed with the circumflex character. Since there is no corresponding keyboard character for delete, it is displayed as <DEL>.

Option	Function
NONULLS	NONULLS is identical to the DUMP option, except that any null characters are removed from the input file before reformatting. In DUMP mode, the null character is displayed as ^@. NONULLS is convenient when you are searching binary format files, such as EXE or OBJ files, that generally contain many zero bytes.

/HEADING
/NOHEADING

Controls whether the file names and window separators are printed in the output. With the default heading format, file names are printed only when more than one file is specified or when wildcard characters are used. The separator, a line of 30 asterisks, is displayed between groups of lines that belong to different files. If you specify the /WINDOW keyword, a line of 15 asterisks separates each window within a file.

/LOG
/NOLOG

Controls whether the SEARCH command produces a line containing the file name and the number of records and matches for each file searched. This line is output to the current SYS$OUTPUT device.

/MATCH=option

Indicates how the SEARCH command matches multiple search strings. The /MATCH keyword has four options:

Option	Function
OR	Matches if a record contains any of the search strings
AND	Matches if and only if all the search strings are present in the single record
NAND	The negation of AND; matches only if the record does not contain one or more of the search strings
NOR	The negation of OR; matches only if the record does not contain all the search strings

When only one search string is specified, the effects of OR and AND are identical. Similarly, the effects of NOR and NAND are identical when only one search string is supplied.

/NUMBERS
/NONUMBERS

Controls whether the source line number is displayed at the left margin of each line. By default, line numbers are not printed.

/OUTPUT[=file-spec]
/NOOUTPUT

Controls whether the results of the search are output to a specified file. By default, the results are output to the current SYS$OUTPUT device.

The SEARCH command performs much more efficiently when /NOOUTPUT is specified, because it saves a large amount of CPU time. If you are searching through a large number of files, you might want to issue the SEARCH/NOOUTPUT/LOG command first. Then you can issue the SEARCH command, specifying only those files that contain matches.

/REMAINING
/NOREMAINING

Controls whether the SEARCH command displays all records from the first matched record to the end of the file. The /REMAINING keyword overrides the /WINDOW=n2 keyword. However, you can still use the /WINDOW=n1 keyword to specify the number of lines to be printed above the line containing the matched record.

The default is /NOREMAINING.

/WINDOW[=(n1,n2])
/NOWINDOW

Controls the number of lines listed along with the line containing the matching string.

If you specify the /WINDOW keyword with a single number n, n—1 additional lines are displayed for each line containing the search string. Half the additional lines are listed above the line containing the match and half are listed below. If n—1 is an odd number, the extra line is listed below the search string.

For example, if you specify /WINDOW=10, nine additional lines are listed along with the line containing the search string: Four lines are listed above the line containing the search string and five lines are listed below it, making a total of ten lines.

If you specify /WINDOW without specifying a number, the default number of five lines—two above, one containing the search string, and two below—is used.

If the form /WINDOW=(n1,n2) is used, n1 refers to the number of lines above the matched line and n2 to the number of lines below. Either of these numbers can be zero.

IF /WINDOW=0 is specified, SEARCH will display the file name of each file containing a match, but no records. You can use this specification to create a file (using the /OUTPUT keyword) that can be inserted into a command file to manipulate the files containing matches.

If you omit the /WINDOW keyword entirely, only the line in which the match is satisfied is displayed.

EXAMPLES

1
```
> SEARCH CABLE.MEM,JOYNER.MEM "MANUAL TITLE"
```

The SEARCH command searches the files CABLE.MEM and JOYNER.MEM for occurrences of the character string MANUAL TITLE. Each line containing the string is displayed at the terminal. The string must be enclosed in quotation marks because it contains a space character.

2

```
> SEARCH/OUTPUT=RESULTS.DAT/WINDOW=9 DISLIST.MEM NAME
```

The SEARCH command searches the file DISLIST.MEM for occurrences of the character string NAME and outputs the results to the file RESULTS.DAT. The four lines preceding and following each occurrence of NAME are included in the output.

3

```
> SEARCH/OUTPUT=ALLSUB.COM/WINDOW=20000 *.COM SUBMIT
```

The SEARCH command searches all command files in the current directory for the string SUBMIT. If a match is found, SEARCH will copy the entire command file to the output file because the window is so large.

4

```
> SEARCH/OUTPUT=COLUMBUS.OH/WINDOW=(2,0)/NOHEAD/MATCH=AND-
>_*.DAT COLUMBUS,OH
```

The SEARCH command searches all files of type DAT for lines containing both COLUMBUS and OH. When a match is found, the two previous lines (containing name and street address) are copied to the new file. The new file COLUMBUS.OH will be ready to use since it will not contain headings and window separators.

5

```
> SEARCH/OUTPUT=SWAP.LIS/FORMAT=PASSALL/NUMBERS/EXACT-
>_/WINDOW=10000 SWAP.PAS SWAP
```

The SEARCH command produces a listing file with the line numbers at the left margin. The /FORMAT=PASSALL keyword is specified so that formfeed characters in the source are passed through. The /EXACT keyword is specified for efficiency (since the name SWAP in the program statement is in uppercase). The /WINDOW keyword is used so that the entire file is copied to the output file SWAP.LIS.

6

```
> SEARCH/REMAINING CABLE.LOG FORTRAN
```

The SEARCH command displays all the lines that follow the first occurrence of the string FORTRAN in the CABLE.LOG file.

7

```
> SEARCH OMAHA::XXX1:[EXP]SUB.DAT,DATA.LIS RSX
```

The SEARCH command searches through the files SUB.DAT and DATA.LIS at remote node OMAHA for all occurrences of the string RSX. The list of all records containing the string RSX is displayed at the local terminal.

SET

The SET command provides a number of options that allow you to set characteristics and defaults for your process, job, and certain devices. Table 4–3 briefly summarizes all the SET options. Note that RSX–11 MCR supports only the /UIC SET command option.

Most of the SET options are equivalent to those described in detail in the *VAX/VMS DCL Dictionary* or the *Guide to VAX/VMS System Management and Daily Operations*. Each option references the manual in which a full description can be found.

Table 4–3 SET Command Options

Option	Function
ACCOUNTING [2]	Selectively enables or disables the recording of particular kinds of accounting information
ACL [2]	Associates an Access Control List (ACL) with one or more system objects
AUDIT [2]	Enables or disables forms of security auditing
BROADCAST [2]	Allows you selectively to prevent various kinds of messages from being broadcast to your terminal
CARD_READER [2]	Defines the translation mode for a card reader
COMMAND [1]	Adds commands defined in a command description file to your process command set or to a command tables file (VAX–11 RSX users may want to use the /CLI=MCR keyword with this option)
[NO]CONTROL [1]	Enables or disenables interrupts caused by CTRL/Y
DAY [2]	Changes the default day type, which controls when certain users can or cannot log in
DEFAULT [2]	Establishes a disk and/or directory as the current default
DEVICE [2]	Establishes the spooling and error-logging status on a device
DEVICE/ACL [2]	Allows you to modify the (ACL) of a device
DIRECTORY [2]	Modifies the characteristics of a directory

[1] Described in this manual.
[2] Described in the *VAX/VMS DCL Dictionary*.

Table 4–3 (Cont.) SET Command Options

Option	Function
DIRECTORY/ACL [2]	Allows you to modify the ACL of one or more directories.
FILE [2]	Modifies the characteristics of a file
FILE/ACL [2]	Modifies the ACL of one or more files
HOST [2]	Establishes a virtual communication link between a terminal and a network node to which the terminal is not directly connected
LOGINS [2]	Establishes the maximum number of users able to log in to the system
MAGTAPE [2]	Defines the density of a magnetic tape device or rewinds the tape
MESSAGE [2]	Overrides or supplements system messages
PASSWORD [2]	Allows users to change their own passwords
PRINTER [2]	Defines characteristics of a printer
PROCESS [1]	Changes execution characteristics of a process
PROTECTION [2]	Defines the protection status of a file or group of files
PROTECTION/DEFAULT [2]	Establishes the default protection for all files subsequently created during the terminal session or batch job
PROTECTION/DEVICE [2]	Establishes the protection to a specific non-file-structured device
QUEUE [2]	Changes the current status or attributes of the specified queue
QUEUE/ENTRY [2]	Changes the current status or attributes of a job that is not currently executing in a specified queue
RESTART_VALUE [2]	Establishes a test value for restarting portions of batch jobs
RMS_DEFAULT [2]	Provides default multiblock and multibuffer count values to be used by VAX RMS for file operations

[1] Described in this manual.
[2] Described in the *VAX/VMS DCL Dictionary*.

Table 4–3 (Cont.) SET Command Options

Option	Function
TERMINAL [2]	Changes characteristics of a terminal
TIME [2]	Resets the system clock to the specified time
/UIC [1]	Changes the UIC of the current process
VOLUME [2]	Changes characteristics of one or more Files–11 volumes
WORKING_SET [2]	Changes the current working set quota or limit

[1] Described in this manual.
[2] Described in the *VAX/VMS DCL Dictionary*.

SET COMMAND

This command invokes the Command Definition Utility to add commands to your process command set or to a specified command table file. For a complete description of the Command Definition Utility, including information about the command SET COMMAND, see the *VAX/VMS Utilities Reference Volume*.

FORMAT

SET COMMAND *[file-spec[, . . .]]*

Command Keywords	**Defaults**
/DELETE=(verb[, . . .])	
/[NO]LISTING[=file-spec]	*/NOLISTING*
/OBJECT[=file-spec]	
/[NO]OUTPUT[=file-spec]	*/OUTPUT*
/REPLACE	*See text*
/TABLE[=filespec]	*/TABLE*

SET CONTROL

This command determines whether control passes to the command interpreter when CTRL/Y is pressed.

FORMAT

SET [NO]CONTROL[=(Y)]

Command Keywords	**Defaults**
None.	*None.*

restrictions

None.

prompts

None.

command parameter

(Y)

Specifies that CTRL/Y be enabled or disabled by the SET CONTROL command. By default, if you do not specify the control character, Y is assumed.

DESCRIPTION

The CTRL/Y function provides a general-purpose escape from the current operation. You can use CTRL/Y during an interactive terminal session to interrupt the current command, command procedure, or program image.

You can use the SET NOCONTROL=Y command for special application programs. When the SET NOCONTROL=Y command is executed in a system-specified command procedure for a particular user at login, that user can communicate only with the application program that controls the terminal.

When SET NOCONTROL=Y is in effect, the INTERRUPT message is displayed but no interruption takes place.

By using the SET NOCONTROL=Y command, you also disable the CTRL/C cancel function for all commands and programs that do not have special action routines responding to CTRL/C.

EXAMPLE

```
> SET NOCONTROL=Y
```

The SET NOCONTROL command disables the CTRL/Y function as well as most CTRL/C functions.

SET PROCESS

This command changes the execution characteristics associated with the process specified for the current terminal session or job. If no process is specified, changes are made to the current process.

FORMAT

SET PROCESS *[process-name]*

Command Keywords	**Defaults**
/IDENTIFICATION=pid	*See text*
/NAME=string	*/NAME=current-process-name*
/PRIORITY=n	*See text*
/PRIVILEGES=(privilege[, . . .])	*See text*
/[NO]RESOURCE_WAIT	
/RESUME	
/[NO]SUSPEND	
/[NO]SWAPPING	*/SWAPPING*

restrictions

- Restrictions apply to the following keywords:

  /IDENTIFICATION
  /PRIORITY
  /PRIVILEGE
  /SUSPEND
  /SWAPPING

  See the keyword descriptions for more information.

prompts

None.

command parameters

process-name

Specifies the name of the process for which the characteristics are to be changed. The process name can contain from 1 to 15 alphanumeric characters. Process names are linked to group numbers. You can specify a process name only if it has the same group UIC as the current process, provided you have GROUP privilege.

If you specify the /IDENTIFICATION keyword, the process name parameter is ignored. If you include neither the process-name nor the /IDENTIFICATION keyword, the current process is assumed. You can use the /IDENTIFICATION keyword to reference a process outside your group, providing you have the user privilege WORLD.

The process-name parameter is limited to use only with the /PRIORITY, /RESUME, and /SUSPEND keywords.

command keywords

/IDENTIFICATION=pid

Specifies the process identification value (PID) of the process for which characteristics are to be changed. The PID is assigned by the system when the process is created. When you specify a PID, you can omit the leading zeros.

If you use the /IDENTIFICATION keyword, the process-name parameter is ignored. The GROUP or WORLD privilege is required to control other processes on the system.

The /IDENTIFICATION keyword can be used only with the /PRIORITY, /RESUME, and /SUSPEND keywords.

/NAME=string

Changes the name of the current process to the name specified. The string parameter can have from 1 to 15 alphanumeric characters. In an indirect command file, this affects only the subprocess that Indirect is using to execute MCR commands. The name of your main process is not changed.

/PRIORITY=n

Changes the priority for the process specified. To increase the priority for any process to a value higher than the base priority of the current process, you must have the ALTPRI privilege. If you do not have the ALTPRI privilege, the value you specify is compared to your current base priority, and the lower value is always used.

The GROUP or WORLD privilege is required to change the priority of other processes on the system.

/PRIVILEGES=(privilege[, . . .])

Enables or disables the user privileges for the current process. If you specify only one privilege, you can omit the parentheses.

You must have SETPRV user privilege to enable a privilege that you are not authorized to possess. Otherwise, the privilege is not enabled, and no warning message is issued. Use the SHOW PROCESS/PRIVILEGES command to determine what privileges are currently enabled.

/RESOURCE_WAIT /NORESOURCE_WAIT

Enables or disables resource wait mode for the current process.

If you specify /NORESOURCE_WAIT, the process will receive an error status code when system dynamic memory is not available or when the process exceeds one of the following resource quotas:

Direct I/O limit
Buffered I/O limit
Buffered I/O byte count (buffer space) quota

/RESUME

Specifies that a process suspended by a previous SET PROCESS command is to be resumed.

/SUSPEND /NOSUSPEND

Controls whether to suspend or resume the process.

If you specify /SUSPEND, the process is placed in a suspended state. You must have either GROUP or WORLD privilege to use this keyword. You cannot suspend your current process.

You can specify /NOSUSPEND to have a previously suspended process resume operation.

/SWAPPING /NOSWAPPING

Enables or disables process swap mode for the current process. By default, a process that is not currently executing can be removed from physical memory so that other processes can execute.

If you specify /NOSWAPPING, the process is not swapped out of the balance set when it is in a wait state. You must have the user privilege (PSWAPM) to disable swapping for your process.

EXAMPLES

1

```
> SET PROCESS/NORESOURCE_WAIT
```

The SET PROCESS command disables resource wait mode for the current process.

2

```
> RUN/PROCESS_NAME=TESTER  CALC
%RUN-S-PROC_ID, identification of created process is 0005002F
> SET PROCESS/PRIORITY=10  TESTER
```

The RUN command creates a subprocess and gives it the name TESTER. Subsequently, the SET PROCESS/PRIORITY command assigns the subprocess a priority of 10.

SET /UIC

The SET /UIC command establishes a new directory default and, if you have change-mode-to-kernel (CMKRNL) privilege, establishes a new User Identification Code (UIC) for your process.

To display your UIC, use the SHOW PROCESS command described in the *VAX/VMS DCL Dictionary*.

RSX–11 MCR supports the SET /UIC command.

FORMAT

SET /UIC=[g,m]

Command Keywords	**Defaults**
None	*None*

restrictions

- You must have change-mode-to-kernel (CMKRNL) privilege to change your UIC.
- There is no VAX/VMS directory format to correspond to UICs greater than [377,377]. If you use SET /UIC to change to a UIC greater than [377,377], your UIC will be changed if you have CMKRNL privilege. However, the default directory will remain unchanged.

prompt

_UIC:	uic

command parameters

g

Specifies the UIC group number in the range of 0 through 37777 (8).

m

Specifies the UIC member number in the range of 0 through 177777 (8).

EXAMPLE

```
> SET /UIC=[200,200]
```

This command establishes [200,200] as the default directory and UIC for the issuing process.

SHOW

The SHOW command provides a number of options that let you display characteristics and defaults for your process, jobs, and devices. Table 4–4 briefly summarizes all the SHOW options. Note that RSX–11 MCR does not support any of the SHOW command options.

Since the operation of the options of the MCR command SHOW are equivalent to the operation of the options of the DCL command SHOW, the options are described in the *VAX/VMS DCL Dictionary* and are only summarized in Table 4–4.

Table 4–4 SHOW Command Options

Option	Display
ACL	The Access Control List (ACL) associated with a system object
AUDIT	The security features that are enabled
BROADCAST	The current message broadcast status
CLUSTER	Cluster activity and performance
[DAY]TIME	The current date and time
DEFAULT	The current default disk and directory device
DEVICES	The status of devices in the system
ERROR	The error count for the CPU, the memory, and any devices with an error count greater than 0
LOGICAL	Current logical name assignments
MAGTAPE	Information about a designated magnetic tape unit
MEMORY	Information about the use of physical memory and other systemwide resources
NETWORK	The availability of network nodes, including the current node
PRINTER	Information about a designated printer
PROCESS	Attributes of the current process, including privileges, resource quotas, memory usage, priority, and accounting information
PROTECTION	The current default protection applied to files
QUEUE	Print or batch jobs that have been queued but not yet printed or processed
QUEUE/CHARACTERISTICS	Characteristic names and numbers that are available on queues
QUEUE/FORM	Form names and numbers that are available on queues

Table 4–4 (Cont.) SHOW Command Options

Option	Display
QUOTA	Displays the current disk quota that is authorized and used by a specific user on a specific disk
RMS_DEFAULT	The current default multiblock and multibuffer counts used by VAX–11 RMS for file operations
STATUS	The status of the current job, including accumulated CPU time, open file count, and count of I/O operations
SYMBOL	Current symbol definitions
SYSTEM	A list of all processes in the system and related process information
TERMINAL	The device characteristics of your terminal
TRANSLATION	The result of translating a logical name
USERS	A list of the current users, in ascending order of their terminal device names
WORKING_SET	The current working set size limit and quota

SORT

The SORT command invokes the VAX/VMS Sort Utility (SORT) to reorder the records in a file into a defined sequence and to create either a new file of the reordered records, or an address file for accessing the reordered records. The following description provides a functional overview of the SORT command. For a complete functional description of the Sort Utility, including more information about the SORT command, see the *VAX/VMS Utilities Reference Volume*.

FORMAT

SORT *input-file-spec[, . . .] output-file-spec*

Command Keywords	**Defaults**
/COLLATING_SEQUENCE=sequence	*/COLLATING_SEQUENCE=ASCII*
/[NO]DUPLICATES	*/DUPLICATES*
/KEY=(field[, . . .])	*See text*
/PROCESS=type	*/PROCESS=RECORD*
/SPECIFICATION[=file-spec]	*See text*
/[NO]STABLE	*/NOSTABLE*
/[NO]STATISTICS	*/NOSTATISTICS*
/WORK_FILES=n	*/WORK_FILES=2*

Input File Keyword	**Default**
/FORMAT=(file-attribute[, . . .])	*None*

Output File Keywords	**Defaults**
/ALLOCATION=n	*None*
/BUCKET_SIZE=n	
/CONTIGUOUS	
/FORMAT=(record-format[, . . .])	
/INDEXED_SEQUENTIAL	
/OVERLAY	
/RELATIVE	
/SEQUENTIAL	

restrictions

None.

prompts

_File: input-file-spec
_Output: output-file-spec

command parameters

input-file-spec[, . . .]

Specifies the name of the file(s) whose records are to be sorted. You can sort up to 10 input files to create 1 output file. Multiple input file specifications must be separated by commas. If the file specifications do not include a file type, SORT assumes the default file type of DAT.

No wildcard characters are allowed in the file specification.

output-file-spec

Specifies the name of the file into which the sorted records are to be written. If an address sort or index sort is selected, the output file specification names the address file. If the file specification does not include a file type, SORT uses the file type of the input file.

No wildcard characters are allowed in the file specification.

command keywords

/COLLATING_SEQUENCE=sequence

Determines which collating sequence, ASCII, EBCDIC, or MULTINATIONAL is used by SORT. SORT arranges characters in ASCII sequence by default.

If the EBCDIC option is selected, input files are sorted as if the EBCDIC key characters were translated into ASCII key characters and then sorted as an ASCII key. Records do not change.

Specifying the MULTINATIONAL option arranges the characters according to the MULTINATIONAL sequence.

/[NO]DUPLICATES

Eliminates all but one of a record with equal keys. Because the record retained is unpredictable, you must specify which duplicate record to keep by invoking SORT at the program level.

The /STABLE and the /NODUPLICATES keywords are mutually exclusive.

/KEY=(field[, . . .])

Defines a sort key, including position, size, order, and data type. If the key field starts in the first position, encompasses the entire record, will be sorted in ascending order, and contains character data, the /KEY keyword need not be specified.

The key field is defined by options that must be separated with commas and enclosed in parentheses.

<table>
<tr><th>Required Option</th><th>Function</th></tr>
<tr><td>POSITION:n</td><td>Specifies the position of the key within each record, where the first character of each record is in position 1.</td></tr>
<tr><td>SIZE:n</td><td>Specifies the length of the sort key in characters, bytes, or digits, depending on the data type of the key field. The valid sizes, based on data types, are:
<table>
<tr><th>Data Type</th><th>Values for n</th></tr>
<tr><td>character</td><td>1–32767</td></tr>
<tr><td>binary</td><td>1, 2, 4, 8, 16</td></tr>
<tr><td>any decimal data type</td><td>1–31</td></tr>
</table>
The total of all key field sizes must be less than 32,767 bytes.</td></tr>
</table>

Elective Option	Function
NUMBER:n	Specifies the precedence of the sort key being defined, where 1 represents the primary sort key, 2 represents the secondary sort key, and so on. If this option is not specified on the first /KEY keyword, NUMBER:1 is assumed. If not specified on any subsequent /KEY keywords, the default value is the number assigned to the previous key plus 1. The legal values are 1 through 255.
BINARY CHARACTER DECIMAL	Indicates the type of data appearing in the sort key field. If not specified, SORT assumes that the data type is CHARACTER.
SIGNED UNSIGNED	Indicates whether the key data type is signed or unsigned binary data.
LEADING_SIGN TRAILING_SIGN	Indicates whether the sign of a decimal data type key appears at the beginning or end of the key. If the data type of the key is specified as DECIMAL and neither of these options is specified, the default is TRAILING_SIGN.
OVERPUNCHED_SIGN SEPARATE_SIGN	Indicates whether the sign of a decimal data type key is superimposed on the decimal value or is separated from the decimal value. If the data type of the key is specified as DECIMAL and neither of these options is specified, the default is OVERPUNCHED_SIGN.
ASCENDING DESCENDING	Indicates whether the key is to be sorted into ascending or descending order. If neither of these options is specified, the default is ASCENDING.

/PROCESS=type

Defines the type of sort. Record sort is the default process. You can specify one of the following sort types:

Type	Function
ADDRESS	Requests that SORT produce an address file sorted by record keys. The output file can be read as an index to read the original file in the desired sequence.
INDEX	Requests that SORT produce an address file containing the key field of each data record and a pointer to its location in the input file. The output file can be read to randomly access the data in the original file in the desired sequence.
RECORD	Requests SORT to resequence the entire contents of the input file and create an output file containing the reordered records.
TAG	Requests SORT to sort only the record keys and then to randomly reaccess the input file to create an output file with the resequenced records.

/SPECIFICATION[=file-spec]

Indicates that the command and file keywords, including key field definitions, are contained in a specification file. If no file specification is included, SORT reads the specification file from SYS$INPUT.

The format and contents of the specification file are described in detail in the *VAX/VMS Utilities Reference Volume.*

No wildcard characters are allowed in the file specification.

/STABLE

Specifies that SORT is to keep records with equal keys in their original order. If /STABLE is not specified, the order of records with equal keys is unpredictable.

/STATISTICS

Specifies that a statistical summary is to be displayed after the SORT operation has completed. By default, no statistical summary is displayed.

/WORK_FILES=n

Specifies the number of temporary work files to be used during the sort process. You can specify 0 or any value from 2 through 10. A value of 0 indicates that no work files are necessary because the data will fit in physical memory.

input file keyword

/FORMAT=(file-attribute[, . . .])

Specifies attributes of the input file to override the existing data that SORT normally obtains through VAX–11 RMS. You can specify one or both of the following options:

Option	Function
RECORD_SIZE:n	Specifies, in bytes, the length of the longest record, overriding the record size defined in the file header or label. You can specify a record size for an input file when the input file is not on disk or when the longest record size is known to be inaccurate. The longest record length that can be specified is 32,767 bytes for sequential files, 16,383 bytes for relative files, and 16,362 bytes for indexed sequential files.
FILE_SIZE:n	Defines the size of the file in blocks. You can specify a file size for a file that is not on disk. If no file size is specified for an input file, SORT uses the default value of 1000 blocks. The maximum file size that can be specified is 4,294,967,295 blocks.

If you specify both options, they must be separated by a comma and enclosed in parentheses.

output file keywords

/ALLOCATION=n

Specifies the number of 512-byte blocks to be allocated for the output file. By default, SORT allocates blocks based on the number of records sorted.

The number of blocks that can be specified ranges from 1 through 4,294,967,295.

/BUCKET_SIZE=n

Specifies the number of 512-byte blocks per bucket for the output file. The maximum size you can specify is 32 blocks.

If you do not specify a bucket size, the bucket size of the output file is the same as that of the input file (when the output file has the same organization as the input file) or defaults to a value of 1 (when the output file has a different organization from the input file).

/CONTIGUOUS

Controls whether the allocation of disk space for the output file is to be contiguous. If you specify /CONTIGUOUS, you must also specify /ALLOCATION to define the number of blocks to allocate for the output file.

By default, SORT output is not contiguous.

/FORMAT=(record-format[, . . .])

Defines the output file record format. You can specify one or more of the following options:

Option	Function
CONTROLLED[:n] FIXED[:n] VARIABLE[:n]	Defines the record format and length of the output file, where n is the length of the longest record in the output file. If n is not specified, it defaults to the length of the longest record in the input file. The maximum record size you can specify is 32,767 (less any control) bytes.
SIZE:n	Specifies, when CONTROLLED is indicated, the size of the fixed portion of the controlled record in bytes. The maximum size of the fixed-length control field is 255 bytes. If CONTROLLED is specified but no size is given, SORT uses the default value of 2 bytes.
BLOCK_SIZE:n	Specifies the block size in bytes when the output file is directed to a magnetic tape. By default, SORT uses the block size of the input file if the input file is a tape file. If the input file is not a tape file, SORT uses, by default, the block size specified when the output tape volume was mounted. You can specity a block size from 20 through 65,532 bytes. Note, however, that to ensure compatible interchange with most non-DIGITAL systems, the block size should be less than 2048 bytes.

If you do not specify /FORMAT to define the record format of the output file, SORT assumes a default output format based on the sort process selected, as follows:

- If the RECORD or TAG type of sort is selected, the output format and record length defaults to the format and record length of the input file.
- If the ADDRESS or INDEX type of sort is selected, the output record format defaults to FIXED.

/INDEXED_SEQUENTIAL

Specifies that the output file is in indexed sequential order. If you specify /INDEXED_SEQUENTIAL, the output file must exist, its organization must be indexed, and it must be empty. Therefore, you must also specify /OVERLAY to overlay the existing file.

By default, a record or tag sort creates an output file that is in the same order as the input file. Specify /INDEXED_SEQUENTIAL to create an indexed sequential output file from a sequential or relative input file.

/OVERLAY

Indicates that the existing file is to be overlaid with the sorted records of the input file.

By default, SORT creates a new output file and does not overlay an existing file.

/RELATIVE

Specifies that the output file is in relative file order.

By default, a record or tag sort results in an output file that has the same order as the input file. Use /RELATIVE to create a relative output file from a sequential or indexed sequential input file.

/SEQUENTIAL

Specifies that the format of the output file is in sequential order. This is the default for an address or index sort process. For a record or tag sort process, the output file format defaults to the order of the input file.

Specify /SEQUENTIAL to create an output file that is in a different format from the input file.

EXAMPLES

1

```
> SORT/KEY=(POSITION:1,SIZE:70)-
>_BOATS.LST  BOATS.TMP
```

This SORT command sorts the records in the file BOATS.LST and creates an output file named BOATS.TMP. All the records in the input file are sorted in alphanumeric order based on the first 80 characters in each record.

2

```
> SORT/KEY=(PO:1,SI:7) DENVER::DB1:[RECS]RNDM.FIL ALPHNAM.SRT
```

This SORT command sorts the file RNDM.FIL at the remote node DENVER according to the key specified. The output file ALPHNAM.SRT is created in the local default directory.

SPAWN

The SPAWN command creates a subprocess of the current process. Portions of the current process context are copied by the subprocess.

FORMAT

SPAWN *[command-string]*

Command Keywords	**Defaults**
/[NO]CLI[=name]	*See text*
/INPUT=file-spec	*See text*
/[NO]LOG	*/LOG*
/[NO]LOGICAL_NAMES	*/LOGICAL_NAMES*
/[NO]NOTIFY	*/NONOTIFY*
/OUTPUT=file-spec	*See text*
/PROCESS=subprocess-name	*See text*
/[NO]TABLE[=command-table]	*See text*
/[NO]WAIT	*/WAIT*

restrictions

- The RESOURCE_WAIT state is required to spawn a process.
- SPAWN does not manage terminal characteristics.
- Restrictions apply to the following keyword:

  /[NO]NOTIFY

  See the keyword description for more information.

prompts

None.

command parameter

command-string

Specifies a command string to be executed in the context of the created subprocess. When the command completes, the subprocess terminates and control is returned to the parent process. If you use both the /INPUT keyword and a command string, commands are obtained from the input file after the specified command string executes. The command string cannot exceed 132 characters.

DESCRIPTION

The SPAWN command enables you to create a subprocess of your current process with the following context copied from the parent process:

- All process logical names and logical name tables except those explicitly marked CONFINE or those created in executive or kernel mode
- Default disk and directory
- Current SET MESSAGE settings

- Current process privileges
- CTRL/Y state

Note that some context items, such as the process's current command tables, are not copied.

When the subprocess is created, the process-permanent open files and any image or procedure context are not copied from the parent process. The subprocess is set to command level 0 (MCR level with the current prompt).

If you do not specify the /PROCESS keyword, the name of this subprocess is composed of the same base name as the parent process and a unique number. For example, if the parent process name is SMITH, the subprocess name can be SMITH_1, SMITH_2, and so on.

The login command file of the parent process is not executed for the subprocess since the context is copied separately, thus allowing for quicker initialization of the subprocess. When the /WAIT keyword is in effect, the parent process remains in hibernation until the subprocess terminates or returns control to the parent by way of the ATTACH command.

More than one process will simultaneously attempt to use the same input or output stream when several processes share the same stream. This applies also if you terminate a subprocess to which you are not currently attached (or a process that is not spawned from the process to which you are currently attached).

You should use the LOGOUT command to terminate the subprocess and return to the parent process. You can also use the ATTACH command (see ATTACH) to transfer control of the terminal to another process in the subprocess tree, including the parent process. (The SHOW PROCESS/SUBPROCESSES command displays the processes in the subprocess tree and points to the current process.)

Note: **Because a tree of subprocesses can be established using the SPAWN command, you must be careful when terminating any process in the tree. When a process is terminated, all subprocesses below that point in the tree are automatically terminated.**

Keywords used with the SPAWN command must directly follow the command verb. The command string parameter begins after the last keyword and continues to the end of the command line.

command keywords

/CLI=name
/NOCLI

Specifies the name of an alternate CLI to be used by the subprocess. If /CLI is not specified, the CLI used will be the same as the parent process.

Context information is propagated as specified by other keywords or defaults to three recognized CLIs: MCR, DCL, or SHELL.

When you specify a CLI other than MCR, DCL, or SHELL, no context at all is copied to the subprocess.

/INPUT=file-spec

Specifies an input file containing one or more CLI commands to be executed by the spawned subprocess. The input file cannot contain any indirect command procedure commands. However, the input file can invoke an indirect command file. If you specify a command string with the SPAWN command and an input file with the /INPUT keyword, the command string is processed before the input file. Once processing of the input file is complete, the subprocess is terminated.

No wildcards are allowed in the file specification.

You cannot explicitly specify non-record-oriented process-permanent files (NRO PPFs) with the /INPUT keyword. The system displays an error message when it encounters such a file as the value for /INPUT.

Note that when NRO PPFs are used as implicit input (that is when /INPUT is not specified and SYS$INPUT is a NRO PPF), the SPAWN command can succeed. The following chart shows what happens.

Process Type	SYS$INPUT	Implicit Input
Interactive	NRO PPF	SYS$COMMAND
Noninteractive	NRO PPF	Null Device
Any	Any other	SYS$INPUT

If SYS$INPUT is a terminal, it cannot have an associated terminal mailbox.

/LOG
/NOLOG

Controls whether the assigned subprocess name is displayed along with any messages indicating transfer of control between processes. This information is displayed by default.

/LOGICAL_NAMES
/NOLOGICAL_NAMES

Determines whether the system passes process logical names and logical name tables to the subprocess. By default, all process logical names and logical name tables are copied to the subprocess except those explicitly marked CONFINE or those created in executive or kernel mode.

/NOTIFY
/NONOTIFY

Controls whether a message is broadcast to the terminal at which you are logged in, notifying you that your subprocess has been completed or aborted. This keyword should not be used unless you specify the /NOWAIT keyword. /NOTIFY cannot be specified when the SPAWN command is executed from within a noninteractive process. The default is /NONOTIFY.

/OUTPUT=file-spec

Identifies the output file to which the results of the SPAWN operation are written. When using the /NOWAIT keyword interactively, do not specify SYS$COMMAND as the /OUTPUT file specification parameter. If you do, the output from both your process and subprocesses will be displayed simultaneously at your terminal.

No wildcards are allowed in the file specification.

You cannot explicitly specify non-record-oriented process-permanent files (NRO PPFs) with the /OUTPUT keyword. The system displays an error message when it encounters such a file as the value for /OUTPUT.

Note that when NRO PPFs are used as implicit output, the SPAWN command can succeed. The following chart shows what happens.

Process Type	SYS$OUTPUT	Implicit Output
Any	NRO PPF	Mailbox transmitting records for the process to write to its current SYS$OUTPUT device
Any	Any other	SYS$OUTPUT

If you omit the /OUTPUT keyword, output is written to the current SYS$OUTPUT device.

/PROCESS=subprocess-name

Specifies the name of the subprocess to be created. If you specify a process name that already exists, an error message is displayed. By default, if you omit the /PROCESS keyword, a unique process name is assigned with the same base name as the parent process and a unique number. The default subprocess name format is:

```
username_n
```

If the /LOG keyword has been specified, the assigned name of the subprocess is displayed.

/TABLE=command-table
/NOTABLE

Specifies the name of an alternate command table for use by the subprocess.

If you do not specify a CLI or command tables, the CLI and command tables used will be the same as those of the parent process. If you specify either /CLI=DCL or /CLI=MCR but you do not specify command tables, the DCL TABLES will be used with DCL and the MCR TABLES will be used with MCR.

/WAIT
/NOWAIT

Controls whether the system waits until the current subprocess is completed before allowing more commands to be issued in the parent process. The /NOWAIT keyword allows you to issue new commands while the specified subprocess is running. When you use the /NOWAIT keyword interactively, be sure to use the /OUTPUT keyword as well so that output from the subprocess is directed to a file rather than to your terminal. Otherwise, your terminal will be used by more than one process at the same time.

By default, you must wait for the subprocess to terminate before you can issue another command.

EXAMPLE

```
> RUN MYPROG
    .
    .
    .
> CTRL/Y  INTERRUPT
> SPAWN MAIL
%MCR-S-SPAWNED, process SMITH_1 spawned
%MCR-S-ATTACHED, terminal now attached to process SMITH_1
MAIL> READ
    .
    .
    .
MAIL> EXIT
%MCR-S-RETURNED, control returned to process SMITH
> CONTINUE
```

The SPAWN command in this example allows you to enter the MAIL utility without terminating the currently running program. After you exit from MAIL, control is returned to the parent process.

STOP

The STOP command terminates execution of one of the following:

- A program or an indirect command file that was interrupted by CTRL/C or CTRL/Y
- A subprocess or a detached process

FORMAT

STOP *[process-name]*

Command Keyword	**Default**
/IDENTIFICATION=pid	*None*

restrictions

- The GROUP user privilege is required to stop any process in the same group.
- The WORLD user privilege is required to stop any process in the system.
- RSX–11 MCR does not support the same functionality provided by the VAX–11 RSX MCR command STOP.

prompts

None.

command parameter

process-name

Specifies the name, in a string of 1 through 15 alphanumeric characters, of the process to be deleted. The specified process must have the same group number in its User Identification Code (UIC) as the current process.

If you specify the /IDENTIFICATION keyword, the process name is ignored. If you specify neither the process-name parameter nor the /IDENTIFICATION keyword, the program executing in the current process is terminated.

DESCRIPTION

The STOP command causes an abnormal termination of the program currently executing. If the program has declared any exit handling routines, they are not given control. Stopping the program executing in a subprocess causes the specified subprocess or detached process to be deleted.

When a program is interrupted by CTRL/Y and the RUN command is issued to execute another program, the interrupted program is also terminated. However, in this case, exit handling routines are allowed to execute before the next program is run.

If you interrupt a command procedure by CTRL/Y and you issue the STOP command, the command procedure is terminated. If the STOP command is executed in a command procedure, it performs no operation.

If you specify a process name or process identification, the STOP command terminates the program currently executing in the specified process and deletes the process. If the process is a batch job process, no notification of deletion occurs; the log file for the batch job does not print.

For more information on how to use commands like this one in command procedures, consult the *Guide to Using DCL and Command Procedures on VAX/VMS.*

command keyword

/IDENTIFICATION=pid

Specifies the process identification code (PID) that the system assigned to the process when the process was created. When you create a process with the RUN command, the RUN command displays the PID of the process it creates.

When you specify the PID, you can omit leading zeros.

EXAMPLES

1

```
> RUN MYPROG
CTRL/Y INTERRUPT
> STOP
```

The RUN command begins executing the program MYPROG. Subsequently, CTRL/Y interrupts the execution and the STOP command terminates the program.

2

```
> @TESTALL
CTRL/Y INTERRUPT
> STOP
```

The @ (Execute Procedure) command executes the procedure TESTALL.CMD. Pressing CTRL/Y interrupts the procedure and the STOP command returns control to the VAX-11 RSX MCR command interpreter.

3

```
> RUN/PROCESS_NAME=LIBRA  LIBRA
%RUN-S-PROC_ID, identification of created process is 0013340D
    .
    .
    .
> STOP LIBRA
```

The RUN command creates a subprocess named LIBRA to execute the program LIBRA.EXE. Subsequently, the STOP command forces the program to exit and deletes the process.

STOP/QUEUE/ABORT

The STOP/QUEUE/ABORT command aborts a job that is currently being printed. The /ABORT keyword is required and the /QUEUE keyword is optional.

FORMAT

STOP/QUEUE/ABORT *queue-name[:]*

Command Keywords	**Defaults**
None	*None*

restrictions

- Requires operator (OPER) privilege, execute (E) access to the queue, or delete (D) access to the current job.
- RSX-11 MCR does not support the STOP/QUEUE/ABORT command.

prompt

_Queue: queue-name[:]

command parameter

queue-name[:]

Specifies the name of the printer queue in which the job was entered.

DESCRIPTION

Use the STOP/QUEUE/ABORT command only to abort the printing of jobs entered in the system output queues, that is, the line printer or terminal queues.

When you issue the STOP/QUEUE/ABORT command, the job currently being printed is terminated and the next job in the queue is dequeued, provided you have sufficient privileges. You can always abort your own job.

EXAMPLE

```
> STOP/QUEUE/ABORT LPA1
```

This command aborts the job currently printing on line printer LPA1.

STOP/QUEUE/ENTRY

The STOP/QUEUE/ENTRY command deletes an entry from a batch queue while it is running. The /ENTRY keyword is required.

FORMAT

STOP/QUEUE/ENTRY *=entry-number*
queue-name[:]

Command Keywords	**Defaults**
None	*None*

restrictions

- You must have GROUP user privilege to stop execution of jobs of other users in your group.
- You must have WORLD or OPERATOR privilege to stop execution of a job that is not yours and that is not in your group.
- RSX–11 MCR does not support the STOP/QUEUE/ENTRY command.

prompts

None.

command parameters

entry-number
Specifies the entry number of the job to be deleted from the batch queue.

queue-name[:]
Specifies the name of the queue in which the job was entered.

DESCRIPTION

Use the STOP/QUEUE/ENTRY command to terminate the execution of a batch job while it is running. This command cannot, however, delete a job while it is waiting to be executed. If you want to delete an entry from a device or batch job queue while the entry is waiting to be executed, use the DELETE/ENTRY command. Note that you can always stop your own job while it is running.

EXAMPLE

```
> STOP/QUEUE/ENTRY=230 SYS$BATCH
```

The STOP/QUEUE/ENTRY command deletes the job associated with entry number 230 in the batch queue SYS$BATCH.

STOP/QUEUE/REQUEUE

Stops the printing of the job currently being printed and places that job at the end of the output queue. The /REQUEUE keyword is required.

FORMAT

STOP/QUEUE/REQUEUE *[=queue-name[:]]*
queue-name[:]

Command Keywords	**Defaults**
/ENTRY=entry-number	*See text*
/HOLD	*See text*
/PRIORITY=n	*See text*

restrictions

- You must have GROUP user privilege to requeue jobs belonging to other users in your group.
- You must have WORLD or OPERATOR user privilege to requeue a job that is not your own and that is not in your group.
- RSX–11 MCR does not support the STOP/QUEUE/REQUEUE command.

prompt

_Queue: queue-name[:]

command parameter

queue-name[:]

Specifies the name of the queue to be stopped. When you also specify a queue name as a parameter for the /REQUEUE keyword, the job is requeued to that queue.

DESCRIPTION

When you requeue a job, that job is placed at the end of the queue with its priority level lowered to 1. The next job in the queue is immediately dequeued for printing.

This command is useful when the line printer runs out of paper while it is printing a job, or when a large job of low priority is currently printing and one or more other jobs in the queue must be printed immediately. Note that you can always requeue your own job.

command keywords

/ENTRY=entry-number

Used with batch queues to stop a currently executing batch job. The entry-number parameter refers to the job entry number that was assigned to the job when it was submitted to the queue. The job entry number that you specify must match the job entry number of an executing job in order for the STOP/QUEUE/REQUEUE/ENTRY command to tàke effect.

You can only specify one entry number for each STOP/QUEUE/REQUEUE /ENTRY command.

/HOLD

When you specify /HOLD, the aborted job is placed in a hold state for later release with the DCL command SET QUEUE/ENTRY/RELEASE or SET QUEUE/ENTRY/NOHOLD. If you do not need to process a job that is in the hold state, you can delete the job with the DCL command DELETE/ENTRY.

/PRIORITY=n

You can use the /PRIORITY keyword to change the priority of the aborted job. The n parameter can be any integer from 0 to 255.

The default value for /PRIORITY is the same as the priority value that the job had when it was stopped. You must have either OPER or ALTPRI privilege to raise the priority value above the value of the SYSGEN parameter MAXQUEPRI. No privilege is needed to set the priority lower than the MAXQUEPRI value.

Generally, the /PRIORITY keyword is used to lower the priority of a job so that it will run at a time when the queue has no other jobs.

EXAMPLE

```
> STOP/QUEUE/REQUEUE LPBO:
```

This command suspends the current print operation on LPB0: and places the job that was currently printing at the end of the queue with a priority of 1. The print operation resumes by starting the next job in the queue.

SUBMIT

The SUBMIT command queues an MCR indirect command file or a DCL command procedure for execution as a batch job.

FORMAT

SUBMIT *file-spec [, . . .]*

Command Keywords	**Defaults**
/AFTER=time	
/CHARACTERISTICS=(c[, . . .])	
/CLI=file-spec	*See text*
/CPUTIME=option	*/CPUTIME=NONE*
/[NO]HOLD	*/NOHOLD*
/[NO]IDENTIFY	*/IDENTIFY*
/[NO]KEEP	*See text*
/[NO]LOG_FILE=[file-spec]	*See text*
/NAME=job-name	*See text*
/[NO]NOTIFY	*/NONOTIFY*
/PARAMETERS=(parameters[, . . .])	
/[NO]PRINTER[=queue-name]	*/PRINTER=SYS$PRINT*
/PRIORITY=n	*See text*
/QUEUE=queue-name[:]	*/QUEUE=SYS$BATCH*
/REMOTE	
/[NO]RESTART	*/NORESTART*
/USER=username	
/WSDEFAULT=n	*See text*
/WSEXTENT=n	*See text*
/WSQUOTA=n	*See text*

Positional Keyword	**Default**
/[NO]DELETE	*/NODELETE*

restrictions

- RSX–11M MCR does not support the SUBMIT command, but RSX–11M–PLUS MCR does.
- Care is advised when you must rely on the default file types used by the SUBMIT command. The default file type is always COM regardless of your current command interpreter or the default command interpreter in the user authorization file.
- If your default command interpreter is DCL, you can only execute a DCL command procedure, and the default file type is COM. If your default command interpreter is MCR, you can only execute MCR commands in the file, and the default file type is COM, not CMD (use the RENAME command to change file names).

  The batch file is not processed automatically by Indirect. Your batch file must contain an @File command to invoke an indirect command file.

- The current directory for the batch job is initially the default login directory. However, the batch command file can change the current directory for the batch job. The current directory for the batch job is the directory that is searched.
- Requires OPER privilege, execute (E) access to the queue, or write (W) access to the queue. (Generally, users have write access to the queue(s) on which they plan to submit jobs.)
- Restrictions apply to the following keywords:

  /PRIORITY
  /REMOTE
  /USER
  /WSDEFAULT
  /WSEXTENT
  /WSQUOTA

  See the keyword descriptions for more information.

prompt

_File: file-spec[, . . .]

command parameter

file-spec[, . . .]

Specifies the name of a command file to be submitted for batch execution. If no file type is specified, the SUBMIT command uses the default file type of COM. If you specify more than one file, use either a comma (,) or a plus (+) to separate file specifications. Files are processed in the order of specification.

If the file specification contains a network node name, the /REMOTE keyword must be specified.

Wildcard characters are allowed in the file specification.

DESCRIPTION

The file(s) queued for process using the SUBMIT command are considered a job. The system assigns a unique job identification number to each job. When you submit a batch job, the system displays both the job number it assigned to the job and the name of the batch job queue in which it entered your job. You can use the SHOW QUEUE command described in the *VAX/VMS DCL Dictionary* to determine the job identification of a queued batch job.

If your default CLI is MCR, you can submit files containing only MCR commands for batch processing. Indirect command processing is not directly available. However, you can invoke an indirect command procedure from a batch command file. If your CLI is DCL, you can submit only DCL command procedures for batch processing. All output from the command file is queued to the device assigned the logical name SYS$PRINT at the time the job terminates unless you specify otherwise. When the command file is dequeued for execution, it is processed by the default command interpreter of the user who submitted the job.

You should specify the file type in file specifications used for the SUBMIT command, especially if your default command interpreter is MCR. Regardless of your default command interpreter, if the submitted command file activates any nested command files, the directory name may have to be present as part of the file specification when the nested command files are specified (see the preceding restrictions).

The attributes of a queued job can be changed using the SET QUEUE/ENTR command described in the *VAX/VMS DCL Dictionary*.

You can use the DELETE/ENTRY command to remove a job from a batch queue or to delete an entry that is being processed.

Batch Job Output: When you submit command procedures for processing b the SUBMIT command, all output from the command procedure is written to a file called name.LOG, where name is the file name of the first command procedure file in the job. (Use the /NAME keyword to give the job a differen name.) This file is initially written on your default disk. When the batch job completes, the system queues the file to SYS$PRINT and deletes it after it ha printed unless the /NOPRINT keyword is used. In that case, the log file is neither printed nor deleted.

When multiple procedures are submitted, the job terminates if any procedure exits with an error or fatal error status.

command keywords

/AFTER=time

Requests that the job be processed after a specific time of day. Specify the time according to the rules in Section 2 for specifying absolute times or a combination of absolute and delta times.

If the specified time has already passed, the job is queued for immediate processing.

By default, jobs are submitted immediately.

/CHARACTERISTICS=(c[, . . .])

Specifies one or more characteristics desired for the job. If you specify only one characteristic, you can omit the parentheses. Codes for characteristics can be either options or digits and are defined at installation. (See the system manager for the list of characteristics available on your system.)

A job can execute on a batch queue only if each characteristic specified with the SUBMIT command is also indicated for that batch queue. If you specify a characteristic that had not been specified for that batch queue, the job is placed in a hold status. (For your job to run, the system manager should stop the queue, change the characteristic(s) of the queue, and restart the queue, specifying the new values listed in the /CHARACTERISTICS keyword.)

Specification of a characteristic for a queue does not prevent jobs not specifying that characteristic from being executed.

Use the SHOW QUEUE/CHARACTERISTICS command to find out the characteristics of the batch queue.

/CLI=file-spec

Enables you to specify a different command interpreter (CLI) to use in processing the job. The file name specifies that the command interpreter be SYS$SYSTEM:filename.EXE. If you do not specify /CLI, the job is run by the command interpreter specified in the user's authorization record.

/CPUTIME=option

Defines a CPU time limit for the batch job. You may specify a delta time (see Section 2), the value 0, or the words NONE or INFINITE for option.

You cannot request more CPU time than is permitted by the base limits or the value in your UAF entry. However, you can use this keyword to override the base queue value established by the system manager or the value in your UAF entry when you need less CPU time than is authorized.

If the CPU time is not restricted by the base queue limits or the value in your UAF entry, you can specify the options 0 or INFINITE to request an infinite amount of CPU time. Specify NONE when you want the amount of CPU time to default to your UAF entry value or the limit specified on the queue.

/HOLD
/NOHOLD

Controls whether the job is to be made available for immediate processing. If you specify /HOLD, the job is not released until you issue a SET QUEUE/NOHOLD command for it.

/IDENTIFY
/NOIDENTIFY

Controls whether the command interpreter displays the job identification number assigned to it and the name of the device to which it has been queued.

/KEEP
/NOKEEP

Controls whether the log file is deleted after it is printed. /NOKEEP is the default unless /NOPRINTER is specified.

/LOG_FILE=file-spec
/NOLOG_FILE

Controls whether a log file with the specified name is created for the job or whether a log file is created.

When you use the /LOG_FILE keyword, the system writes the log file to the file you specify. If you use /NOLOG_FILE, no log file is created. If you use neither form of the keyword, the log file is written to a file in your default directory that has the same file name as the first command file and a file type of LOG. Using neither /LOG_FILE nor /NOLOG_FILE is the default.

You can use /LOG_FILE to specify that the log file be written to a different device. The file specification can either be a valid file specification or a logical name that is defined in the context of the person submitting the job. The process executing the batch job must have access to the device on which the log file will reside.

If you omit /LOG_FILE and specify /NAME, the log file is written to a file having the same file name as that specified by /NAME and the file type LOG. When you omit the /LOG_FILE keyword, the job-name value used with /NAME must be a valid file name.

/NAME=job-name

Defines a 1- to 39-character alphanumeric name string to identify the job. The name string is displayed in response to the SHOW QUEUE command described in the *VAX/VMS DCL Dictionary*.

If you do not specify a name, the name string defaults to the file name of the first file, truncated to eight characters if necessary.

Output files from the indirect command file are named job-name.LOG.

/NOTIFY
/NONOTIFY

Controls whether a message is broadcast to any terminal at which you are logged in, notifying you that your job has been completed or aborted.

/PARAMETERS=(parameter[, . . .])

Specifies from one to eight optional parameters to be passed to the indirect file. The parameters define values that are equated to the symbols p1, p2, . . . p8 in the indirect command file. The symbols are local to the specified indirect command file.

If you specify more than one parameter, separate them with commas and enclose them in parentheses.

The commas delimit the parameters. To specify a parameter that contains special characters or delimiters, enclose the parameter in quotation marks.

/PRINTER[=queue-name]
/NOPRINTER

Controls whether the job log is queued for printing when your job is completed. The /PRINTER keyword allows you to specify a particular print queue. By default, the job log is printed on SYS$PRINT.

If you specify /NOPRINT, the /KEEP keyword is assumed.

/PRIORITY=n

Specifies the priority of the job being submitted. The value of n must be in the range 0 through 255, where 0 is the lowest priority and 255 is the highest.

By default, jobs are assigned the same priority as your process. You can assign lower priorities to nonessential jobs to allow your other jobs to be processed sooner.

The privilege OPER is required to set a priority value that is higher than your process's base priority.

/QUEUE=queue-name[:]

Specifies the name of a specific batch job queue to which the job is to be submitted.

/REMOTE

Indicates that the specified command procedure be executed on a remote node. The file specification must contain the name of the node on which the file resides and at which the procedure is to be executed.

If you specify /REMOTE, you cannot specify any other keywords.

/RESTART
/NORESTART

Specifies whether the job will restart after a crash or a STOP/REQUEUE command. /NORESTART is the default.

/USER=username

Allows you to submit a job for another user. The job will run exactly as if that user had submitted it. The job runs under that user's privileges, UIC, and process priority. Accounting information is logged to that user's account. By default, the user identification comes from the requesting process.

The specifier can be any username that is validated on your system.

You must have CMKRNL privilege and read access to the UAF to use the /USER keyword.

/WSDEFAULT=n

Defines a working set default for the batch job. You may specify a positive integer in the range 1 through 65,535, the value 0, or the word NONE for n.

Use this keyword to override the base queue value established by the system manager or the value in your UAF entry, provided you want to impose a lower value. You may not request a higher value than is specified in your UAF entry.

Specify 0 or NONE if you want the working set value to default to either your user authorization file or the working set default specified on the queue.

/WSEXTENT=n

Defines a working set default for the batch job. You may specify a positive integer in the range 1 through 65,535, the value 0, or the word NONE in place of n.

Use this keyword to override the base queue value established by the system manager or the value authorized in your UAF entry, provided you want to impose a lower value. Specify 0 or NONE if you want the working set extent defaulted to either your UAF or the working set extent specified on the queue. However, you may not want to request a higher value than your default.

/WSQUOTA=n

Defines the maximum size working set for the batch job. This is the working set quota. You may specify a positive integer in the range 1 through 65,535, the value 0, or the word NONE for n.

Use this keyword to override the base queue value established by the system manager or the value in your user authorization file, provided you want to impose a lower value. You may not request a higher value than your default.

Specify 0 or NONE if you want the working set quota to default to either your UAF or the working set quota specified on the queue.

positional keyword

/DELETE
/NODELETE

Controls whether files are deleted after processing. If you specify the /DELETE keyword after the SUBMIT command name, all files in the job are deleted. If you specify the /DELETE keyword following a file specification, only the associated file is deleted after it is processed.

The protection code on the input file(s) must allow delete access to the default User Identification Code (UIC) of the user who submitted the job.

EXAMPLES

1

```
> SUBMIT AVERAGE
  Job Average (queue SYS$BATCH, entry 112) started on VAXRSX_BATCH
```

The SUBMIT command enters the procedure AVERAGE.COM in the batch job queue. When the batch job completes, the log file AVERAGE.LOG is queued for printing.

2

```
> BATCH/NAME=BATCH24/HOLD TESTALL
  Job BATCH24 (queue SYS$BATCH, entry 467) holding
```

The SUBMIT command enters the procedure TESTALL.COM for processing as a batch job, but in a HOLD status. The job will not be released until the SET QUEUE/RELEASE command is issued. The /NAME keyword requests that the batch job be identified as BATCH_24.

TIME

The TIME command displays the current date and time. This command is equivalent to the SHOW DAYTIME and SHOW TIME commands described in the *VAX/VMS DCL Dictionary*.

RSX–11 MCR supports the TIME command.

FORMAT

TIME

Command Keywords	**Defaults**
None	*None*

restrictions

None.

prompts

None.

EXAMPLE

```
> TIME
  22-DEC-1984  12:43:12
```

The TIME command displays the current date and time.

TYPE

The TYPE command displays the contents of a file or a group of files on the terminal.

FORMAT

TYPE *file-spec[, . . .]*

Command Keywords	**Defaults**
/BACKUP	*/CREATED*
/BEFORE[=time]	*/BEFORE=TODAY*
/BY_OWNER[=uic]	
/[NO]CONFIRM	*/NOCONFIRM*
/CREATED	*/CREATED*
/EXCLUDE(=file-spec[, . . .])	
/EXPIRED	*/CREATED*
/MODIFIED	*/CREATED*
/OUTPUT=file-spec	*/OUTPUT=SYS$OUTPUT*
/[NO]PAGE	*/NOPAGE*
/SINCE[=time]	*/SINCE=TODAY*

restrictions

RSX–11 MCR does not support the TYPE command.

prompt

_File: file-spec[, . . .]

command parameter

file-spec[, . . .]

Is the file specification of one or more files to be displayed.

If multiple files are specified, they can be separated with either a comma (,) or a plus sign (+). In either case, the files are displayed in the order listed.

You can use a wildcard instead of the directory, file name, file type, or version fields. The TYPE command displays all files that satisfy the file description in alphanumeric order for Files–11 Structure Level 2. Files are displayed in random order for Files–11 Structure Level 1.

If you omit the file type, the TYPE command uses LIS by default.

DESCRIPTION

When the TYPE command displays output, you can control the display as follows:

- Use CTRL/S to stop the output temporarily. Use CTRL/Q to resume the output display at the point of interruption.
- Use CTRL/O to suppress the display but continue command processing. If you press CTRL/O again before the TYPE command terminates, output resumes at the current point in command processing. However, if you press CTRL/O when the TYPE command is displaying files in a list, the TYPE command suppresses typing the current file and begins typing the

next file in the list. This behavior is an exception to normal CTRL/O processing.

- Use CTRL/Y to stop command execution entirely.

TYPE opens the specified file with shared read and write access. Therefore, any file that has its attributes set to shared write will be typed, even if it is currently opened by another user.

command keywords

/BACKUP

Selects files according to the dates of their most recent backup. This keyword is only relevant when used with the /BEFORE or /SINCE keyword. Use of the /BACKUP keyword is incompatible with /CREATED, /MODIFIED, and /EXPIRED. /CREATED is the default.

/BEFORE[=time]

Selects only those files that are dated before the specified time.

You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values. You can also use the options TODAY, TOMORROW, and YESTERDAY. If no time is specified, TODAY is assumed.

/BY_OWNER[=uic]

Selects one or more files only if their owner User Identification Code (UIC) matches the specified owner UIC.

The UIC can be specified either as an eight-digit octal number or with the asterisk wildcard character.

If the /BY_OWNER keyword is specified without a UIC, the UIC of the current process is assumed.

/CONFIRM
/NOCONFIRM

Controls whether a request is issued before each individual TYPE operation to confirm that the operation should be performed on that file.

When the system issues the prompt, you can issue any of the following responses:

- Affirmative responses

  Yes
  True
  1

- Negative responses

  No
  False
  0
  <RET>

- Stop processing responses

  Quit
  CTRL/Z

- Continue processing without further prompts response

  All

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE). When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, the prompt will be reissued.

/CREATED

Selects files based on their dates of creation. This keyword is only relevant when used with the /BEFORE or /SINCE keyword. Use of the /CREATED keyword is incompatible with /MODIFIED, /EXPIRED, and /BACKUP. /CREATED is the default.

/EXCLUDE=(file-spec[, . . .])

Excludes any files that matched the listed file specification(s) from the TYPE operation. The file specification can contain a directory specification. If you specify two or more files, separate the file specifications with commas.

/EXPIRED

Selects files according to the dates on which they will expire. This keyword is only relevant when used with the /BEFORE or /SINCE keyword. Use of the /EXPIRED keyword is incompatible with /CREATED, /MODIFIED, and /BACKUP. /CREATED is the default.

/MODIFIED

Selects files according to the dates on which they were last modified. This keyword is only relevant when used with the /BEFORE or /SINCE keyword. Use of the /MODIFIED keyword is incompatible with /CREATED, /EXPIRED, and /BACKUP. /CREATED is the default.

/OUTPUT=file-spec

Requests that the output from the TYPE command be written to the specified file rather than to the SYS$OUTPUT device.

/PAGE
/NOPAGE

Requests that output from the TYPE command be displayed one screen at a time. If more that one file has been requested, CTRL/Z may be used to cancel the display of the current file and to continue with the next file.

Use of the /PAGE keyword is incompatible with the /OUTPUT keyword.

/SINCE[=time]

Selects only those files that are dated after the specified time.

You can specify either an absolute time or a combination of absolute and delta times. See Section 2 for complete information on specifying time values. You can also use the options TODAY, TOMORROW, and YESTERDAY. If no time is specified, TODAY is assumed.

EXAMPLE

```
> TYPE BARRETT
```

The TYPE command locates BARRETT.LIS in the default directory and types it on the terminal.

UFD

The UFD command creates a User File Directory (UFD) or subdirectory on a Files–11 Structure Level 1 or 2 volume and enters the name of the UFD in the volume's Master File Directory (MFD). The volume must have been initialized and must be mounted before you can define UFDs for it.

FORMAT

UFD *ddcu:[directory]*

Command Keywords	**Defaults**
/PROTECTION=code */UIC=[g,m]*	*None*

restrictions

VAX–11 RSX does not support the /UF switch for PIP to create directories.

command parameter

ddcu:

Specifies the device, controller, and unit number of the device containing the volume.

[directory]

Specifies the directory or subdirectory name. If you create a subdirectory on a Structure Level 1 volume, RSX–11 cannot read it.

DESCRIPTION

To create a first-level UFD, you must have write access to the MFD on the volume where you are creating the directory. On a system volume, normally only users with a system user identification code (UIC) or the SYSPRV or BYPASS user privilege are allowed write access to the MFD to create a first-level directory. To create a subdirectory, you must be allowed write access to the lowest-level directory that currently exists. The protection and UIC for the MFD are established during volume initialization; they are the volume protection and volume owner UIC.

RSX–11 MCR supports the UFD command.

command keywords

/PROTECTION=code

Defines the protection to be applied to the directory. You can specify the protection code according to the rules given in Section 2.

If you do not specify a value for each access category or if you omit the /PROTECTION keyword when you create a directory, the command uses the protection in effect for the next-highest-level directory, less any delete access for each unspecified category. If you are creating a first-level directory, then the next-highest-level directory is the MFD. (The protection of the MFD is established by the INITIALIZE command.)

/UIC=[g,m]

Specifies the owner UIC for the directory or subdirectory. If you do not specify /UIC, the UFD command by default uses the owner UIC of the directory in which the new directory is being created.

EXAMPLE

```
> MOU DBB2: MYVOL
> UFD DBB2:[230,222]/PROTECTION=(SYSTEM:RWED,OWNER:RWED,GROUP,WORLD)
```

The first command mounts the volume labeled MYVOL. The UFD command requests the creation of a directory with UIC [230,222] on the volume and establishes the protection for that directory.

5 Indirect Command File Processing

This section describes the Indirect Command Processor (Indirect) and indirect command files. Also included are descriptions of the processor directives and symbols that control the execution of Indirect.

5.1 Indirect Command Files

Indirect command files can be used for many different things—from doing simple tasks to performing complex system-control and programming functions.

There are two types of indirect command files: indirect task command files and indirect command interpreter (CLI) command files. The following sections describe these files.

5.1.1 Indirect Task Command Files

An indirect task command file is a text file containing a list of task-specific command lines. Rather than retyping a commonly used sequence of commands, you can type the sequence once, store it in a file, and direct the task to read the file for its commands instead of prompting at the terminal. Tasks respond to commands contained in an indirect command file as if the commands were entered directly from the terminal. Most system-supplied tasks on VAX–11 RSX, such as the MACRO–11 Assembler or the Task Builder, accept indirect task command files.

To initiate indirect task command files, replace the command line for a task with a file specification preceded by an at sign (@). The task requesting input then accesses the specified file and starts to read and respond to the commands contained within it. For example, to initiate a file of MACRO–11 commands from MCR, input the following:

```
> MAC @INPT.CMD RET
```

The MACRO–11 Assembler accesses the file and then executes the command lines contained within the file INPT.CMD.

The default file type for indirect task command files is .CMD. Thus, the command line in the previous example could also be input as follows:

```
> MAC @INPT RET
```

Some tasks allow nested command files (one file invokes another). See the appropriate task documentation for the maximum nesting depth allowed.

Note that indirect task command files can only contain valid task-specific command lines. The Indirect directives (described in 5.6) cannot be used in such command files.

5.1.2 Indirect CLI Command Files

An indirect CLI command file is a text file containing a list of MCR-specific command lines and special directives that allow you to control command file processing (see 5.3). The Indirect Command Processor (Indirect), which has the task name ICM.EXE and the prompt IND, reads the indirect CLI command file, interprets the directives, and passes the CLI commands to MCR.

To initiate an indirect CLI command file, enter the file specification preceded by an at sign (@), whenever MCR can accept input. For example:

```
> @CLIIPT.CMD RET
```

The default file type for indirect CLI command files is also .CMD. Thus, the command line in the previous example could also be input as follows:

```
> @CLIIPT RET
```

You can also initiate indirect command files from DCL. To execute an indirect command file from DCL, you can use the DCL command MCR as shown in the following example:

```
$ MCR ICM @CLIIPT RET
```

You can also define a symbol to invoke Indirect from DCL and then specify the symbol to initiate an indirect command file. For example:

```
$ ICM :== $ICM RET
$ ICM @CLIIPT RET
```

You can initiate indirect command files from either MCR or DCL; however, the CLI commands in the command file must be MCR commands.

Indirect CLI command files can also be nested to a maximum level of four. A maximum level of four means that you can run one command file, which can run another file, which can run a third file, which can run a fourth file, which can run a fifth file.

Note that the Indirect directives described in this section can only be used in indirect CLI command files. All further references to indirect command files refer to CLI command files.

5.2 Indirect Command Processor

When processing an indirect command file, the Indirect task (ICM.EXE) first reads the command file and interprets each command line either as a command to be passed directly to MCR or as a request for action by Indirect. The directives to Indirect are distinguished by a period (.) as the first character in the line.

The Indirect directives form a procedural language that allows you to:

- Define and assign values to logical, numeric, and string symbols
- Substitute a symbol's value into any line of the command file
- Perform arithmetic
- Manipulate strings
- Display text on the user's terminal
- Ask questions of a user

- Control the sequence in which a command file executes
- Call subroutines
- Detect error conditions
- Test symbols and conditions
- Create and access data files
- Parse commands and data
- Enable or disable any of several operating modes
- Control time-based and parallel task execution

Two directives (.BEGIN and .END) permit you to block-structure the command file and create Begin-End blocks. Modular, block-structured command files are easier to debug and maintain. More importantly, Begin-End blocks isolate local symbol definitions as well as labels and thus conserve symbol table space.

When you define a symbol, Indirect creates an entry for the definition in an internal symbol table. Normally, these symbol table entries retain their definitions throughout the execution of the command file under the following conditions:

1 If defined locally, throughout the execution of a command file

2 If defined globally, throughout the execution of all levels of nested command files (a dollar sign ($) at the beginning of the symbol indicates a global symbol)

Local symbols defined within a block, however, retain their definitions only throughout the execution of the commands within that block; they are erased from the symbol table when Indirect encounters an .END directive.

An Indirect directive (.ENABLE GLOBAL; see 5.6.12) allows some symbols to be defined as global for all file levels. Otherwise, each time Indirect enters a deeper level (and if the /LO switch has not been specified; see 5.5), it masks all symbols defined by the previous level out of the symbol table so that only symbols defined at the current level are available. These symbols are local only to the level of the file at which they are defined. When control returns to a previous level, the symbols defined in that level become available once again and the ones from the lower level are lost.

When Indirect reaches the end of the highest level indirect command file, it displays the message (unless inhibited by the .DISABLE DISPLAY directive)

```
> @ <EOF>
```

and then exits.

Indirect displays every MCR command line as it is executed on the requesting terminal. However, if Indirect is activated by @filename/-MC or by @filename/-CLI, MCR command lines in the Indirect file are converted to comments with a leading exclamation point (!); the MCR command lines are displayed but not executed.

An Indirect file can also include comments, which the processor prints at the requesting terminal. Comments that begin a line interpreted by MCR have a leading semicolon (;). (In some cases, the semicolon can be preceded by a directive to Indirect on the same line.) Comments that appear after the start of an MCR command line have a leading exclamation point (!). All lines in an indirect command file that begin with a period in column 1 followed by a semicolon (.;) are treated as command file comments and are not displayed when the file is processed.

Note that no command or comment lines are displayed if .ENABLE QUIET is in effect.

5.3 Indirect Directive Summary

The Indirect directives described in this section are listed here by category. A detailed description of each directive is presented in alphabetical order in 5.6.

Label Definition

.label:	Assigns a name to a line in the command file so that the line may be referenced.

Symbol Definition

.ASK	Defines or redefines a logical symbol and assigns the symbol a true or false value.
.ASKN	Defines or redefines a numeric symbol and assigns the symbol a numeric value.
.ASKS	Defines or redefines a string symbol and assigns the symbol a character string value.
.ERASE	Deletes all local or global symbol definitions or a single global symbol definition.
.SETD/.SETO	Redefines the radix of a numeric symbol to decimal or octal.
.SETL	Defines or redefines a logical symbol and assigns the symbol a true or false value.
.SETN	Defines or redefines a numeric symbol and assigns the symbol a numeric value.
.SETS	Defines or redefines a string symbol and assigns the symbol a character string value.
.SETT/.SETF	Defines or redefines a logical symbol and assigns the symbol a true or false value.

File Access

.DATA	Specifies a single line of data to output to a file.
.CHAIN	Closes the current indirect file and uses commands from another file.
.CLOSE	Closes a file.
.OPEN	Creates and opens an output data file (if the file already exists, creates a new version and opens it).
.OPENA	Opens an existing file and appends subsequent data (does not create a new version). Defaults to .OPEN if the file does not exist.
.OPENR	Opens a file for reading with the .READ directive.

.PARSE Parses strings into substrings.

.READ Reads a record from a file into a specified string variable.

Logical Control

.BEGIN Marks the beginning of a Begin-End block.

.END Marks the end of a Begin-End block.

.EXIT Terminates processing of either Indirect or the current command file, returns control to the invoking terminal or to the previous level, and optionally sets special symbol <EXSTAT> value.

.GOSUB Calls a subroutine within the command file.

.GOTO Branches to a label within the command file.

/ Defines logical end-of-file. Terminates file processing and exits. This directive is equivalent to the .STOP directive.

.ONERR Branches to a label upon detecting a specific Indirect error condition.

.RETURN Effects an exit from a subroutine and returns to the line immediately following the subroutine call.

.STOP Terminates indirect command file processing and optionally sets Indirect exit status. This directive is equivalent to the logical end-of-file (/) directive.

Logical Tests

.IF Determines whether a symbol satisfies a condition.

.IFACT/.IFNACT Determines whether a task is active. For VAX–11 RSX, a task is always inactive. That is, .IFNACT is true and .IFACT is false.

.IFDF/.IFNDF Determines whether a symbol is defined.

.IFENABLED/ .IFDISABLED Tests the .ENABLE/.DISABLE options.

.IFINS/.IFNINS Determines whether a task is installed in the system. For VAX–11 RSX, a task is always installed. That is, .IFINS is true and .IFNINS is false.

.IFLOA/.IFNLOA Determines whether a device driver is loaded. For VAX–11 RSX, a device driver is always loaded. That is, .IFLOA is true and .IFNLOA is false.

.IFT/.IFF Determines whether a logical symbol is true or false.

.TEST Tests the length of a string symbol or locates a substring.

.TESTDEVICE Returns information about a device in the system.

.TESTFILE Determines whether a specified file exists, and determines the equivalence device associated with a logical device name (performs device translation).

.TESTPARTITION Returns information about a partition in the system. This directive returns a default response because VAX/VMS does not contain partitions.

Enable or Disable an Operating Mode

.ENABLE
/.DISABLE Enables or disables control of the following modes:

substitution
timeout parameter
lowercase
terminal attachment
data
global symbol
symbol radix
command line echo
command display
field display
command passing to default CLI
file deletion
truncate error suppression
escape
control-Z exit
overflow

Increment or Decrement Numeric Symbols

.DEC Decrements the value of a numeric symbol by one.

.INC Increments the value of a numeric symbol by one.

Execution Control

.DELAY Delays the execution of an indirect command file for a specified period of time.

.PAUSE Suspends the indirect command file processor and allows user action.

5.4 Symbols

The indirect command processor enables you to define symbols. These symbols can then be tested or compared to control flow through the indirect command file. They may also be substituted in MCR commands, data records for data files, or comments to be displayed on the terminal.

Symbol names are ASCII strings from one to six characters in length. They must start with a letter (A through Z) or a dollar sign ($). The remaining characters must be alphanumeric or a dollar sign.

There are three symbol types:

- Logical
- Numeric
- String

A logical symbol has either a true or a false value.

A numeric symbol can have a numeric value in the range 0 through 65,535 (10). The symbol can be defined to have either a decimal or octal radix. The radix is relevant only when the symbol is substituted (see 5.4.5).

A string symbol has as its value a string of ASCII characters, with a length of 0 through 132 (10) characters.

A symbol's type (logical, numeric, or string) is defined by the first assignment directive that assigns a value to the symbol. Assignment directives can assign the following:

- A true or false value to define a logical symbol (defined by .ASK, .SETL, .SETT, or .SETF)
- An octal or decimal number to define a numeric symbol (defined by .ASKN, .SETN, .SETO, or .SETD)
- A character string to define a string symbol (defined by .ASKS, .READ, or .SETS)

5.4.1 Special Symbols

Indirect automatically defines special symbols dependent on specific system characteristics and on the replies to queries presented during command file execution. Like other symbols, special symbols can be compared, tested, or substituted; and they have three types: logical, numeric, and string. The format of all special symbols is a pair of angle brackets (<>) enclosing a special symbol name.

5.4.1.1 Special Logical Symbols

The special logical symbols are assigned a true or false value under the following circumstances:

Symbol	Value
<ALPHAN>	Set to true if last string entered in response to an .ASKS directive or tested with a .TEST directive contains only alphanumeric characters. An empty string also sets <ALPHAN> to true.
<ALTMOD>	Set to true if last question was answered with an ALTmode or ESCape. Otherwise, <ALTMOD> is set to false.
<BASLIN>	Always set to false for VAX–11 RSX.
<DEFAUL>	Set to true if answer to last query was defaulted (the RETURN key was pressed once) or a timeout occurred.
<EOF>	Set to true if the last .READ or .ASKx directive resulted in reading past the end of the file. Otherwise, <EOF> is set to false.

Symbol	Value
<ERSEEN>	Set to true if any of the following conditions are true (<ERRNUM>, <EXSTAT>, and <FILERR> are described in 5.4.1.2): • <FILERR> is less than 0 if a negative error code was returned. • An exit status (<EXSTAT>) value worse than <WARNING> was returned. • <EOF> is set to true. • <ERRNUM> is not 0. • You used the command line .SETT <ERSEEN>. The command line .SETF <ERSEEN> sets the following conditions: • <FILERR> is set to 0. • <EXSTAT> is set to 0. • <EOF> is set to false. • <ERRNUM> is set to 0.
<ESCAPE>	Set to true if last question was answered with an ALTmode or ESCape. Otherwise, <ESCAPE> is set to false.
<FALSE>	Logical constant used for comparisons with the .IF directive or as a default for the .ASK directive.
<IAS>	Always set to false for VAX–11 RSX.
<LOCAL>	Always set to true for VAX–11 RSX.
<MAPPED>	Always set to true for VAX–11 RSX.
<NUMBER>	Set to true if the last string entered in response to an .ASKS directive or tested with a .TEST directive contains only numeric characters. An empty string also sets <NUMBER> to true.
<OCTAL>	Set to true if the answer to the last .ASKN directive or the radix of the numeric symbol tested in the last .TEST directive is octal or if the last string tested with a .TEST directive contained all numeric characters in the range 0 through 7.
<PRIVIL>	Always set to false for VAX–11 RSX.
<RAD50>	Set to true if the last string entered in response to an .ASKS directive or tested with a .TEST directive contains only Radix–50 characters. Radix–50 characters are the uppercase alphanumeric characters plus period (.) and dollar sign ($). A blank is not a Radix–50 character in this context. An empty string also sets <RAD50> true.
<RSX11D>	Always set to false for VAX–11 RSX.
<TIMOUT>	Set to true if timeout mode is enabled and the last .ASKx directive timed out waiting for a user response.
<TRUE>	Logical constant used for comparisons with the .IF directive or as a default for the .ASK directive.

5.4.1.2 Special Numeric Symbols

The special numeric symbols are assigned the following values:

Symbol	Value
<ERRCTL>	Controls how Indirect processes errors. The symbol is treated as an eight-bit mask. For each class of error that a user's .ONERR target routine processes (see 5.6.21), the appropriate bit is set in the mask. If the bit is clear, Indirect exits after printing the error information. If the eighth bit, which is the sign bit or 200 (8), is set, Indirect does not print any information about the error. The initial default value for <ERRCTL> is 1, which implies that only class 1 errors can be handled with an .ONERR address and that error messages will be printed.
Note:	**If you attempt to trap errors other than default class 1, processing cannot continue in most cases. The error service routine is limited to a fatal error message and .EXIT. The internal state of Indirect is indeterminate in all but class 1 error cases.** See Section 9 for a list of error messages and their assigned class values.
<ERRNUM>	Assigned the class number of an error that Indirect has finished processing. This value can be used for processing specific error types with an .ONERR routine. See Section 9 for a list of error messages and their assigned class numbers.
<ERRSEV>	Assigned the error severity mask associated with the error that Indirect has finished processing. This bit mask corresponds to the bit mask <ERRCTL> used to control the processing.
<EXSTAT>	Assigned the value of 0, 1, 2, 4, or 17 depending on the exit status from the last MCR command line executed. This special numeric symbol is modified at the completion of a synchronous MCR command line. The .EXIT directive can also modify <EXSTAT>. The value is returned from the task that has completed if the task exits with status. Otherwise, the value is returned from MCR. The values 0, 1, 2, 4, and 17 and their corresponding special symbols indicate the following: 0 <WARNIN> Warning 1 <SUCCES> Success 2 <ERROR> Error 4 <SEVERE> Severe error 17 <NOSTAT> The task could not return exit status
<FILERR>	Assigned the FCS-11 or I/O driver status code resulting from a .TESTFILE, .OPENx, or .READ directive operation. See the *RSX-11M/M-PLUS I/O Drivers Reference Manual* for a description of the codes.
<FILER2>	Assigned the error code 0 for VAX-11 RSX.

Symbol	Value
<FORATT>	Assigned the octal value of the file attributes that were used to open the data files.
<MEMSIZ>	Assigned the value of the current system memory size in K words. (K is 1024.) For VAX–11 RSX, this is always set to 2044K (10) words, which is the largest memory size addressable under RSX–11M. This variable does not give you the actual size of the physical memory of your VAX–11 processor.
<SPACE>	Assigned the number, in octal, of free bytes in the internal symbol table for Indirect. The number does not reflect the amount of space that could be gained by the automatic extension of the Indirect task.
<STRLEN>	Assigned the length of the string entered in response to the last .ASKS directive or the string tested by the last .TEST directive. The symbol is also set when a command file is invoked (<STRLEN> contains the number of variables used in the command line) and as the result of a .PARSE statement.

<SYMTYP>	Assigned the numeric code for the type of symbol tested with a .TEST directive. The symbol types have the following code numbers:
	`String  4`
	`Numeric 2`
	`Logical 0`

<SYSTEM>	Assigned an octal number to represent the operating system on which Indirect is running. The value is 5 for VAX–11 RSX.
<SYUNIT>	Assigned the unit number of the user's default device (SY:). The number is always 0 for VAX–11 RSX.
<TICLPP>	Assigned the current page length setting for the terminal. Indirect attempts to determine the length when you first invoke it. If the information is not available, <TICLPP> defaults to 24 (10).
<TICWID>	Assigned the current page width setting for the terminal. Indirect attempts to determine the width when you first invoke it. If the information is not available, <TICWID> defaults to 80 (10).
<TISPED>	Assigned the baud rate for transmitting characters from the host system to the terminal. For VAX–11 RSX the baud rate is 0. VAX–11 RSX does not support the use of this symbol.
<TITYPE>	Assigned the terminal type of the terminal from which Indirect is running. Indirect attempts to determine the terminal type when you first invoke it. If the attempt fails, <TITYPE> is set to zero (0). See the *RSX–11M/M–PLUS I/O Drivers Reference Manual* for a list of the octal values that correspond to the terminal types.

5.4.1.3 Special String Symbols

The special string symbols are assigned the following string values.

Symbol	Value
<ACCOUN>	Assigned certain accounting information from a User's Accounting Block (UAB). For VAX–11 RSX, the fields of <ACCOUN> are null.
<CLI>	Assigned the acronym (three through six letters) of the current command line interpreter. For VAX–11 RSX, the acronym is MCR.
<CONFIG>	Contains the parameter defaults specified when the current Indirect task was built. See the module INDCFG in the system procedure library SYS$LIBRARY:INDSYS.CLB for more details.
<DATE>	Assigned the current date; format is dd-mmm-yy.
<DIRECT>	For VAX–11 RSX, contains the name of the current default directory; format is [NAME].
<EXSTRI>	When Indirect is first initiated, contains build-time information about the Indirect task. The information includes the version number of the task and the time the task was built. Afterward, <EXSTRI> contains the string results from a more deeply nested indirect command file. The results are sent to the calling command file.
<FILATR>	Assigned the seven words of file attribute information contained in the FCS-11 File Descriptor Block (FDB). Returns a string containing ASCII values for the record type, record attributes, record size, highest allocated block, end-of-file block, first free byte, record access, and low word of the device characteristics. The information is from the FDB used in the last .OPENx operation. See the *IAS/RSX–11 I/O Operations Reference Manual* for a description of file attributes and the values of each field.
<FILSPC>	Assigned the fully qualified specification for the file referred to with the last .OPEN, .OPENA, .OPENR, or .TESTFILE directive operation, or in the last specification for a nested command file.
<FMASK>	Contains octal values representing answers to some of the system generation questions. Refer to the module INDSFN in the system procedure library SYS$LIBRARY:INDSYS.CLB for an explanation of the values.
<LIBUIC>	For VAX–11 RSX, assigned using the system logical name RSX$LIBUIC. If the logical name RSX$LIBUIC does not exist, the symbol is assigned the value [1,1].
<LOGDEV>	Assigned the device name and unit number of the user's login account on multiuser protection systems. For VAX–11 RSX systems, <LOGDEV> is assigned SY0: (the current default device).
<LOGUIC>	For VAX–11 RSX, assigned the current UIC for the process.
<NETUIC>	For VAX–11 RSX, assigned using the system logical name RSX$NETUIC and initially set to [1,54], the UIC in which DECnet-related tasks are stored on the volume. <NETUIC> is used with <SYSUIC> and <LIBUIC> to separate the components of the system. If the logical name RSX$NETUIC is not defined, the symbol is assigned the value [1,54].

Symbol	Value
<NETNOD>	Assigned the DECnet node name of the system. If DECnet is not running on the system, <NETNOD> is assigned VAXAME.
<SYSDEV>	For VAX–11 RSX, the device name that is assigned is SY0:.
<SYDISK>	Assigned the device mnemonic (two letters) of the user's default device (SY:); format is dd.
<SYSID>	Assigned the current base level number of VAX–11 RSX.
<SYSUIC>	For VAX–11 RSX, assigned using the system logical name RSX$SYSUIC, if the logical name exists. If the logical name does not exist, the symbol is assigned the value [1,54].
<TIME>	Assigned the current time; format is hh:mm:ss.
<UIC>	Assigned the current UIC; format is [ggg,mmm].

5.4.2 Numeric Symbols and Expressions

A numeric symbol is a string of digits representing a value in the range of 0 through 177777 (8) (or 0 through 65,535 (10) if immediately followed by a period or if decimal mode has been enabled). If an arithmetic operation yields a result outside of this range, a fatal error occurs and the message

```
IND---Numeric under- or overflow
```

is displayed.

A numeric symbol or constant may be combined with another numeric symbol or constant by a logical or arithmetic operator to form a numeric expression. Arithmetic operators are used to add (+), subtract (–), multiply (*), and divide (/). Logical operators are the inclusive OR (!), logical AND (&), and NOT (#). No embedded blanks or tabs are permitted between operators.

Numeric expressions are evaluated from left to right unless parentheses are used to form subexpressions that are evaluated first. For example, the directive lines:

```
.SETN N1 2
.SETN N2 3
.SETN N3 N1+N2*4
```

assign numeric symbol N3 the value 24 (8) whereas the directive lines

```
.SETN N1 2
.SETN N2 3
.SETN N3 N1+(N2*4)
```

assign numeric symbol N3 the value 16 (8).

Numeric expressions are permitted as second operands in numeric .IF and .SETN directives. They are also permitted as range and default arguments in .ASKN and .ASKS directives. The directives .EXIT and .STOP permit numeric expressions to represent exit status.

With each numeric symbol, Indirect associates a radix—either octal or decimal. The radix of a numeric symbol changes each time the symbol is assigned a new value. If you use a numeric expression to assign a new value to a symbol and all operands in the expression are octal, then the symbol is set to octal. If any operand in the expression is decimal; the symbol is set to decimal. For example:

```
.SETN N1 2           ! N1 is octal
.SETN N2 3.          ! N2 is decimal
.SETN N3 N1+3        ! N3 is octal
.SETN N3 N1+3.       ! N3 is decimal
.SETN N3 N1+N2       ! N3 is decimal
```

You can also assign a new value to a symbol with the .ASKN directive. See 5.6.3 for more information.

The .SETO and .SETD directives allow you to change the radix of a numeric symbol without changing the value of the symbol. For example:

```
.SETN N1 10.        ! N1 = 10 decimal
.SETO N1            ! N1 = 12 octal
```

See 5.6.31 for more information on .SETO and .SETD.

The radix of a numeric symbol does not affect arithmetic operations or comparisons. The radix is important only in substituting a numeric symbol into a string. If the radix of the symbol is octal, the value of the symbol is substituted into the string as an octal number. If the radix is decimal, the value is substituted as a decimal number. For example:

```
.SETN N1 10.                ! N1 = 10 decimal
; N1 = '1'                  ! Displayed as ; N1 = 10
.SETO N1                    ! Make N1 octal
; N1 = 'N1'                 ! Displayed as ; N1 = 12
```

If you substitute a numeric symbol into a string and the substituted number is decimal, a period (.) following the symbol name causes a trailing period to be included in the string (following the substituted number). For example:

```
.SETN N1 10.                ! N1 = 10 decimal
; N1 = 'N1.'                ! Displayed as ; N1 = 10.
.SETO N1                    ! Make N1 octal
; N1 = 'N1.'                ! Displayed as ; N1 = 12
```

You can also force a numeric symbol to be substituted as an octal or decimal number by using a substitution format control string. For example:

```
.SETN N1 10.                ! N1 = 10 decimal
; N1 = 'N1%D'               ! Displayed as ; N1 = 10
; N1 = 'N1%O'               ! Displayed as ; N1 = 12
```

See 5.4.5.1 for more information on substitution format control strings.

5.4.3 String Symbols, Substrings, and Expressions

A string constant is a string of any printable characters enclosed by quotation marks (") or pound signs (#). Empty strings are also permitted. The number of characters cannot exceed 132 (10). For example:

```
"ABCDEF"
  ""
```

String symbols may have the value of any string constant. The value is assigned by a .SETS or .ASKS directive. For example, the directive statements

```
.SETS   S1   "ABCDEF"
.SETS   S2    S1
```

assign string symbol S2 the value of string symbol S1 (that is, ABCDEF).

A substring facilitates the extraction of a segment from the value of a string symbol. You can use substrings only in second operands of .SETS and .IF directives. For example, the directive statements

```
.SETS   S1    "ABCDEF"
.SETS   S2     S1[1:3]
```

assign string symbol S2 the value of string symbol S1 beginning at character one and ending at character three (that is, ABC).

You can also use the syntax [n:*] to extract the characters from position n to the end of the string. For example, the directive statements

```
.SETS  S1  "ABCDEF"
.SETS  S2  S1[3:*]
```

assign string symbol S2 the value CDEF.

You can combine a string constant, symbol, or substring with another string constant, symbol, or substring by the string concatenation operator (+) to form a string expression.

String expressions are permitted as second operands in .SETS and .IF directives in which the first operand is a string symbol. For example, the directive statements

```
.SETS   S1    "A"
.SETS   S2    "CDEF"
.SETS   S3    S1+"B"+S2[1:3]
```

assign string symbol S3 the value of the concatenation of string symbol S1, string constant "B" , and the first three characters of string symbol S2 (that is, ABCDE).

5.4.4 Reserved Symbols

Parameters for a command file can be passed to Indirect for processing. The parameters are stored in the following reserved local symbols:

P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, COMMAN

The symbol COMMAN contains everything in the issuing command line, including the specification for the command file. The symbols P0 through P9 contain individual elements of the command line. The elements are delimited by spaces. P0 contains the name of the command file that is being invoked.

With the .GOSUB directive (see 5.6.16), any parameters to the right of the label and to the left of a comment are transferred to the symbol COMMAN. The value of COMMAN can then be parsed to obtain formal call parameters.

5.4.5 Symbol Value Substitution

Substitution can occur in any command line. Indirect can use the values assigned to logical, numeric, string, or special symbols by replacing a normal parameter (for example, a device unit) with the symbol name enclosed in apostrophes (for example, 'DEV'). When a previous directive has enabled substitution mode (.ENABLE SUBSTITUTION), Indirect replaces the symbol name enclosed in apostrophes with the value assigned to the symbol.

When Indirect encounters an apostrophe, it treats the subsequent text, up to a second apostrophe, as a symbol name. Indirect then searches the table of symbols for the corresponding symbol and substitutes the value of the symbol for the symbol name and surrounding apostrophes in the command line.

For example, the first three lines in the following example appear in an indirect command file. When the processor executes these lines, it displays the last two lines at the entering terminal.

```
.ENABLE SUBSTITUTION
.ASKS DEV MOUNT ON DEVICE?
MOUNT 'DEV' LABEL

> * MOUNT ON DEVICE? [S]: DK2:
> MOUNT DK2: LABEL
```

DK2: was entered in response to the displayed question. This reply assigned the string value DK2: to string symbol DEV. Then when Indirect read:

```
MOUNT 'DEV' LABEL
```

it substituted the value assigned to DEV (that is, DK2:) for 'DEV'. If substitution mode was not enabled, Indirect would simply have passed the line to MCR as it appeared in the command file (that is, MOUNT 'DEV' LABEL).

If substitution mode is enabled, an apostrophe signals the beginning of a string symbol. Thus, to include an apostrophe as text within a command line, rather than as the start of a symbol, you must replace the single apostrophe with two contiguous apostrophes ('').

If substitution mode is enabled, Indirect displays the command file line

```
; DON''T SHOOT
      as
; DON'T SHOOT
```

5.4.5.1 Substitution Format Control

Converting numeric values to strings and placing string and logical values in a substitution operation can be controlled with a format control string. The control string is in the following form:

```
...'symbol%controlstring'...
```

The control string begins with the percent sign (%) and ends with the second of the two apostrophes that denote the substitution operation. The control string consists of one or more of the following characters:

C Compress leading blanks, embedded blanks, and trailing blanks, and remove embedded nulls.

D Force the conversion of a numeric symbol to decimal.

O Force the conversion of a numeric symbol to octal.

S Perform signed conversion for a numeric symbol.

M Perform magnitude conversion for a numeric symbol.

Z Return leading zeros for a positive numeric value.

Rn Right-justify the resulting string, truncating to 'n' decimal characters if necessary.

Ln Left-justify the resulting string, truncating to 'n' decimal characters if necessary.

X Convert the variable to Radix–50 characters.

V If the symbol being substituted is numeric, convert the low byte to its equivalent ASCII character and substitute it.

If the symbol being substituted is a string, convert the first character to its octal representation and substitute it.

As an example, the following command file shows various control strings being used and the results of using the control strings:

```
.ENABLE SUBSTITUTION
.SETS STRING "  A B  CD    "
; STRING = 'STRING%C'      ! Compress spaces
.SETS STRING "ABCD"
; STRING = 'STRING%R5'     ! Right-justify string, truncating to 5
;                          ! characters
; STRING = 'STRING%R3'     ! Right-justify string, truncating to 3
;                          ! characters
.SETN NUMBER 10.
; NUMBER = 'NUMBER%D'      ! Convert numeric symbol to decimal
; NUMBER = 'NUMBER%O'      ! Convert numeric symbol to octal
; NUMBER = 'NUMBER%ZO'     ! Return leading zeros for positive
;                          ! numeric value, convert to octal
; NUMBER = 'NUMBER%ZOR4'   ! Return leading zeros; convert to
;                          ! octal; right-justify, truncating to 4
;                          ! characters
```

When the command file is executed, Indirect displays the text as follows:

```
> @CONTROL RET
> ; STRING = A B CD  ! Compress spaces
> ; STRING =  ABCD   ! Right-justify string, truncating to 5
> ;                  ! characters
> ; STRING = ABC     ! Right-justify string, truncating to 3
> ;                  ! characters
> ; NUMBER = 10      ! Convert numeric symbol to decimal
> ; NUMBER = 12      ! Convert numeric symbol to octal
> ; NUMBER = 000012  ! Return leading zeros for positive
> ;                  ! numeric value, convert to octal
> ; NUMBER = 0012    ! Return leading zeros; convert to
> ;                  ! octal; right-justify, truncating to 4
> ;                  ! characters
```

Indirect does not perform a consistency check on the control string. If you specify conflicting format characters, Indirect uses the last one specified.

5.5 Switches

The indirect command processor accepts five switches: /TR, /MC, /CLI, /LB, and /LO. They are described as follows:

Switch	Function
/[NO]TR	Displays a trace of the indirect command file on the entering terminal. This function is useful for debugging an indirect command file. Each command line, including Indirect directive statements, is displayed. As each Indirect command line is processed, a number representing the nesting depth of the command file is displayed, followed by an exclamation mark and the command line. If the command line causes some action to occur, the next printed line indicates the action; usually, this line consists of the MCR commands issued as a result of the previous directive. The default is /NOTR.
/[NO]MC	Passes commands not processed by Indirect to MCR. This switch is synonymous with /[NO]CLI. The default is /MC.
/[NO]CLI	Passes commands not processed by Indirect to MCR. This switch is synonymous with /[NO]MC. The default is /CLI.
/LB	Indicates that the specified file is a universal library of command procedures and that the specified module is the procedure to be executed. Only RSX–11 compatible file specifications can be used. See Section 2 for more information on file specification formats.

When command procedures, which are indirect command files, are inserted into a universal library with the Librarian Utility Program (see the *RSX–11M/M–PLUS Utilities Manual*), you can then reference them with /LB:module. If you want the universal library to be accessed with the /LB switch, the library must have been created using the RSX–11 Librarian Utility Program (LBR), not the VAX/VMS Librarian.

Command libraries are built by creating a universal library and inserting command files into it. You can then reference the procedures in the library with the following command line:

```
@command-library/LB:module
```

The default file type for a command library is .CLB.

If you do not specify a module (@command-library/LB), Indirect attempts to locate a module called .MAIN..

If you do not specify a library name (@/LB:module), the following actions occur:

- If the command is issued from the terminal or from a file that is not in the library, Indirect ignores the /LB switch and treats the command line as though you had used @module.CMD. Note that if the command is issued from a command file, the default device and directory of the specified module are the same as those for the current file, not necessarily the same as those for the terminal.
- If the command was issued from within a library, the specified module is searched for in the current library.

Switch **Function**

These default actions for an unspecified library allow a collection of procedures to be developed in a given directory with the @/LB:module or .CHAIN /LB:module commands. When the procedures are then placed in a library, no source changes are required.

Example:

The command file PARAM.CMD contains parameter definitions for the .SETN directive and the command file SYSPRC.CMD contains system-specific procedures. Use the following command lines to create the command library and enter the command files into it:

```
LBR> SYSTART.CLB/CR::::UNI:CMD
LBR> SYSTART.CLB=PARAM/IN:PARAM
LBR> SYSTART.CLB=SYSPRC/IN:PROCED
```

You can then use the following command lines to reference the command library modules:

```
@SYSTART/LB:PARAM      !Define global symbols
@SYSTART/LB:PROCED     !Run init procedure
```

DIGITAL supplies a library of command procedures. The library is SYS$LIBRARY:INDSYS.CLB, and it contains the following procedures:

INDCFG	Displays the current build parameters for the running Indirect task.
INDDMP	Dumps the contents of the Indirect symbol table to the terminal.
INDSFN	Returns system configuration information. You must specify one three-letter feature name as a parameter.
INDVFY	Displays the values of all the special symbols.
QIOERR	Returns a string expansion of the <**FILERR**> error codes. You must call this as a nested procedure or subroutine. The string expansion is returned in the special string symbol <**EXSTRI**>.

The following command line shows the format for invoking a command procedure:

```
@SYS$LIBRARY:INDSYS/LB:procedurename
```

Before you attempt to access a command procedure, make sure that INDSYS.CLB is in SYS$LIBRARY:. If it is not in this directory, your system manager must copy the library from the source kit for the system.

/[NO]LO Indicates that when a new command file is executing, it can access the local symbols created by its calling command file, and that any local symbols created by the new command file will be defined as local symbols for the calling command file. The default is /NOLO.

Use any combination of the switches in the command line

```
@filename/switches
```

or the directive statement

```
 CHAIN filename/switches
```

The switches you specify in the terminal command line to initiate Indirect processing are used as defaults when executing the following commands: @filename[/switches] or .CHAIN filename[/switches].

5.6 Indirect Directive Descriptions

Directives must be separated from their arguments and from MCR-specific commands by at least one space. Only one directive per command line is allowed.

You can insert any number of blanks and horizontal tabs in three places of a command line:

- At the start of the command line
- Immediately following the colon (:) of a label
- At the end of the command line
- Between the parameters and keywords of an MCR command line

This allows you to format the command files for readability. The recommended procedure is to begin labels in column 1 and everything else in column 9 (after one horizontal tab).

An important exception is the lines processed between .ENABLE and .DISABLE DATA directives; no blanks or tabs are removed from these lines. For example:

```
        .IFT Z .GOTO 10
        MAC @ASM
.10:    TKB @BLD
        .OPEN DATFIL
        .DATA XXXXX
.ENABLE DATA
This is data
that goes into
the data file.
.DISABLE DATA
        .GOTO 20
```

Note that the .DISABLE DATA statement must begin in column 1 or Indirect will place it in the data file. You can also use the .CLOSE directive in place of .DISABLE DATA. It too must begin in column 1.

5.6.1 Define a Label .label:

Labels always appear at the beginning of the line; a label may be on a line with additional directives and/or an MCR command, on a line with a comment, or on a line by itself. When control passes to a line with a label, the line is processed from the first character after the colon.

Commands do not have to be separated from the label by a space. Only one label is permitted per line. Labels are one through six characters in length and must be preceded by a period and terminated by a colon. A label may contain only alphanumeric characters and/or dollar signs ($).

Labels may also be defined for direct access. Once such a label is found, its position in the command file is saved. This allows subsequent jumps to frequently called labels or subroutines that must be effected quickly. The first statement processed after a jump to a direct access label is the one on the next line. A fixed number of direct access labels are cached by Indirect.

If you define more direct access labels than can be cached, the most recent direct access label replaces the earliest direct access label.

To declare a label for direct access, leave the line following the colon blank.

Example

```
        .IFT A .GOSUB 200
           .
           .
           .
.200:
        .;THIS IS THE START OF A SUBROUTINE
           .
           .
           .
        .RETURN
```

5.6.2 Ask a Question and Wait for a Reply .ASK

The .ASK directive prints a question on the terminal, waits for a reply, and sets a specified logical symbol to the value of true or false, depending on the reply. If the symbol has not already been defined, Indirect makes an entry in the symbol table. If the symbol has been defined, Indirect resets its value (true or false) in accordance with the reply. Indirect exits with a fatal error if the symbol was previously defined as a string or numeric symbol.

Formats (brackets are required syntax):

```
.ASK ssssss txt-strng
.ASK [default:timeout] ssssss txt-strng
.ASK [:timeout] ssssss txt-strng
```

where

ssssss

is the 1- to 6-character symbol to be assigned a true or false value.

txt-strng

is the question or prompt that Indirect displays.

default

is the default response; used if the question is answered with an empty line (null) or if timeout occurs. The default can be <TRUE> or <FALSE> or another logical variable that has been assigned a value.

timeout

is the timeout count. Indirect waits this long for a response and then applies the default answer. The format for timeout is nnu, where nn is the decimal number of time units to wait and u is T (ticks), S (seconds), M (minutes), or H (hours). The timeout count is valid only if timeout mode is enabled (.ENABLE TIMEOUT).

The entire .ASK statement must fit on one command line.

Note that if you omit the default value but specify a timeout count, the colon is required.

When executing an .ASK directive, Indirect displays txt-strng prefixed by an asterisk and suffixed with "? [Y/N]:" . Indirect recognizes five answers:

1 Y <RET> —Set symbol ssssss to true.

2 N <RET> —Set symbol ssssss to false.

3 <RET> —Set symbol to false or to user-specified default value. <RET> indicates the RETURN key.

4 <ESC> —Set symbol ssssss to true and set the special logical symbol <ESCAPE> to true only if escape recognition has been enabled. <ESC> indicates the ESCAPE or ALTMODE key.

5 <CTRL/Z> —If Control-Z mode is enabled, set <EOF> to true and proceed or else exit immediately.

Example

```
.ASK DEV DO YOU WANT TO MOUNT DM1:
```

This directive statement displays

```
> * DO YOU WANT TO MOUNT DM1:? [Y/N]:
```

on the terminal. Symbol DEV will be set to true or false after you type Y, N, the RETURN key, or the ESCAPE key (if escape recognition is enabled).

5.6.3 Ask for Definition of a Numeric Symbol .ASKN

The .ASKN directive prints a request for a numeric value on the terminal, waits for it to be entered, optionally tests the range for the numeric response and/or applies a default value, and sets the specified symbol accordingly. If the symbol has not previously been defined, Indirect makes an entry in the symbol table. If the symbol has already been defined, Indirect resets its value in accordance with the reply. Indirect exits with a fatal error if the symbol was previously defined as a logical or string symbol.

Formats (brackets are required syntax):

```
.ASKN ssssss txt-strng
.ASKN [low:high:default:timeout] ssssss txt-strng
```

where

ssssss

is the 1- to 6-character symbol to be assigned a numeric value.

txt-strng

is the question or prompt that Indirect displays.

low:high

is a numeric expression giving the range for the response.

default

is a numeric expression or symbol giving the default value.

timeout

is the timeout count. Indirect waits this long for a response and then applies the default answer. The format for timeout is nnu, where nn is the decimal number of time units to wait and u is T (ticks), S (seconds), M (minutes), or H (hours). The timeout count is valid only if timeout mode is enabled (.ENABLE TIMEOUT).

The entire .ASKN statement must fit on one command line.

Note that if you omit any of the parameters within the square brackets, any preceding colons are required.

The command line cannot exceed 132 (10) characters in length. When executing an .ASKN directive, Indirect displays txt-strng prefixed by an asterisk and suffixed with [O]: to indicate that the response is considered octal or [D]: to indicate that the response is considered decimal. The reply must be a number either within the specified range or in the range 0 through 177777 (8) (by default) or 0 through 65,535 (10).

If the response is outside the specified range, the message

```
IND---Value not in range
```

is displayed and the query repeated.

If an arithmetic operation yields a result greater than 177777 (8) when computing the actual value of any of the arguments (low, high, or default), a fatal error occurs and the message

```
IND---Numeric under- or overflow
```

is displayed.

If the response is an empty line (null) and a default value (default) was not specified, Indirect applies a default of 0. Note that in this case, the range, if specified, must include 0.

The response may be either octal or decimal; a leading pound sign (#) forces octal, and a trailing period (.) forces decimal. In the absence of both, Indirect applies a default radix. The default radix is decimal if either the range or the default values are decimal expressions (followed by a period). Otherwise, the default radix is octal (unless decimal mode has been enabled). Indirect displays the default type as either [O] or [D].

To force a default decimal radix without specifying a range argument, use the following construction:

```
.ASKN [::0.] A ENTER VALUE
```

or

```
.ENABLE DECIMAL
.ASKN A ENTER VALUE
```

Examples

1

```
.ASKN SYM DEFINE NUMERIC SYMBOL A
```

This directive statement displays the following statement on the terminal:

```
> *DEFINE NUMERIC SYMBOL A [O]:
```

where [O] is the default radix (octal).

Indirect then defines symbol SYM according to the reply entered.

2
```
.ASKN [2:35:16:20S] NUMSYM DEFINE NUMERIC SYMBOL A
```

This directive statement displays the following statement on the terminal:

```
> * DEFINE NUMERIC SYMBOL A [O R:2-35 D:16 T:20S]:
```

in the format

```
[x R:low-high D:default T:timeout]
```

where

x

is O if the default radix is octal or D if it is decimal.

R:low-high

is the specified range.

D:default

is the specified default.

T:timeout

is the specified timeout count before the default answer is applied.

Indirect then checks that the response string is in the specified range.

3
```
.ASKN [NUMSYM+10:45:NUMSYM+10] SYM DEFINE NUMERIC SYMBOL B
```

This directive statement displays (assuming the value of 16 (8) for NUMSYM)

```
> * DEFINE NUMERIC SYMBOL B [O R:26-45 D:26]:
```

5.6.4 Ask for Definition of a String Symbol .ASKS

The .ASKS directive prints a request for a string value to define a specified symbol on the terminal and optionally tests that the number of characters in the response string falls within the specified range. If the symbol has not previously been defined, Indirect makes an entry in the symbol table. If the symbol has already been defined, Indirect resets its value in accordance with the reply. Indirect exits with a fatal error if the symbol was defined previously as a logical or numeric symbol. If the number of characters is out of the specified range, the message

```
IND---String length not in range
```

is displayed and the question repeated.

Formats (brackets are required syntax):

```
.ASKS ssssss txt-strng
.ASKS [low:high:default:timeout] ssssss txt-strng
```

where

ssssss

is the 1- to 6-character symbol to be assigned a string value.

txt-strng

is the question or prompt that Indirect displays.

low:high

is a numeric expression giving the range for the number of characters permitted in the response string.

default

is a string expression or symbol giving the default value.

timeout

is the timeout count. Indirect waits this long for a response, then applies the default answer. The format for timeout is nnu, where nn is the decimal number of time units to wait and u is T (ticks), S (seconds), M (minutes), or H (hours). The timeout count is valid only if timeout mode is enabled (.ENABLE TIMEOUT).

The entire .ASKS statement must fit on one command line.

Note that if you omit any of the parameters within the square brackets, any preceding colons are required.

When executing an .ASKS directive, Indirect displays txt-strng prefixed by an asterisk (*) and suffixes it with [S]:. The reply must be an ASCII character string.

Examples

1

```
.ASKS NAM PLEASE ENTER YOUR NAME
```

This directive statement displays the following statement on the terminal:

```
> * PLEASE ENTER YOUR NAME [S]:
```

Indirect then defines symbol NAM according to the string reply entered.

2

```
.ASKS [1:15::10S] MIDNAM PLEASE ENTER YOUR MIDDLE NAME
```

This directive statement displays the following statement on the terminal:

```
> * PLEASE ENTER YOUR MIDDLE NAME [S R:1-15 T:10S]:
```

in the format

```
[S R:low-high T:timeout]
```

where

S

is the symbol type (string).

R:low-high

is the specified range for number of characters.

T:timeout

is the specified timeout count.

5.6.5 Begin Block .BEGIN

The .BEGIN directive marks the beginning of a Begin-End block. The block must be terminated with an .END directive.

Labels and local symbols defined following the .BEGIN directive are local to the block instead of being used throughout the entire command file. Therefore, labels and local symbols defined inside a block lose definition outside the block. (Labels and symbols defined outside a block retain definition throughout the file.) Labels and symbols defined outside a block and then modified within the block, however, assume and retain the value assigned in the block.

Labels and local symbols defined within a block lose definition with an .ERASE LOCAL directive statement (see 5.6.14) or with the .END directive.

.BEGIN must be the only directive on a command line. For example, the .BEGIN directive cannot appear on the same line as an .IF directive.)

Format:

```
.BEGIN
```

5.6.6 Continue Processing Using Another File .CHAIN

The .CHAIN directive closes the current file, erases all local symbols, clears any .ONERR arguments, empties the direct access label cache, and continues processing using command lines from another file. However, the .CHAIN directive does not close data files or change the nested-file level.

Format (brackets not part of syntax):

```
.CHAIN filename[/switches]
```

where

filename

is the name of the file that contains the new command lines. You must specify an RSX–11 compatible file specification.

/switches

are any of the optional switches described in 5.5.

Example

```
.CHAIN OUTPUT
```

This directive statement transfers control to the file OUTPUT.CMD.

5.6.7 Close Secondary File .CLOSE

The .CLOSE directive closes the secondary file opened by an .OPEN directive.

Format (brackets not part of syntax):

```
.CLOSE [#n]
```

where

#n

is an optional file number in the range 0 to 3. The default is #0. You can substitute a numeric symbol for the value n by enclosing the symbol in apostrophes.

5.6.8 Output Data to Secondary File .DATA

The .DATA directive specifies text that is to be output to a secondary file previously opened by an .OPEN directive.

When Indirect processes the text string that follows the .DATA directive, it ignores the leading space (if present), assuming it to be a separator between the directive and the text string. Any other spaces are transferred to the data file. If a TAB follows the directive, it is transferred to the file. If no other characters follow the directive, a blank line is transferred to the file. This processing has the following results:

Command File	Open File
.DATA foo<RET>	foo<RET>
.DATA foo<RET>	to foo<RET>
.DATA <TAB> foo<RET>	<TAB> foo<RET>
.DATA <TAB> foo<RET>	<TAB> foo<RET>
.DATA <RET>	null line

Note that if a comment follows a .DATA statement (that is, .DATA data !comment), Indirect also outputs the comment to the secondary file because it cannot tell whether the comment pertains to the .DATA statement itself or to the data being output to the file.

Format (brackets not part of syntax):

```
.DATA [#n] txt-strng
```

where

txt-strng

is the text to be output to the secondary file.

#n

is an optional file number in the range 0 to 3. The default is #0. You can substitute a numeric symbol for the value n by enclosing the symbol in apostrophes.

The command line cannot exceed 132 (10) characters and the specified text string cannot continue onto the next line. If a secondary file is not open, an error condition exists; Indirect issues an error message and begins error processing.

Example

```
.SETS SEND "THIS IS DATA"
.OPEN TEMP
.DATA 'SEND'
.CLOSE
```

These directives output THIS IS DATA to the secondary file TEMP.DAT (.DAT is the default file type for a data file).

5.6.9 Decrement Numeric Symbol .DEC

The .DEC directive decrements a numeric symbol by one. Indirect exits with a fatal error if the symbol was defined previously as a logical or string symbol.

Format:

```
.DEC ssssss
```

where

ssssss

is the 1- to 6-character numeric symbol.

Example

```
.DEC X
```

This directive decrements the value assigned to the numeric symbol X by 1.

5.6.10 Delay Execution for a Specified Period of Time .DELAY

The .DELAY directive delays further processing of the file for a specified period of time.

Format:

```
.DELAY nnu
```

where

nn

is the decimal number of time units to delay.

u

is

T - ticks
S - seconds
M - minutes
H - hours

The parameter nn is decimal by default, or octal if preceded by a pound sign (#). For example:

```
10S is 10 (10) seconds
#10S is 10 (8) seconds
```

If quiet mode is disabled when the .DELAY directive is executed, Indirect issues the message

```
IND---Delaying
```

When the time period expires and the task resumes, Indirect issues the message

```
IND---Continuing
```

Example

```
.DELAY 20M
```

This directive statement delays processing for 20 (10) minutes.

5.6.11 Disable Option .DISABLE

The .DISABLE directive disables a specified operating mode previously activated by an .ENABLE directive. See 5.6.12 for operating mode information.

Format:

```
.DISABLE option[,option...]
```

The following is a list of the operating modes that can be disabled:

ATTACH
CONTROL-Z
DATA
DECIMAL
DETACH
DISPLAY
ESCAPE
GLOBAL
LOWERCASE
MCR
OVERFLOW
QUIET
SUBSTITUTION
TIMEOUT
TRACE
TRUNCATE

5.6.12 Enable Option .ENABLE

The .ENABLE directive is used to invoke several operating modes. Each mode is independent of the others; all of them can be active simultaneously. When Indirect starts to process a file, the initial settings are:

ATTACH	enabled	LOWERCASE	enabled
CONTROL-Z	disabled	MCR	enabled
DATA	disabled	OVERFLOW	disabled
DECIMAL	disabled	QUIET	disabled
DETACH	enabled	SUBSTITUTION	disabled
DISPLAY	enabled	TIMEOUT	enabled
ESCAPE	disabled	TRACE	disabled
GLOBAL	enabled	TRUNCATE	disabled

In **ATTACH** mode, Indirect attaches to a terminal when displaying comment lines. In **DETACH** mode, it detaches from the terminal when processing command lines. Enabling both of these modes allows you to type CTRL/O to suppress a lengthy comment.

Enabling **CONTROL-Z** mode allows a command file to detect a CTRL/Z response to a question and continue processing. If Control-Z mode is disabled and you type CTRL/Z in response to an .ASKx question, Indirect exits. If Control-Z mode is enabled, the end-of-file (<EOF>) symbol is set to true and Indirect continues processing the command file.

In **DATA** mode, Indirect outputs lines that follow an .ENABLE DATA directive statement to a secondary file. (The .DATA directive sends a single line of text to a secondary file; see 5.6.8.) To disable data mode, the .DISABLE DATA (or .CLOSE) statement must begin in the first column. Otherwise, Indirect copies the statement itself into the data file. The .ENABLE DATA directive also has an optional argument (#n) that specifies which file the data is to go into.

In **DECIMAL** mode, all numeric symbols are created or redefined by default as decimal instead of as octal.

In **DISPLAY** mode, Indirect displays the current fields for the .ASKx directives and the end-of-file (@ <EOF>) symbol. If display mode is disabled, Indirect displays only the text string for the .ASKx directive and suppresses the end-of-file (@ <EOF>) symbol.

ESCAPE recognition (.ENABLE ESCAPE) permits an escape character to be the response to an .ASK, .ASKN, or .ASKS directive. A question answered with a single escape character sets the special logical symbol <ESCAPE> to true. The escape character must be used only as an immediate terminator to the question; if one or more characters precede the escape character, an error condition exists. In this case, the message

```
IND---Invalid answer or terminator
```

is printed and the question repeated. Note that if you press the ESCAPE key in response to an .ASK directive, the specified logical symbol (ssssss of .ASK ssssss txt-strng) is also set to true.

In **GLOBAL** symbol mode, symbol names that begin with a dollar sign ($) are defined as global to all levels of indirect files; once such a symbol has been defined, all levels recognize it. Symbols that do not begin with a dollar sign are recognized only within the level that defines them.

In **LOWERCASE** mode, characters that are read from the terminal in response to .ASKS directives are stored in the string symbol without lower- to uppercase conversion. The representation of characters is significant when you compare strings (see 5.6.18 since the .IF directive distinguishes between lowercase and uppercase characters).

In **MCR** mode, commands not processed by Indirect are passed to MCR. MCR mode is equivalent to the functions of the /CLI and /MC switches.

OVERFLOW mode allows signed arithmetic in numeric expressions. Enabling the mode provides for numeric expressions and operations that otherwise would result in the "Numeric under- or overflow" error message.

In **QUIET** mode, the processor does not echo MCR command lines or comments. The command lines are executed normally and, if they return a message or display, the message or display is printed on the terminal.

In **SUBSTITUTION** mode, Indirect substitutes a string for a symbol. The symbol must begin and end in apostrophes ('symbol'). For example, if the symbol A has been assigned the string value THIS IS A TEST, then every 'A' will be replaced by THIS IS A TEST. When substitution mode is enabled, Indirect performs substitutions in each line before scanning the line for directives and MCR commands. (While obeying a .GOTO label directive, however, Indirect ignores any undefined symbols encountered before the target line, that is, the line containing the specified label.)

In **TIMEOUT** mode, Indirect uses the timeout parameters specified with the .ASKx directives. Indirect waits for the timeout count to elapse and then applies the default answer to the directives. Timeout mode (the default) must be enabled to use the timeout counts for the .ASKx directives.

In **TRACE** mode, command lines that Indirect has processed are displayed on the terminal. As each line is processed, it is displayed with its nesting level and an exclamation mark (!). Trace mode is equivalent to the function of the /TR switch.

In **TRUNCATE** mode, Indirect ignores any truncate errors on a .READ directive. A truncate error occurs when a line in a file is too long. If the full record cannot fit within the 132 (10)-character limit of the symbol, the record is truncated.

Formats (brackets not part of syntax):

```
.ENABLE option[,option...]
.ENABLE DATA [n]
```

where

#n

is an optional file number in the range 0 to 3. The default is #0. You can substitute a numeric symbol for the value n by enclosing the symbol in apostrophes.

Examples

```
1     .ENABLE CONTROL-Z
      .ASK RESP DO YOU WISH TO CONTINUE
      .IFT <EOF> .GOTO CLENUP
      .IFF RESP .GOTO CLENUP
```

CONTROL-Z mode:

If you type CTRL/Z in response to the question, <EOF> is set to true and Indirect transfers to CLENUP.

2
```
;IF YOU WANT A LIST OF OPTIONS, TYPE <ESC>.
.ENABLE ESCAPE
.ASKS A ENTER OPTION
.IFT <ESCAPE> .GOTO LIST
        .
        .
        .
.LIST: ;OPTIONS ARE: A (ADD), S (SUBTRACT), M (MULTIPLY)
```

ESCAPE recognition mode:

If you press the ESCAPE key in response to ENTER OPTION, the lines displayed at the terminal are:

```
> ;IF YOU WANT A LIST OF OPTIONS, TYPE <ESC>.
> * ENTER OPTION [S]: <ESC>
> ;OPTIONS ARE: A (ADD), S (SUBTRACT), M (MULTIPLY)
```

3
```
.ENABLE GLOBAL
.SETS $X "TEST"
```

GLOBAL symbol mode:

These two lines appear in an indirect command file called TEST1.

A file called TEST2.CMD contains the following lines:

```
 ENABLE SUBSTITUTION
@TEST1
MAC '$X','$X'='$X'
```

Therefore, MCR displays the following when the file TEST2.CMD is run:

```
> MAC TEST,TEST=TEST
> @ <EOF>
```

4
```
.ASK QUIET DO YOU WANT COMMAND LINES SUPPRESSED
.IFT QUIET .ENABLE QUIET
.IFF QUIET .DISABLE QUIET
SHOW DEFAULT
```

QUIET mode:

If the response is affirmative, Indirect displays the default but not the SHOW DEFAULT command. For example:

```
> DO YOU WANT COMMAND LINES SUPPRESSED? [Y/N]: Y
DRA1:[JOHNSON.CATHY]
```

5
```
.ENABLE SUBSTITUTION
.ASKS FIL SPECIFY SOURCE FILE
MAC  'FIL'='FIL'
```

SUBSTITUTION mode:

When the file is executing, the corresponding lines displayed at the terminal are:

```
> * SPECIFY SOURCE FILE [S]: SOURCE
> MAC SOURCE=SOURCE
```

5.6.13 End Block .END

The .END directive marks the end of the Begin-End block. If Indirect encounters more .END directives than .BEGIN directives, command processing terminates and the following message is displayed:

```
IND---Illegal nesting
```

Format:

```
.END
```

as the only directive on the line.

5.6.14 Delete Symbols .ERASE

The .ERASE directive deletes all local or global symbol definitions, or a specific global symbol definition. When you define a symbol, either locally (by defining a symbol value) or globally (by enabling global symbol mode and preceding the symbol name with a dollar sign ($)), Indirect creates an entry in the symbol table. The .ERASE directive erases either all local or all global entries, or a specific global entry, in the table.

Following an .ERASE directive, you can redefine symbol values as well as symbol type.

Formats:

```
.ERASE LOCAL
.ERASE GLOBAL
.ERASE SYMBOL global-symbol
```

An .ERASE LOCAL directive outside of a Begin-End block erases all local symbols defined within the current file.

An .ERASE LOCAL directive within a Begin-End block erases only those local symbols defined within the block.

However, note that the following actions also occur:

1. Local symbols defined within a nested file are erased when that file exits.
2. Local symbols defined within a Begin-End block are erased with .END.
3. Local symbols defined outside of Begin-End blocks are visible, modifiable, and not erasable within a Begin-End block.

An .ERASE GLOBAL, either outside or inside a Begin-End block, erases all global symbols.

An .ERASE SYMBOL global-symbol erases the specified global symbol. (Individual local symbols are not erasable.)

Examples

1

```
.ERASE LOCAL
```

This directive erases all local symbol definitions used in the indirect command file.

2

```
.ERASE SYMBOL $SWITC
```

This directive erases the single global symbol "$SWITC."

5.6.15 Exit Current Command File .EXIT

The .EXIT directive terminates processing of the current command file or Begin-End block and returns control to the previous-level command file or, if the directive is executed within a block, to the line following the .END directive. If the directive is encountered at the uppermost indirect nesting level, Indirect exits and passes control to the CLI (see the .STOP directive, 5.6.33).

The .EXIT directive also optionally allows you to specify a value to copy into the special symbol <EXSTAT>.

Format (brackets not part of syntax):

```
.EXIT [value]
```

where value is an optional numeric expression to be copied to the special symbol <EXSTAT>.

Example

```
@TEST2
```

This line appears in an indirect command file called TEST1.

The file TEST2.CMD contains the following line:

```
.EXIT
```

When Indirect encounters the .EXIT directive in TEST2, control returns to TEST1.CMD.

If the .EXIT directive in TEST2.CMD includes a numeric expression, for example,

```
.EXIT N+2
```

Indirect evaluates the expression and copies the value into <EXSTAT>.

5.6.16 Call a Subroutine .GOSUB

The .GOSUB directive saves the current position in an indirect command file and then branches to a label. The label identifies an entry point to a subroutine that is terminated by a .RETURN directive.

When you issue a .GOSUB directive from within a Begin-End block, Indirect saves the current block context and then scans down the file searching for the first occurrence of the subroutine label. Note that during the scan, Indirect ignores any intervening .BEGIN or .END directives. The .RETURN directive restores previous block context. Thus, the subroutine can be contained within a Begin-End block.

The maximum nesting depth for subroutine calls is eight on VAX–11 RSX systems.

Format:

```
.GOSUB label parameters
```

where

label

is the label that designates the first line of a subroutine, but without the leading period and trailing colon. Any parameters to the right of the label and to the left of a comment are transferred to the reserved local symbol COMMAN. The value of COMMAN can then be parsed with the .PARSE directive (see 5.6.25) to obtain formal call parameters.

Example

```
.GOSUB EVAL
```

This directive statement transfers control to the subroutine labeled .EVAL:.

5.6.17 Branch to a Label .GOTO

The .GOTO directive causes a branch from one line in an indirect command file to another. All commands between the .GOTO directive and the specified label are ignored. Branches can go forward or backward in the file.

The target of a .GOTO branch from within a Begin-End block must be contained in that block. The .GOTO directive cannot branch into another block. When Indirect encounters a .GOTO directive within a Begin-End block, it searches for the specified label in that block. Since Indirect only searches the one Begin-End block, you can use the same label more than once in a command file.

Format:

```
.GOTO label
```

where

label

is the name of the label, but without the leading period and trailing colon.

Example

```
.GOTO 100
```

This directive statement transfers control to the line containing the label .100:.

5.6.18 Logical Test .IF

A number of directives make tests; if the test is true, Indirect processes the remainder of the command line. Logical tests can be combined into a compound logical test by using the .AND and .OR directives.

5.6.18.1 Test Whether Symbol Meets Specified Condition (.IF)

The .IF directive compares a numeric or string symbol with another expression of the same type to determine whether one of several possible conditions is true. If the condition is satisfied, Indirect executes the remainder of the command line.

When comparing a string symbol with a string expression, Indirect compares the ASCII values of each operand's characters (from left to right) one by one. An operand is considered greater if the first nonequal character has a greater value than the corresponding character in the other operand. Numeric symbols are compared strictly on the basis of magnitude.

Format:

```
.IF symbol relop expr directive-statement
```

where

symbol

is the 1- to 6-character numeric or string symbol.

relop

is one of the following relational operators:

EQ or =	Equal to
NE or <>	Not equal to
GE or >=	Greater than or equal to
LE or <=	Less than or equal to
GT or >	Greater than
LT or <	Less than

expr

is an expression of the same type as symbol.

directive-statement

is the Indirect command line to be processed if the condition is satisfied.

Examples

1

```
.SETS X "A"
.SETS Y "a"
.IF X LT Y .GOTO 200
```

The ASCII value of string symbol X is less than the ASCII value of string symbol Y, which satisfies the less-than condition. Thus, control passes to the line containing the label .200:.

2

```
.SETN N1 2
.SETN N2 7
.IF N1 <= N2 PIP /LI
```

With the condition satisfied (numeric symbol N1 less than or equal to numeric symbol N2), the PIP command is processed.

3

```
.SETS S1 "AAb"
.SETS S2 "AA"
.SETS S3 "BBBB"
.IF S1 >= S2+S3[1:1] .INC N
```

Since the condition that string symbol S1 is greater than or equal to string symbol S2 concatenated with the first character of string symbol S3 (AAb >= AAB) is satisfied, Indirect increments numeric symbol N.

5.6.18.2 Test Whether Task Is Active or Dormant (.IFACT/.IFNACT)

The .IFACT or .IFNACT directive tests whether a task is active (.IFACT) or dormant (.IFNACT). If the test is true, the rest of the command is processed. For VAX–11 RSX, a task is always inactive; that is, .IFACT is false and .IFNACT is true. Therefore, Indirect assumes the dormant condition. These directives have standard responses because they test for conditions not defined in the VAX–11 RSX environment.

Formats:

```
.IFACT  taskname directive-statement
.IFNACT  taskname directive-statement
```

where

taskname

is a 1- to 6- character legal task name.

directive-statement

is the Indirect command line to be processed if the condition is satisfied.

Examples

```
1    .IFACT   REPORT  .GOTO 350
```

```
2    .IFNACT  REPORT  RUN REPORT
```

5.6.18.3 Test Whether Symbol Is Defined or Not Defined (.IFDF/IFNDF)

The .IFDF or .IFNDF directive tests whether a logical, numeric, or string symbol has been defined (.IFDF) or not defined (.IFNDF). If the test is true, the rest of the command line is processed. This directive does not test the value of the symbol.

Formats:

```
.IFDF  ssssss  directive-statement
.IFNDF  ssssss  directive-statement
```

where

ssssss

is the 1- to 6-character symbol being tested. The symbol can be local, global, or an Indirect special symbol.

directive-statement

is the Indirect command line to be processed if the condition is satisfied.

Examples

```
1   .IFDF  A  .GOTO 100


2   .IFNDF  A  .ASK A DO YOU WANT TO SET TIME
```

5.6.18.4 Test Whether Task Is Installed or Not Installed (.IFINS/.IFNINS)

The .IFINS or .IFNINS directive tests whether a task is installed (.IFINS) or not installed (.IFNINS) in the system. If the test is true, the rest of the command line is processed. For VAX–11 RSX, a task is always installed; that is, .IFINS is true and .IFNINS is false. These directives have standard responses because they test for conditions that are not defined in the VAX–11 RSX environment.

Formats:

```
.IFINS  taskname  directive-statement
.IFNINS  taskname  directive-statement
```

where

taskname

is a 1- to 6-character task name.

directive-statement

is the Indirect command line to be processed if the condition is satisfied.

Examples

```
1   .IFINS  PIP  .GOTO 250


2   .IFNINS  PIP  INS $PIP
```

5.6.18.5 Test Whether Mode Is Enabled or Disabled (.IFENABLED/.IFDISABLED)

The .IFENABLED or .IFDISABLED directive tests whether an operating mode has been enabled with the .ENABLE directive or disabled with the .DISABLE directive. (See the description of the .ENABLE directive in 5.6.12 for the list of operating modes.)

Formats:

```
.IFENABLED option directive-statement
.IFDISABLED option directive-statement
```

where

option

is the same operating mode option (with the exception of DATA) used with the .ENABLE or .DISABLE directive, or FULL-DUPLEX, POTASK, or FMS.

directive-statement

is the Indirect command line to be processed if the condition is satisfied.

5.6.18.6 Test Whether Driver Is Loaded or Not Loaded (.IFLOA/.IFNLOA)

The .IFLOA or .IFNLOA directive tests whether a driver is loaded (.IFLOA) or not loaded (.IFNLOA) in the system. If the test is true, the rest of the command line is processed. For VAX–11 RSX, a driver is always loaded; that is, .IFLOA is true and .IFNLOA is false. These directives have standard responses because they test for conditions not defined in the VAX–11 RSX environment.

Formats:

```
.IFLOA dd: directive-statement
.IFNLOA dd: directive-statement
```

where

dd:

is a device driver.

directive-statement

is the Indirect command line to be processed if the condition is satisfied.

Examples

1
```
.IFLOA DK: .GOTO 250
```

2
```
.IFNLOA DK: LOA DK:
```

5.6.18.7 Test Whether Symbol Is True or False (.IFT/.IFF)

The .IFT or .IFF directive tests whether a logical symbol is true or false. If the test is true, Indirect processes the remainder of the command line.

Indirect exits with a fatal error if the symbol being tested was previously defined as a numeric or string symbol.

Formats:

```
.IFT  ssssss  directive-statement
.IFF  ssssss  directive-statement
```

where

ssssss

is the 1- to 6-character logical symbol being tested.

directive-statement

is the Indirect command line to be processed if the condition is satisfied.

Examples

1
```
.IFT  A  .GOTO  100
```

2
```
.IFF  B  .GOTO  200
```

5.6.18.8 Compound Tests

You can combine .IF tests by using the .AND and .OR directives. In addition, an implied .AND is effected when more than one .IF appears on the same line without being separated by an .AND directive.

The .AND directive takes precedence over the .OR directive as shown in the following example:

```
.IFT A .OR .IFT B .AND .IFT C .GOTO D
```

That is, Indirect reads the line as:

```
.IFT A .OR (.IFT B .AND .IFT C) .GOTO D
```

Examples

1
```
.IFT A .AND .IFF B  .GOTO HELP
```

If the logical symbol A is true and the logical symbol B is false, control passes to the line containing the label .HELP:.

2
```
.IFT A .IFF B  .GOTO HELP
```

Same effect as the previous directive (.AND implied).

3
```
.IFT A .OR  .IFF B RUN PIP
```

If the logical symbol A is true or if the logical symbol B is false, the RUN command is issued.

5.6.19 Increment Numeric Symbol .INC

The .INC directive increments a numeric symbol by 1. Indirect exits with a fatal error if the symbol was previously defined as a logical or string symbol.

Format:

```
.INC ssssss
```

where

ssssss

is the 1- to 6-character numeric symbol being incremented.

Example

```
.INC B
```

Increment by 1 the value assigned to the numeric symbol B.

5.6.20 Define Logical End-of-File /

The logical end-of-file directive (/) terminates file processing and exits. The message

```
> @ <EOF>
```

is then displayed (if display mode has not been disabled).

Format:

```
/
```

as the first nonblank character on a line.

You can use this directive at any location in the command file to terminate file processing quickly, but care should be taken to avoid an inadvertent exit.

Example

```
       .ASK CONT DO YOU WISH TO CONTINUE
       .IFT CONT .GOTO 100
       /
.100:
```

5.6.21 Branch to Label on Detecting an Error .ONERR

If Indirect detects one of the following errors:

- Undefined symbol
- Bad syntax
- Unrecognized command
- String substitution error
- Symbol type error (.IF, .IFT, .IFF, .INC, .DEC)
- Redefinition of a symbol to a different type (.ASK, .ASKN, .ASKS, .SETT, .SETF, .SETL, .SETN, .SETD, .SETO, .SETS)
- Data file error (.OPEN, .OPENA, .OPENR, .DATA, .CLOSE, or .READ between .ENABLE DATA and .DISABLE DATA)

control passes to the line containing the specified label. This feature provides you with a means of gaining control to terminate command file processing in an orderly manner.

Note that the .ONERR directive applies only to the error conditions listed; errors returned from a task external to Indirect (for example, an MCR syntax error) are not processed by the .ONERR directive.

Format:

```
.ONERR  label
```

where

label

is the name of the label, but without the leading period and trailing colon.

Upon detecting an error, the processor passes control to the line starting with .label:. The .ONERR directive must be issued before Indirect encounters the error condition. If the directive is executed (that is, if one of the listed errors is encountered), error processing passes to the specified label. If the label specified by the .ONERR directive does not exist and an error condition has occurred, command processing terminates.

Once an .ONERR condition has occurred, another .ONERR directive must be issued to trap a future error.

The .ONERR directive works with the special symbol <ERRCTL> (see 5.4.1.2). For each class of error that an .ONERR target routine processes, the appropriate bit is set in the symbol. The initial default value for <ERRCTL> is 1, which implies that only class 1 errors can be handled with an .ONERR routine. (Note that if you attempt to process errors other than default class 1, Indirect cannot continue in most cases. The error service routine is limited to a fatal error message and .EXIT. The internal state of Indirect is indeterminate in all but class 1 error cases.) See Section 9 for a list of error messages and their assigned class values.

Example

```
.ONERR  100
```

Upon detecting one of the error conditions, Indirect passes control to the line labeled .100:.

5.6.22 Open Secondary File .OPEN

The .OPEN directive opens a specified secondary file as an output file. The .DATA directive is used to place data in this secondary file.

Format (brackets not part of syntax):

```
.OPEN [#n] filename
```

where

filename

is a file to be opened as an output file. The default file type is .DIND. You must specify an RSX–11 compatible file specification.

#n

is an optional file number in the range 0 to 3. The default is #0. You can substitute a numeric symbol for the value n by enclosing the symbol in apostrophes.

Note that you cannot include a comment that begins with a semicolon (;comment) in an .OPEN statement. Doing so results in a syntax error. (Comments that begin with an exclamation mark (!comment) are accepted.)

Example

```
.OPEN SECOUT
```

This directive opens the file SECOUT.DAT as an output file.

5.6.23 Open Secondary File for Append .OPENA

The .OPENA directive opens a secondary file and appends all subsequent data to the file.

Format (brackets not part of syntax):

```
.OPENA [#n] filename
```

where

filename

is a secondary file to be opened with subsequent data appended to it. The default file type is .DIND. You must specify an RSX–11 compatible file specification.

#n

is an optional file number in the range 0 to 3. The default is #0. You can substitute a numeric symbol for the value n by enclosing the symbol in apostrophes.

Note that you cannot include a comment beginning with a semicolon (;comment) in an .OPENA statement. Doing so results in a syntax error. (Comments that begin with an exclamation mark (!comment) are accepted.)

If the specified file does not already exist, .OPENA becomes the .OPEN directive by default.

Example

```
.OPENA SECOUT
```

This directive opens the file SECOUT.DAT as an output file and appends subsequent data to it.

5.6.24 Open File for Reading .OPENR

The .OPENR directive opens a file for reading with the .READ directive.

Format (brackets not part of syntax):

```
.OPENR [#n] filename
```

where

filename

is a file to be opened for reading. The default file type is .DIND. You must specify an RSX–11 compatible file specification.

#n

is an optional file number in the range 0 to 3. The default is #0. You can substitute a numeric symbol for the value n by enclosing the symbol in apostrophes.

Note that you cannot include a comment beginning with a semicolon (;comment) in an .OPENR statement. Doing so results in a syntax error. (Comments that begin with an exclamation mark (!comment) are accepted.)

Examples

1
```
.OPENR INDADD
```

This directive opens the file INDADD.DAT for reading with the .READ directive.

2
```
.OPENR DATLIB.ULB/LB:DATINP
```

This directive opens for reading the library module DATINP that is contained in the universal library DATLIB.

5.6.25 Parse Strings into Substrings .PARSE

The .PARSE directive parses strings in a command line into substrings.

Format:

```
.PARSE <string> <control-string> <var1> <var2> ... <varn>
```

The string is broken up into substrings as specified by the control string. The substrings are stored in the specified variables. The first character of the control string delimits the first substring, the second character of the control string delimits the second substring, and so on. The last character of the control string is repeated if the number of variables exceeds the length of the control string. If you specify more variables than substrings, the additional variables are set to null strings. If you specify fewer variables than the number of substrings that can be parsed, the last variable contains the unparsed fragment of <STRING>.

The symbol <STRLEN> contains the actual number of substrings that Indirect processed (including explicit null substrings).

Example

```
.PARSE COMMAN " ," FILE A1 A2 A3 A4 A5
```

Given that COMMAN contains "TESTFILE IND,MCR,,LOA" , this directive has the following results:

```
FILE = TESTFILE
A1   = IND
A2   = MCR
A3   = null
A4   = LOA
A5   = null
<STRLEN> contains a 5.
```

5.6.26 Pause for Operator Action .PAUSE

The .PAUSE directive interrupts processing of an indirect command file to wait for user action. A .PAUSE directive causes Indirect to stop itself. You can then perform some operations and subsequently cause the task to resume.

Format:

```
.PAUSE
```

When Indirect stops itself, it displays the following message on the entering terminal:

```
IND---Pausing. To continue type RES
```

where RES is the command to be issued to resume the task.

You then type the appropriate command line to resume the task. Indirect displays the message:

```
IND---Continuing
```

and it continues processing where it left off.

During a .PAUSE state, any commands you enter must be MCR commands.

5.6.27 Read Next Record .READ

The .READ directive reads the next record into a specified string variable. The entire record is written into the variable. If the record is longer than 132 (10) characters, an error occurs.

After every .READ operation, the special symbol <FILERR> contains the FCS-11 file code for the read, and the special symbol <EOF> reflects whether an end-of-file was found. (Note that .OPENR does not clear <EOF>.) If an error or end-of-file occurs, the string variable remains unchanged from its previous state.

Format (brackets not part of syntax):

```
.READ [#n] ssssss
```

where

#n

is an optional file number specifying the file from which the record is to be read. The file number must be one of the numbers used in a previous .OPENR statement.

ssssss

is the string variable into which the record will be read.

Example

```
          .ENABLE SUBSTITUTION
          .OPENR FILE
          IF <FILERR> NE 1 .GOTO ERROR
.LOOP:
          .READ RECORD
          .IFT <EOF> .GOTO DONE
          .IF <FILERR> NE 1 .GOTO ERROR
           ; 'RECORD'
          .GOTO LOOP
             .
.ERROR:      .
             .
.DONE:    .CLOSE
             .
             .
             .
```

These directives open the file FILE.DAT for reading, read each record into the string variable RECORD, display each record on the terminal, and close the file.

5.6.28 Return from a Subroutine .RETURN

The .RETURN directive signifies the end of a subroutine and returns control to the line immediately following the .GOSUB directive that initiated the subroutine.

Format:

```
.RETURN
```

5.6.29 Set Symbol to True or False .SETT/.SETF/.SETL

The .SETT, .SETF, and .SETL directives define or change the value of a specified logical symbol. If the symbol has not been defined, Indirect makes an entry in the symbol table and sets the logical symbol to the value specified. If the symbol has already been defined, Indirect resets the symbol accordingly. Indirect exits with a fatal error if the logical symbol was defined previously as a numeric or string symbol.

Formats:

```
.SETT  ssssss
.SETF  ssssss
.SETL  ssssss  llllll
```

where

ssssss

is the 1- to 6-character logical symbol to be assigned a true or false value.

llllll

is a logical or numeric expression. ssssss is assigned the value of llllll when the logical expression is evaluated.

Examples

```
1    .SETT  X
```

This directive sets the logical symbol X to true.

```
2    .SETF  ABCDE
```

This directive sets the logical symbol ABCDE to false.

```
3    .SETL  TEST SWITCHA!SWITCHB
```

This directive sets the logical symbol TEST to true if SWITCHA or SWITCHB is true.

5.6.30 Set Symbol to Numeric Value .SETN

The .SETN directive defines or changes the value of a specified numeric symbol. If the symbol has not been defined, Indirect makes an entry in the symbol table and sets the symbol to the numeric value specified. If the symbol has already been defined, Indirect resets the symbol accordingly. Indirect exits with a fatal error if the numeric symbol was previously defined as a logical or string symbol.

Format:

```
.SETN ssssss numexp
```

where

ssssss

is the 1- to 6-character numeric symbol.

numexp

is a numeric expression. (See 5.4.2.)

When specifying a numeric value to assign to a symbol, you may combine a numeric symbol or constant with another numeric symbol or constant to form a numeric expression. If numeric expressions are used, no embedded blanks or tabs are permitted. Evaluation is done from left to right unless parentheses are used to form subexpressions that are evaluated first. The radix of an expression is octal if all the operands are octal and decimal mode has not been enabled; otherwise the radix is decimal.

Examples

```
1    .SETN NUMBER 27
```

This directive assigns the value 27 (8) to the numeric symbol NUMBER.

```
2    .SETN A1 3*(A2-5)
```

This directive assigns the value of symbol A2 minus 5 multiplied by 3 to the numeric symbol A1.

5.6.31 Set Symbol to Octal or Decimal .SETO/.SETD

The .SETO and .SETD directives redefine the radix of a specified numeric symbol. If the symbol has not been defined, Indirect makes an entry in the symbol table and sets the symbol to the specified radix with a value of 0. If the symbol has already been defined, Indirect resets the symbol accordingly. Indirect exits with a fatal error if the symbol was previously defined as a logical or string symbol.

Formats:

```
.SETO ssssss
.SETD ssssss
```

where

ssssss

is the 1- to 6-character numeric symbol to be assigned an octal or decimal radix.

Example

```
.SETN A 10    ; Sets symbol A to 10 (8)
.SETD A       ; Defines A as a decimal radix symbol with a value
                of 8 (10)
.SETO A       ; Defines A back to original radix with a value
                of 10 (8)
```

5.6.32 Set Symbol to String Value .SETS

The .SETS directive defines or changes the string value of a specified string symbol. If the symbol has not been defined, Indirect makes an entry in the symbol table and sets the symbol to the specified string value. If the symbol has been defined, Indirect resets the symbol accordingly. Indirect exits with a fatal error if the symbol was defined previously as a logical or numeric symbol.

Format:

```
.SETS ssssss strexp
```

where

ssssss

is the 1- to 6-character string symbol.

strexp

is any string expression. (See 5.4.3.)

Indirect assigns the string value represented by the string expression strexp to the specified symbol. If a string constant is used in strexp, the constant must be enclosed by quotation marks ("constant").

You can combine a string symbol, constant, or substring with another string symbol or substring by the string concatenation operator (+) to form a string expression.

Examples

```
1    .SETS A "ABCDEF"
```

This directive assigns the string value ABCDEF to the string symbol A.

```
2    .SETS STR2 "ZZZ"
```

This directive assigns the value ZZZ to the string symbol STR2.

```
3    .SETS X STR2+"ABC"
```

This directive assigns the value of symbol STR2 plus ABC (that is, ZZZABC) to the string symbol X.

```
4    .SETS X STR2+A[1:3]
```

This directive is equivalent to the previous directive; it assigns the string symbol X the string value of STR2 plus the first three characters of string A (that is ZZZABC).

```
5    .SETS MYFILE <UIC>+"MYFILE.TXT"
```

This directive assigns the string symbol MYFILE the string value of the current UIC and the string contained within the quotation marks. (For example, if the current UIC is [303,3], MYFILE is assigned the string value [303,3]MYFILE.TXT.)

5.6.33 Terminate Command File Processing .STOP

The .STOP directive immediately terminates command file processing and exits. The message

```
> @ <EOF>
```

is then displayed (unless .DISABLE DISPLAY is in effect).

The .STOP directive optionally allows you to set the exit status for Indirect execution.

Format (brackets not part of syntax):

```
.STOP [value]
```

where

value

is an optional numeric expression to serve as the exit status for Indirect. If you do not specify an exit status value, the .STOP directive is identical to the logical end-of-file directive (/).

Example

```
.STOP 0
```

This directive terminates command file processing and sets the exit status for Indirect to 0.

5.6.34 Test Symbol .TEST

The .TEST directive has two different functions. It tests a variable and sets various special symbols accordingly, and it does substring searches and sets the special symbol <STRLEN> accordingly.

Format 1:

```
.TEST ssssss
```

where

sssss

is the 1- to 6-character symbol to be tested.

The results of the test are as follows:

- If variable is a string, <SYMTYP> is set to 4 and <STRLEN> contains the length of the string. Also, the special symbols <ALPHAN>, <NUMBER>, <RAD50>, and <OCTAL> are set based on a scan of the characters of variable.
- If variable is numeric, <SYMTYP> is set to 2.
- If variable is octal, <SYMTYP> is set to 2 and <OCTAL> is set to TRUE.
- If variable is logical, <SYMTYP> is set to 0.

Format 2:

```
.TEST string substring
```

where

string

is a string symbol or constant.

substring

is a string expression.

In this case, the search for the substring takes place in the specified string. If the substring is present, <STRLEN> is set to the position of the starting character of the substring within the string. If substring is not present, <STRLEN> is set to 0.

Examples

1
```
.TEST SUM
```

If SUM is a string symbol, this directive statement sets <SYMTYP> to 4 and places the number of characters represented by the symbol SUM into <STRLEN>.

2
```
.SETS MAIN "ABCDEF"
.TEST MAIN "C"
```

This directive statements set <STRLEN> to 3, the position of C in the string ABCDEF.

5.6.35 Test Device .TESTDEVICE

The .TESTDEVICE directive allows a command file to acquire information about any device in the system. The information, including error indications, is contained in the string symbol <EXSTRI>. Each device attribute in the string is separated by a comma (which allows processing by the .PARSE and .TEST directives). The first field of the string is the full physical name of the device. The next four fields are octal representations of the device-characteristics words (U.CW1 through U.CW4 of the Unit Control Block). Additional fields contain more information about the device.

Note that only RSX–11 compatible device specifications are allowed. The device specification can specify either a physical device or a logical name that has the format of an RSX–11 device specification.

Format:

```
.TESTDEVICE dd[nn]:
```

where

dd[nn]:

is the device about which the command file is requesting information.

The information stored in <EXSTRI> is in the following form:

```
ddnn:,xx,xx,xx,xx,atr,atr...,atr,
```

where

ddnn:

is the physical device name for the device specified in the command line.

xx,xx,xx,xx

are the four device-characteristics words in octal notation. For VAX–11 RSX systems, device-characteristic words 2 and 3 are a VAX/VMS device-dependent longword.

atr

is one or more of the following device attributes:

NSD	"No such device" is configured into this system.
LOD	The device driver is loaded.
UNL	The device driver is not loaded.
ONL	The device is on line.
OFL	The device is off line.
MTD	The device is a mountable volume and is mounted.
NMT	The device is not a mountable volume or is not mounted.
FOR	The device is a mountable volume and is mounted foreign.
NFO	The device is not a mountable volume or is not mounted foreign.

PUB	The device is a public device.
NPU	The device is not a public device.
ATT	The device is attached to another task.
ATU	The device is attached to this copy of Indirect.
NAT	The device is not attached.
ALO	The device is allocated to another terminal; the value "NSD" (no such device) is returned.
ALU	The device is allocated to this terminal.
NAL	The device is not allocated.

The symbol <EXSTRI> contains the value "NSD," (no such device) if the device is not present in the current system configuration.

Example

```
.TESTDEVICE SY:
```

This directive acquires information about user logical device SY: and stores it in <EXSTRI>.

5.6.36 Test a File .TESTFILE

The .TESTFILE directive determines whether a specified file exists or the directive determines the physical device associated with a logical name. (That is, the directive performs device translation.)

If you specify a file in the command line, the results of a .TESTFILE operation are contained in the symbols <FILSPC> and <FILERR>. <FILSPC> contains the fully qualified file specification and <FILERR> contains the FCS status code resulting from the search for the file. You must specify a file that is in the RSX-11 compatible format (see Section 2).

If the file specification contains a syntax error, Indirect returns the FCS error code for a bad file name (–54).

If you do not specify a file in the command line, Indirect only performs device translation.

Formats:

```
.TESTFILE filespec
.TESTFILE ll:
```

where

filespec

is the file to be tested.

ll:

is the logical name assigned to be translated to a physical device.

Examples

1 .TESTFILE MP:IND.MAP

This directive assigns the following values if the file exists:

```
<FILERR> = 1
<FILSPC> = DR1:[101,300]IND.MAP;4 (MP: is the logical name
          assigned to the physical device DR1:.)
```

If the file does not exist, the directive assigns the following values:

```
<FILERR> = 346 (230 (10))
<FILSPC> = DR1:[101,300]IND.MAP;0
```

2 .TESTFILE TI:

This directive translates the logical name TI: into its physical device name. It assigns the symbol values as follows:

```
<FILERR> = 1
<FILSPC> = TT23:.DAT;0
```

5.6.37 Test a Partition .TESTPARTITION

The .TESTPARTITION directive allows a command file to acquire information about a partition in the system. Partitions do not exist in VAX–11 RSX systems; therefore, this directive is set to always return standard information (in the special symbol <EXSTRI>) in the following format:

```
partition-name,400,size,SYS.
```

where name is the name you gave to the command, 400 is the base, size is the size of the current task including libraries (as a multiple of 32 words (ICM)), and SYS is for system-controlled partitions.

Format:

```
.TESTPARTITION partition-name
```

where

partition-name

is a 1- to 6-character legal partition name. If you use the wildcard character (*) instead of a partition name, Indirect assumes you are testing the same partition in which the current version of Indirect is executing. Under VAX–11 RSX, the default partition name is GEN.

Example

```
.TESTPARTITION GEN
;GEN,400,541,SYS,
```

This directive acquires information about the partition named GEN. The partition has a starting address of 400, it is 541 bytes long, and it is in a system-controlled partition.

5.7 Examples

1 Invoking Indirect interactively and displaying symbols:

```
> @TI:
IND>
```

Specifying @TI: allows you to work with Indirect interactively. When Indirect issues the IND> prompt, you can enter directive statements, invoke command files, or display the values of special symbols. To display a symbol, use the .ENABLE SUBSTITUTION directive, then request the symbol in the following format:

```
IND>;'<symbol>'
```

2 Using an indirect command file:

A file named ASSEMBLY.CMD contains the following command lines:

```
.ENABLE SUBSTITUTION
;'<TIME>'
MAC @MACASM.CMD
.EXIT
```

To execute the command file, use the following command line:

```
> @ASSEMBLY
```

5.8 RSX–11 System Generation Requirements

VAX/VMS uses the MCR indirect command file capability to serve as the host system for RSX–11S, RSX–11M, or RSX–11M–PLUS system generation. Requirements for system generations are as follows:

- You must have the change-mode-to-kernel (CMKRNL) privilege. Change-mode-to-kernel privilege is required to perform a SET /UIC command that changes the default UIC.
- You must have logical I/O privilege (LOG_IO) for VMR to perform logical I/O on system files.
- You must have system protection privilege (SYSPRV) which allows you to create files in other directories.
- You must have volume protection privilege (VOLPRO) if you are using a magnetic tape kit so you can mount the tape with the /FOREIGN keyword for the MOUNT command.

5.8.1 RSX–11S Requirements

You can perform the RSX–11S system generation on the host system disk or on another disk. If you use the host system disk, you can select group numbers during the system generation procedure for the RSX–11S system and Executive files so that they will not conflict with files already on the disk. The valid group numbers you can select are between 3 and 10.

If you use a disk other than the system disk, the valid group numbers are 1 through 10 for system files and 1 through 11 for Executive files.

Note that if you must choose alternative group numbers for the system and Executive files, you must explicitly specify the UFD in the command line whenever you use such files as the system object library (SYSLIB.OLB) or the system macro library (RSXMAC.SML). Normally, TKB and MACRO–11 expect to find these libraries in UFD [1,1].

Multiple RSX–11S system generations may be performed if there is sufficient room on the specified disk.

You should complete the following steps prior to beginning an RSX–11S system generation (SYSGEN):

1 Mount your target disk for the RSX–11S SYSGEN.

2 Set your default device as the target disk (for example, SET DEFAULT DM0:).

3 Set your UIC to [2,200], as follows:

```
> SET /UIC=[2,200]
```

4 Allocate your RSX–11S kit device, labeled 11SKIT1.

5 Mount your RSX–11S kit device, labeled 11SKIT1.

6 Copy the command file 11SGEN.CMD from the [2,20] directory on the RSX–11S kit device to the [2,200] directory on the target disk. If the [2,200] directory does not exist on your target disk, you will have to create it.

7 Dismount the RSX–11S kit device, but do not unload it (for example, DM0 DM1:/NOUNLOAD).

8 Deallocate the RSX–11S kit device. (This allows the command procedure to allocate and mount the device.)

9 Invoke SYSGEN.

After you complete system generation, bring the system medium to a PDP–11 processor (for which the RSX–11S system was built) and boot it.

For more information on RSX–11S system generation, refer to the *RSX–11S System Generation and Installation Guide.*

5.8.2 RSX–11M Requirements

You can perform RSX–11M Version 4.1 Update D (or a later revision) and system generation (SYSGEN) on VAX–11 RSX Version 1.0. You should complete the following steps prior to beginning an RSX–11M SYSGEN:

1 Make a copy of the distribution kit.

2 Mount the baseline disk.

3 Set your UIC to [200,200], as follows:

```
> SET /UIC=[200,200]
```

4 Set your default disk to the baseline disk (for example, SET DEFAULT DM0:).

5 Apply the update.

6 Invoke SYSGEN.

For more information, refer to the *RSX-11M/11S Version 4.1 Update Installation Guide and Update Notes — Revision D* (or subsequent revisions) for details on applying the RSX-11M update on VAX-11 RSX Version 1.0. Refer to the *RSX-11M System Generation and Installation Guide* for details on performing SYSGEN.

5.8.3 RSX-11M-PLUS Requirements

You can perform RSX-11M-PLUS Version 2.1 Update D (or a later revision) and system generation (SYSGEN) on VAX-11 RSX Version 1.0. You should complete the following basic steps prior to beginning an RSX-11M-PLUS SYSGEN:

1 Disable line editing so that you can request help from SYSGEN by entering an escape character, as follows:

```
> SET TERMINAL/NO_LINE_EDITING
```

2 Make a copy of the distribution kit.

3 Mount the target system disk.

4 Set your UIC to [200,200], as follows:

```
> SET /UIC=[200,200]
```

5 Set your default disk to the target system disk (for example, SET DEFAULT DM0:).

6 Apply the update.

7 Invoke SYSGEN.

For more information, refer to the *RSX-11M-PLUS Version 2.1 Update Installation Guide and Update Notes — Revision D* (or subsequent revisions) for details on applying the RSX-11M-PLUS update on VAX-11 RSX Version 1.0. Refer to the *RSX-11M-PLUS System Generation and Installation Guide* for details on performing SYSGEN.

5.9 Error Message Processing

When Indirect encounters an error, it prints the appropriate error message and the command line in which the error occurred. If the line contained a substitution, the line as it appeared before the substitution took place is also displayed.

Section 9 explains the information-only messages and the error messages. The error messages are divided into classes, depending on the level of severity. Class 2 errors can be handled with the <ERRCTL> symbol (see 5.4.1.2) and class 1 errors can be handled with the .ONERR directive (see 5.6.21). Class 0 errors must be corrected outside of Indirect.

6 DCL Commands

This section describes the DCL commands and qualifiers used specifically by VAX–11 RSX. These commands allow VAX/VMS users to access various RSX–11 utilities and components (for example, MACRO–11). The syntax of the commands and the restrictions on their use are given in detail.

6.1 DCL Command Summary

Table 6–1 summarizes the DCL commands and their functions.

Table 6–1 DCL Command Summary

Command	Function
EDIT/SLP	Invokes the Source Language Input Program (SLP).
LIBRARY/RSX11	Creates or modifies an RSX–11 object module or an RSX–11 macro library, and it inserts, deletes, replaces, or lists modules, macros, or group global symbol names in a library.
LINK/RSX11	Invokes the RSX–11 Task Builder (TKB) to build an RSX–11 task image.
MACRO/RSX11	Invokes the PDP–11 MACRO–11 Assembler (MACRO–11) to assemble one or more MACRO–11 assembly language source programs.
MCR	Provides a means of running RSX–11 components in a manner compatible with the RSX–11 operating system.

The following sections describe the DCL commands and qualifiers.

6.2 DCL Command Descriptions

The descriptions of the commands given in this section observe the following conventions:

- Lowercase text indicates a variable whose actual value is determined when the command is entered. The value of the variable "task name" depends on the name of the task associated with the command.
- Square brackets ([]) indicate optional entries in a command line or file specification. Note that when you actually enter an option, you do not type the brackets.

  Square brackets are also part of the User File Directory (UFD) and User Identification Code (UIC) syntax. When you use a UFD or UIC (for example, in a file specification), brackets are required syntax elements; that is, they do not indicate optional syntax entries.

The following subheadings appear in the command descriptions. Subheadings that do not apply to a particular command are omitted from the command description.

Command Name

This subheading identifies the DCL command name in capital letters. The text following this subheading is an overview of the command's function.

Format

This subheading is followed by the command format and a list of command qualifiers (if any) and defaults as well as a list of file qualifiers (if any) and defaults.

Restrictions

This subheading is followed by a list of restrictions related to the use of the command. Read the restrictions carefully before you use the command.

Prompts

This subheading is followed by the prompt(s), if any, evoked by the command.

Command Parameters

This subheading is followed by an explanation of each command parameter.

Description

Under this subheading is information that supplements the overview appearing after the command name.

Command Qualifiers

This subheading is followed by an explanation of each command qualifier, if any.

File Qualifiers

This subheading is followed by an explanation of each file qualifier, if any.

Examples

This subheading is followed by one or more examples illustrating the use of the command.

EDIT/SLP

The EDIT/SLP command invokes the Source Language Input Program (SLP). The /SLP qualifier is required.

For more information on SLP, see the *RSX–11M/M–PLUS Utilities Manual.*

FORMAT

EDIT/SLP *file-spec*

Command Qualifiers	**Defaults**
/[NO]AUDIT_TRAIL[=(option[,...])]	*/AUDIT_TRAIL=(POSITION:80 - SIZE:8)*
/[NO]CHECKSUM[=value]	*/NOCHECKSUM*
/LIST[=file-spec]	
/[NO]OUTPUT[=file-spec]	*/OUTPUT*
/[NO]REPORT	*/NOREPORT*
/[NO]TAB	*/NOTAB*
/[NO]TRUNCATE[=position]	*/NOTRUNCATE*

restrictions

None.

prompt

_File: file-spec

command parameter

file-spec

Specifies the file to be edited. If you do not include a file type, it is null by default.

The file should be a disk file on a Files–11 formatted volume.

No wildcard characters are allowed in the file specification.

COMMAND QUALIFIERS

These qualifiers can be overridden in the SLP input file.

/AUDIT_TRAIL[=(option[,...])]
/NOAUDIT_TRAIL

Controls whether records in the output file from SLP contain an audit trail and optionally defines the location of the audit trail. You can specify one or both of the following options:

Option	Function
POSITION:n	Defines the starting character position of the audit trail. By default, the audit trail is placed in column 80. If you specify thi: option, SLP rounds the value n to the next highest tab stop.
SIZE:n	Defines the number of characters in the audit trail. By default, the audit trail is eight characters.
	If you specify more than one option, separate them by commas and enclose the list in parentheses.
	If you specify /NOAUDIT_TRAIL, the output file does not contain a record of the changes.

/CHECKSUM[=value]
/NOCHECKSUM

Controls whether a checksum is calculated for the edit commands. If you specify the /CHECKSUM qualifier without a value, SLP calculates and reports the checksum on your terminal. If you specify a value that differs from the one SLP calculates, SLP displays a warning message but completes the edit.

The default is /NOCHECKSUM, which does not calculate a checksum.

/LIST[=file-spec]

Creates a line-numbered listing of a file. By default, no line-numbered listing is produced. Use /LIST when you want a listing of lines in sequential order. If you do not specify a file specification with /LIST, SLP uses the same file name as the input file and a file type of LST. If you enter a file specification that does not include a file type, SLP uses the default file type of LST.

No wildcard characters are allowed in the file specification.

/OUTPUT=file-spec
/NOOUTPUT

Defines the file specification of the output file, if any, created during the editing session. If you do not specify /OUTPUT, the output file will have the same file name and type as the input file, and a version number one higher than the highest existing version of the file.

No wildcard characters are allowed in the file specification.

/REPORT
/NOREPORT

Controls whether line truncations that result from audit trails are reported. If you specify the /REPORT qualifier, not only will warning messages appear on the terminal, but the listing file will contain a question mark (?) in place of the period (.) in the line number of all truncated lines.

The default is /NOREPORT, which does not report line truncation.

/TAB
/NOTAB

Controls whether SLP places spaces or tabs at the end of each record containing an audit trail. The default is /NOTAB, which causes SLP to insert spaces at the end of each record that contains an audit trail.

If you specify the /TAB qualifier, SLP inserts tabs at the end of each record that contains an audit trail.

/TRUNCATE[=position]
/NOTRUNCATE

Requests SLP to truncate each record in the input file at a specified column when it creates the output file. This qualifier allows you to delete an audit trail from a file previously updated with SLP. The default is not to truncate the records.

If you do not specify a position with the /TRUNCATE qualifier, SLP truncates input records at the beginning position of the audit trail.

EXAMPLES

1

```
$ EDIT/SLP AVERAGE.FOR
     .
     .
     .
/
$
```

The command procedure illustrated uses the EDIT/SLP command. All input lines for the SLP editor follow the command in the input stream and are terminated by the slash character (/).

2

```
$ EDIT/SLP/LIST=AVLST AVERAGE.FOR
     .
     .
     .
/
SLP>
CTRL/Z Exit
$
```

This interactive editing session with the SLP editor requests a line-numbered listing, AVLST.LST, as output. The CTRL/Z terminates the session.

LIBRARY/RSX11

The LIBRARY/RSX11 command creates or modifies an RSX–11 object module library or an RSX–11 macro library, and it inserts, deletes, replaces, or lists modules, macros, or global symbol names in a library. The /RSX11 qualifier is required.

FORMAT

LIBRARY *library-file-spec [input-file-spec[,...]]*

Command Qualifiers	**Defaults**
/COMPRESS[=(option[,...])]	*/REPLACE*
/CREATE[=(option[,...])]	*/REPLACE*
/DELETE=(module[,...])	*/REPLACE*
/EXTRACT[=(module[,...])]	
/FULL	
/[NO]GLOBALS]	*/GLOBALS*
/INSERT	*/REPLACE*
/[NO]LIST[=file-spec]	*/NOLIST*
/MACRO	*/OBJECT*
/[NO]NAMES	*/NONAMES*
/OBJECT	*/OBJECT*
/OUTPUT=file-spec	
/REMOVE=(symbol[,...])	
/REPLACE	*/REPLACE*
/SELECTIVE_SEARCH	
/SQUEEZE	

restrictions

None.

prompts

_Library:	library-file-spec
_File:	input-file-spec[,...]

command parameters

library-file-spec

Specifies the name of the library to be created or modified.

No wildcard characters are allowed in the library file specification.

If the file specification does not include a file type, the LIBRARY/RSX11 command assumes a default file type of OLB if /OBJECT is specified either explicitly or by default, or a default file type of MLB if /MACRO is specified.

input-file-spec[,...]

Specifies the names of one or more files that contain modules to be inserted in the specified library. This parameter is required when you specify /INSERT or /REPLACE; it is optional when you specify /CREATE. Note that the default operation for the LIBRARY/RSX11 command is /REPLACE. If you do not specify a qualifier that requests a specific operation, you must enter the input-file-spec parameter.

If you do not specify any of the qualifiers /REPLACE, /INSERT, or /CREATE, the input-file-spec parameter is invalid.

If you specify more than one file, you can separate the file specifications with either plus signs (+) or commas (,). In either case, the contents of each file are inserted in the specified library.

No wildcard characters are allowed in the input file specifications.

If any file specification does not include a file type, the LIBRARY/RSX11 command assumes a default file type of OBJ when /OBJECT is specified either implicitly or by default, and a file type of MAC when /MACRO is specified.

DESCRIPTION

Libraries are files that contain one or more entries, or modules, of a similar type, and directories, or tables, that indicate the locations of individual modules within the library. There are two types of libraries:

- **Object module libraries** catalog frequently called routines. You can use object module libraries as input to the RSX–11 Task Builder. The RSX–11 Task Builder searches the object module library when it encounters a reference it cannot resolve from the input files specified.
- **Macro libraries** catalog macro definitions. You can use macro libraries as input to the assembler. The assembler searches the macro library when it encounters an .MCALL assembler directive.

Both object module libraries and macro libraries have a directory called a module name table (MNT) that lists the modules (or, in the case of macro libraries, the macros) in the library. An object module library also contains a global symbol table (GST) that lists the global symbols defined in each of the modules in the library.

When the LIBRARY/RSX11 command adds a file to an object module library, it catalogs the module according to its module name, not according to the file name of the file containing the module.

The LIBRARY/RSX11 command creates libraries and modifies their contents. You can use DCL commands to manipulate a library in its entirety. For example, the DELETE, COPY, and RENAME commands can delete, make copies of, and rename libraries, respectively.

For more information on RSX–11M and RSX–11M–PLUS libraries, see the *RSX–11M/M–PLUS Utilities Manual.*

When you use the LIBRARY/RSX11 command, you can specify qualifiers that request more than one function in a single command, with some restrictions. The qualifiers that perform LIBRARY functions, related qualifiers, and qualifier incompatibilities are summarized in Table 6–2.

Table 6–2 LIBRARY/RSX11 Command Qualifiers

Qualifier	Related Qualifier(s)	Incompatible Function(s)
/COMPRESS	/OUTPUT	/CREATE, /EXTRACT
/CREATE[1]	/SQUEEZE[2], /GLOBALS[3], SELECTIVE_SEARCH[3]	/COMPRESS, /EXTRACT
/DELETE	—	/EXTRACT
/EXTRACT	/OUTPUT	/COMPRESS, /CREATE, /DELETE, /INSERT, /LIST, /REMOVE, /REPLACE
/INSERT	/SQUEEZE[2], /GLOBALS[3], /SELECTIVE_SEARCH[3]	/EXTRACT
/LIST	/FULL, /NAMES[3]	/EXTRACT
/REMOVE[3]	—	/EXTRACT
/REPLACE	/SQUEEZE[2], /GLOBALS[3], /SELECTIVE_SEARCH[3]	/EXTRACT

[1] The /CREATE, /INSERT, and /REPLACE qualifiers are not incompatible. However, if more than one is specified, /CREATE takes precedence over /INSERT and /INSERT takes precedence over /REPLACE. The related qualifiers for /CREATE are applicable only if you enter one or more input files.
[2] Indicates a qualifier that applies only to macro libraries.
[3] Indicates a qualifier that applies only to object module libraries.

COMMAND QUALIFIERS

/COMPRESS[=(option[,...])]

Requests the LIBRARY/RSX11 command to recover unused space in the library resulting from module deletion. When you specify /COMPRESS, the LIBRARY/RSX11 command by default creates a new library with a version number one higher than the existing library. Use the /OUTPUT qualifier to specify a different name for the compressed library.

If you omit this qualifier, the default is /REPLACE.

You may specify one or more of the following options to increase or decrease the size of the library, overriding the values specified when the library was created:

Option	Function
BLOCKS:n	Specifies the number of 512 (10)-byte blocks to be allocated for the library.
GLOBALS:n	Specifies the maximum number of global symbols the library can contain (for object module libraries only).
MODULES:n	Specifies the maximum number of modules or macros the library can contain.

If you specify more than one option, separate them with commas or plus signs and enclose the list in parentheses.

/CREATE[=(option[,...])]

Requests the LIBRARY/RSX11 command to create a new library. When you specify /CREATE, you can optionally specify a file or a list of files that contain modules to be placed in the library.

If you omit this qualifier, the default is /REPLACE.

By default, the LIBRARY/RSX11 command creates an object module library. Specify /MACRO to indicate that the library is a macro library.

Specify one or more of the following options to control the size of the library, overriding the system defaults:

Option	Function
BLOCKS:n	Specifies the number of 512 (10)-byte blocks to be allocated for the library. By default, the LIBRARY/RSX11 command allocates 100 (10) blocks for a new library.
GLOBALS:n	Specifies the maximum number of global symbols the library can contain. By default, the LIBRARY/RSX11 command sets a maximum of 128 global symbols for an object module library. (A macro library does not have a global symbol directory. Therefore, the maximum for macro libraries defaults to 0.)
MODULES:n	Specifies the maximum number of modules the library can contain. By default, the LIBRARY/RSX11 command sets a maximum of 512 (10) modules for an object module library and 256 (10) modules for a macro library. If you specify the MODULES option, the maximum value allowed for n is 4096.

If you specify more than one option, separate them with commas and enclose the list in parentheses.

/DELETE=(module[,...])

Requests the LIBRARY/RSX11 command to delete one or more modules from a library. You must specify the names of one or more modules to be deleted from the library. If you specify more than one module, separate them with commas and enclose the list in parentheses. No wildcard characters are allowed in the module specification(s).

When the LIBRARY/RSX11 command deletes modules from a library, it issues the message:

```
MODULES DELETED:
```

Then it lists the names of modules it has successfully deleted.

The LIBRARY/RSX11 command does not physically remove a module from a library. Instead, it deletes the entry from the module name table. Use the /COMPRESS qualifier to compress a library from which modules have been deleted.

/EXTRACT[=(module[,...])]

Copies one or more modules from an existing library into a new file. If you specify more than one module, separate the module names with commas and enclose the list in parentheses. No wildcard characters are allowed in the module specification(s).

If you specify the /OUTPUT qualifier with /EXTRACT, the LIBRARY/RSX11 command writes the output into the file specified by the /OUTPUT qualifier. If you specify /EXTRACT and do not specify /OUTPUT, the LIBRARY/RSX11 command writes the file into a file that has the same file name as the library and a file type of OBJ or MAC, depending on the use of the /OBJECT and /MACRO qualifiers.

If you specify /EXTRACT, the module name is optional. All modules in the specified library are concatenated into a single file with a file type of OBJ or MAC, depending on the use of the /OBJECT and /MACRO qualifiers.

/FULL

Requests a full description of each module in the module name table. Use this qualifier with the /LIST qualifier to request that each module in the library be listed in the format:

entry SIZE:nnnnn INSERTED:dd-mmm-yy IDENT:nn

/GLOBALS
/NOGLOBALS

Controls, for object module libraries, whether the names of global symbols in modules being inserted in the library are included in the global symbol table.

By default, the LIBRARY/RSX11 command places all global symbol names in the global symbol table. Use /NOGLOBALS when you do not want the global symbol names in the global symbol table.

/INSERT

Requests the LIBRARY/RSX11 command to add the contents of one or more files to an existing library. If an object module file specified as input consists of concatenated object modules, the LIBRARY/RSX11 command creates a separate entry for each object module in the file; each entry reflects an individual module name. If a macro file specified as input contains more than one macro definition, the LIBRARY command creates a separate entry for each macro, naming each entry according to the names specified on the .MACRO directives.

When the LIBRARY/RSX11 command inserts modules into an existing library, it checks the module name table before inserting each module. If a module name, macro name, or global symbol name already exists in the library, the command issues an error message and does not add the module to the library. One or more modules, however, may be successfully inserted before the error occurs.

To insert or replace a module in a library regardless of whether there is a current entry with the same name, use the /REPLACE qualifier. The default is /REPLACE.

/LIST[=file-spec]
/NOLIST

Controls whether the LIBRARY/RSX11 command creates a listing of the contents of the library.

The default is /NOLIST. If you specify /LIST without including a file specification, the LIBRARY/RSX11 command writes the output file to the current SYS$OUTPUT device. If you include a file specification that does not have a file type, the LIBRARY/RSX11 command uses the default file type of LST.

If you specify /LIST with qualifiers that perform additional operations on the library, the LIBRARY/RSX11 command creates the listing after completing all the other requests. Thus, the listing reflects the status of the library after all changes have been made.

No wildcard characters are allowed in the file specification.

When you specify /LIST, the LIBRARY/RSX11 command provides, by default, the following information about the library:

DIRECTORY OF FILE file-spec

File name, file type, and version number of the library being listed.

library-type LIBRARY CREATED BY: LBR vvvvvv

The type of library (OBJECT or MACRO) and the version number of the librarian that created the library.

LAST INSERT OCCURRED dd-mmm-yy AT hh:mm:ss

The date and time at which the last insertion was made.

MNT ENTRIES ALLOCATED: nn; AVAILABLE: mm

The current status of the module name table: the maximum number of entries that can be entered in the table (nn) and the number of entries that are unused (mm).

EPT ENTRIES ALLOCATED: nn; AVAILABLE: mm

The current status of the entry point table: the maximum number of entries that can be entered in the table (nn) and the number of entries that are unused (mm). For a macro library, both values are always 0.

FILE SPACE AVAILABLE: nnnnn WORDS

The amount of space available in the library for new files.

RECOVERABLE DELETED SPACE: nnnnn WORDS

The amount of space occupied by modules whose entries have been deleted from the module name table. To recover this space, use the /COMPRESS qualifier.

```
module
module
    .
    .
    .
```

The names of all the entries in the module name table.

If you specify /LIST, you can also specify /FULL and /NAMES to request additional information in the listing.

/MACRO

Indicates that the library is a macro library containing macros for the MACRO–11 Assembler.

The input file type defaults to MAC.

/NAMES
/NONAMES

Controls, when /LIST is specified for an object module library, whether the LIBRARY/RSX11 command lists the names of all global symbols in the global symbol table as well as the module names in the module name table.

The default is /NONAMES. If you specify /NAMES, each entry name is displayed in the following format:

```
** MODULE:entry-name
  symbol   symbol   symbol   symbol   symbol   symbol
    .        .        .        .        .        .
    .        .        .        .        .        .
    .        .        .        .        .        .
```

If the library is a macro library and you specify /NAMES, no symbol names are displayed.

/OBJECT

Indicates that the library is an object module library in RSX–11 format. This is the default.

/OUTPUT=file-spec

Specifies, when the /EXTRACT or /COMPRESS qualifier is also specified, the file specification of the output file. No wildcard characters are allowed in the file specification.

For /EXTRACT, the output file contains the modules extracted from a library. For /COMPRESS, the output file contains the compressed library.

/REMOVE=(symbol[,...])

Requests the LIBRARY/RSX11 command to delete entries for one or more global symbols from the global symbol table.

When you specify /REMOVE, the LIBRARY/RSX11 command displays the message:

```
GLOBAL SYMBOLS DELETED:
```

Then it displays the names of the global symbols it successfully deleted.

/REPLACE

Requests the LIBRARY/RSX11 command to replace one or more existing modules in a library with the module(s) in the specified input file(s). The LIBRARY/RSX11 command first deletes an existing entry, if any, for each module or macro in the input file(s) and the corresponding global symbols, then inserts the new module or macro in the library.

This is the default operation. If you specify an input file parameter and do not specify /CREATE, /INSERT, or /REPLACE, the LIBRARY/RSX11·command replaces an existing module or modules in the file with the module(s) in the files specified.

When you use the /REPLACE function, the LIBRARY/RSX11 command displays the name of each module replaced in the format:

```
MODULE "module-name" REPLACED
```

/SELECTIVE_SEARCH

Defines the input files being inserted into a library as candidates for selective searches by the Linker. If you specify /SELECTIVE_SEARCH, the modules are selectively searched by the Linker when the library is specified as a Linker input file. The Linker includes only global symbol(s) in the module(s) that are referenced by other modules in the symbol table of the output image file.

/SQUEEZE

Requests that the LIBRARY/RSX11 command compress individual macros before adding them to a macro library. When you specify /SQUEEZE, trailing blanks, trailing tabs, and comments are deleted from each macro before insertion in the library.

Use /SQUEEZE with the /CREATE, /INSERT, and /REPLACE qualifiers to conserve space in a macro library. By default, macros are compressed.

EXAMPLES

1

```
$ LIBRARY/RSX11/CREATE TESTLIB ERRMSG,STARTUP
```

The LIBRARY/RSX11 command creates an object module library named TESTLIB.OLB and places the modules ERRMSG.OBJ and STARTUP.OBJ in the library.

2

```
$ LIBRARY/RSX11/INSERT TESTLIB SCANLINE
$ LINK/RSX11 TERMTEST,TESTLIB/LIBRARY
```

The LIBRARY/RSX11 command adds the module SCANLINE.OBJ to the library TESTLIB.OLB. The library is specified as input to the Task Builder by using the /LIBRARY qualifier on the LINK/RSX11 command. If the module TERMTEST.OBJ refers to any routines or global symbols not defined in TERMTEST, the RSX-11 Task Builder will search the global symbol table of library TESTLIB.OLB to resolve the symbols.

3

```
$ LIBRARY/RSX11/DELETE=STARTUP TESTLIB
MODULES DELETED:
STARTUP
$ LIBRARY/RSX11/LIST/NAMES TESTLIB
DIRECTORY OF FILE TESTLIB.OLB;2
OBJECT MODULE LIBRARY CREATED BY LBR:  VX4.0
LAST INSERT OCCURRED 10-DEC-84 AT 16:41:23
MNT ENTRIES ALLOCATED:  75 AVAILABLE:  73
EPT ENTRIES ALLOCATED:  275;  AVAILABLE:  270
FILE SPACE AVAILABLE:  18347 WORDS
RECOVERABLE DELETED SPACE:  00134 WORDS
** MODULE:ERRMSG
  ERRMSG    ERR$ERROR    ERR$FATAL    ERR$WARNING
** MODULE:SCANLINE
  SCANLINE
$ LIBRARY/RSX11/COMPRESS=(BLOCKS:50)  TESTLIB
$ PURGE TESTLIB.OLB
```

The LIBRARY/RSX11 command deletes the module STARTUP from the library TESTLIB.OLB. The next LIBRARY/RSX11 command requests a listing of the contents of the library TESTLIB.OLB. The /NAMES qualifier requests a list of each of the global symbols in the modules.

The listing indicates that the deletion of the module STARTUP resulted in unused space. The /COMPRESS function deletes the space and requests that 50 blocks be allocated for the library. By default, /COMPRESS creates a new version of the library. The PURGE command purges the earlier version.

4

```
$ LIBRARY/RSX11/EXTRACT=(DESCRIPTOR,RDTERM,WRTERM)-
$_/OUTPUT=TEMP-
$_DBB2:[GOODWIN.LIB]LOCALMAC.MLB
```

The /EXTRACT qualifier names three macros, DESCRIPTOR, RDTERM, and WRTERM, to be copied from the macro library LOCALMAC.MLB in the subdirectory [GOODWIN.LIB] on the disk DBB2. The /OUTPUT qualifier requests that the output file have a file name of TEMP. The LIBRARY/RSX11 command writes the output to the file TEMP.MAC in your current default disk and directory.

5

```
$ LIBRARY/RSX11/MACRO/CREATE=(BLOCKS:40,MODULES:100)-
$_MYMAC TEMP
$ MACRO MYMAC/LIBRARY+CYGNUS/OBJECT
```

The LIBRARY/RSX11 command creates a macro library named MYMAC.MLB from the macros in the file TEMP.MAC. The new library has room for 100 modules in a 40-block file. If the input file contains multiple macros, each macro is entered in the new library.

The MACRO command assembles the source file CYGNUS.MAC. The /LIBRARY qualifier specifies the library MYMAC.MLB as an input file. If the source file CYGNUS contains any .MCALL assembly directives referencing macros not defined within the file, the assembler searches the library.

6

```
$ LIBRARY/RSX11/LIST=MYMAC.LIS/FULL MYMAC.MLB
```

The LIBRARY/RSX11 command requests a full listing of the macro library MYMAC.MLB. The output is written to a file named MYMAC.LST.

LINK/RSX11

The LINK/RSX11 command invokes the RSX–11 Task Builder (TKB) to build an RSX–11 task image. The /RSX11 qualifier is required.

For more information on TKB, see the *RSX–11M/M–PLUS and Micro/RSX Task Builder Manual.*

FORMAT

LINK/RSX11 *file-spec[,...]*

Command Qualifiers	**Defaults**
/BRIEF	*/BRIEF*
/[NO]CROSS_REFERENCE	*/NOCROSS_REFERENCE*
/[NO]DEBUG[=file-spec]	*/NODEBUG*
/DEFAULT_LIBRARY=file-spec	
/[NO]EXECUTABLE[=file-spec]	*/EXECUTABLE*
/[NO]EXIT[=n]	*/NOEXIT*
/FULL	*/BRIEF*
/[NO]HEADER	*/HEADER*
/[NO]MAP[=file-spec]	*/NOMAP*
/[NO]OVERLAY_DESCRIPTION	*/NOOVERLAY_DESCRIPTION*
/[NO]POSITION_INDEPENDENT	*/NOPOSITION_INDEPENDENT*
/[NO]POST_MORTEM	*/NOPOST_MORTEM*
/[NO]SEQUENTIAL	*/NOSEQUENTIAL*
/[NO]SYMBOL_TABLE	*/NOSYMBOL_TABLE*
/TKB_OPTIONS=file-spec	
/[NO]TRACE	*/NOTRACE*

File Qualifiers	**Defaults**
/[NO]CONCATENATED	*/CONCATENATED*
/INCLUDE=(module[,...])	
/LIBRARY	
/SELECTIVE_SEARCH	

restrictions

None.

prompt

_File: file-spec[,...]

command parameter

file-spec[,...]

File specifications of one or more input files. The input files may be object modules to be linked or libraries to be searched for external references. If multiple input files are specified, they may be separated either with commas (,) or plus signs (+). In either case, a single RSX–11 task image file is produced.

If a file specification does not contain a file type, the Task Builder supplies default file types, based on the nature of the file. All object modules are assumed to have a file type of OBJ; libraries are assumed to have a file type of OLB; overlay descriptor files are assumed to have a file type of ODL.

No wildcard characters are allowed in the file specification.

COMMAND QUALIFIERS

/BRIEF

Requests the Task Builder to produce a brief map (memory allocation) file. The /BRIEF qualifier is valid only if /MAP is also specified. If /MAP is specified, /BRIEF is the default. A brief map contains the following information:

- A summary of the task image attributes
- A list of all segments in the task image
- A summary of Task Builder statistics

/CROSS_REFERENCE
/NOCROSS_REFERENCE

Controls whether the memory allocation listing (map) contains a symbol cross-reference. /CROSS_REFERENCE is valid only if /MAP is also specified.

A symbol cross-reference lists each global symbol referenced in the task image, its value, and all modules in the task image that refer to it.

/DEBUG[=file-spec]
/NODEBUG

Controls whether a task image is bound with a debugger. The /DEBUG qualifier optionally accepts the name of an alternate, user-specified debugging aid. If a file specification is entered but it does not contain a file type, the Task Builder assumes the default file type of OBJ.

If you specify /DEBUG without a file specification, the default debugger, the On-Line Debugging Tool (ODT), is used.

/DEFAULT_LIBRARY=file-spec

Defines an object module library to use in place of the default system library SYSLIB.OLB. The specified library is searched only after all libraries specified as input files when unresolved references are encountered.

No wildcard characters are allowed in the file specification.

/EXECUTABLE[=file-spec]
/NOEXECUTABLE

Controls whether the Task Builder produces an executable task image and optionally provides a file specification for the output file.

By default, the Task Builder creates a task image with the same file name as the first input file and a file type of EXE. If the first input file you specify is the name of a library qualified with /INCLUDE, then the default file name for the object module created is the same as the name of the first or only module specified with /INCLUDE.

Use /NOEXECUTABLE when you want to determine the outcome of task building a set of modules without incurring the Task Builder overhead required to create a task image.

When you specify /EXECUTABLE, you can control the defaults applied to the output file specification by placing the qualifier in the command. The output file type defaults to EXE.

/EXIT[=n]
/NOEXIT

Controls whether the Task Builder exits after a specified number of error diagnostics. By default, the Task Builder does not exit as a result of diagnostic errors. If you specify /EXIT, the Task Builder exits after n diagnostic errors (n is assumed to be a decimal number by default). If you specify /EXIT and do not specify a value for n, it defaults to a value of 1.

/FULL

Requests the Task Builder to produce a full map (memory allocation) listing. The /FULL qualifier is valid only if /MAP is specified. A full map contains the following information:

- All the information included in the brief map
- A file contents section for each module in the task image
- A list of global symbol definitions by module
- A list of unresolved global symbol references

/HEADER
/NOHEADER

Controls whether the Task Builder includes a task header in the task image and in the symbol table file.

/MAP[=file-spec]
/NOMAP

Controls whether a memory allocation listing (map) is produced and optionally defines the file specification. If /MAP is specified, the qualifiers /BRIEF or /FULL can also be specified to control the contents of the map. If neither of these qualifiers is specified, /BRIEF is the default.

When you specify /MAP, you can control the defaults applied to the output file specification by the placement of the qualifier in the command. The output file type defaults to MAP.

No wildcard characters are allowed in the file specification.

/OVERLAY_DESCRIPTION
/NOOVERLAY_DESCRIPTION

Indicates whether the input file describes an overlay structure for the task image. If the input file specification does not contain a file type, the Task Builder uses the default file type of ODL. If you specify /OVERLAY_DESCRIPTION, you can specify only a single input file, which must contain the input file specifications and an overlay description.

/POSITION_INDEPENDENT /NOPOSITION_INDEPENDENT

Indicates whether the task image being built contains position-independent code. By default, the Task Builder assumes that code is not position-independent.

/POST_MORTEM /NOPOST_MORTEM

Controls whether the Task Builder sets the Post Mortem Dump flag. If you specify /POST_MORTEM, the system automatically lists the contents of memory when the task image terminates abnormally.

/SEQUENTIAL /NOSEQUENTIAL

Controls whether the Task Builder reorders program sections alphabetically when it creates the task image. If you specify /SEQUENTIAL, the Task Builder orders program sections in the order in which they are input.

/SYMBOL_TABLE[=file-spec] /NOSYMBOL_TABLE

Requests the Task Builder to create a separate file in object module format containing symbol definitions for all symbols contained in the task image.

If /DEBUG is specified, the Task Builder includes the symbol definitions in the task image for use by the debugger and also creates a separate symbol table file.

The symbol table file can be used as input to subsequent LINK/RSX11 commands to provide the symbol definitions to other task images.

By default, the Task Builder does not create a symbol table file. When you specify /SYMBOL_TABLE, you can control the defaults applied to the output file specification by placing the qualifier in the command. The output file type defaults to STB.

No wildcard characters are allowed in the file specification.

/TKB_OPTIONS=file-spec

Specifies the name of a file containing Task Builder options. If the file specification does not include a file type, the default file type of CMD is assumed. You must omit the initial slash character (/) in an options file specified as input to the LINK/RSX11 command.

No wildcard characters are allowed in the file specification.

/TRACE /NOTRACE

Indicates whether the task image is traceable. If you specify /TRACE, a trace trap occurs following the execution of each instruction when the task image is executed.

FILE QUALIFIERS

/CONCATENATED /NOCONCATENATED

Indicates whether the associated input file consists of concatenated object modules. By default, the Task Builder includes all the modules in the file in the task image. If you specify /NOCONCATENATED, the Task Builder includes only the first module in the file.

/INCLUDE=(module-name[,...])

Indicates that the associated input file is an object module library and that only the module names specified are to be unconditionally included as input to the Task Builder.

At least one module name must be specified. If you specify more than one module name, separate them with commas and enclose the list in parentheses.

If you specify /INCLUDE, you cannot specify /LIBRARY. If you want the library also to be searched for unresolved references, you must specify the library file specification a second time.

/LIBRARY

Indicates that the associated input file is an RSX–11 object module library to be searched for modules resolving any undefined symbols in the input files.

If the associated input file specification does not include a file type, the Task Builder assumes the default file type of OLB.

You cannot specify a library as the first input file.

/SELECTIVE_SEARCH

Indicates that the associated input file is an object module, and that any symbols unnecessary for resolving outstanding references should be excluded from the symbol table of the output task image file as well as from the symbol table file if /SYMBOL_TABLE is specified. The binary code in the object module is always included.

EXAMPLE

```
$ LINK/RSX11 AVERAGE
$ RUN AVERAGE
```

The object module AVERAGE.OBJ is linked to create the task image named AVERAGE.EXE. The RUN command executes the task.

MACRO/RSX11

The MACRO/RSX11 command invokes the PDP–11 MACRO–11 Assembler to assemble one or more MACRO–11 assembly language source programs. The /RSX11 qualifier is required.

For more information on the PDP–11 MACRO–11 Assembler, see the *PDP–11 MACRO–11 Language Reference Manual.*

FORMAT

MACRO/RSX11 *file-spec[,...]*

Command Qualifiers	**Defaults**
None	*None*

File Qualifiers	**Defaults**
/[NO]CROSS_REFERENCE	*/NOCROSS_REFERENCE*
/DISABLE=(function[,...])	*/DISABLE=(ABSOLUTE,DEBUG, TRUNCATION,SUPPRESSION*
/LIBRARY	
/[NO]LIST[=file-spec]	*See text*
/[NO]OBJECT[=file-spec]	*See text*
/[NO]SHOW[=(function[,...])]	*/SHOW=(CONDITIONALS,CALLS, DEFINITIONS*

restrictions

None.

prompt

_File: file-spec[,...]

command parameter

file-spec[,...]

Specifies one or more MACRO–11 assembly language source files to be assembled. If you do not specify a file type for an input file, the assembler uses the default file type of MAC.

You can specify more than one input file. If you separate the file specifications with commas (,), each file is assembled separately. If you separate the file specifications with plus signs (+), the files are concatenated and assembled as a single input file, producing single object and listing files.

FILE QUALIFIERS

/CROSS_REFERENCE
/[NO]CROSS_REFERENCE

Controls whether a cross-reference listing is included in the listing file. The /CROSS_REFERENCE qualifier requires that a listing file exist. The /NOCROSS_REFERENCE qualifier excludes it.

/DISABLE=(function[,...])

Provides initial settings for the functions controlled by the assembler directives .ENABLE and .DISABLE. You must specify at least one of the functions listed below. You can disable the following:

Function	Description
ABSOLUTE	Assembly of relative addresses as absolute addresses
DEBUG	Inclusion of local symbol table information in the object file for use with the debugger
TRUNCATION	Truncation of floating-point numbers (if truncation is disabled, numbers are rounded)
GLOBAL	Assumption that undefined symbols in the assembly are external symbols
SUPPRESSION	Suppression of listing of unreferenced symbols in the symbol table
TRACEBACK	Providing information to the debugger traceback mechanism

If you specify more than one function, separate them by commas and enclose the list in parentheses.

/LIBRARY

Indicates that the associated input file is a macro library. If you do not specify a file type, the assembler uses the default file type of MLB.

If you specify more than one macro library as input files, the libraries are searched in reverse order of their specification when a macro call is issued in a source program.

/LIST[=file-spec]
/NOLIST

Controls whether an output listing is created and optionally provides an output file specification for the listing file.

If you issue the MACRO command interactively, the assembler, by default, does not create a listing file. When /NOLIST is present, either explicitly or by default, errors are reported on the current output device.

If you execute the MACRO command in a batch job, /LIST is the default. When you specify /LIST, you can control the defaults applied to the output file specification by placing the qualifier in the command. The default file type provided for listing files is LST.

No wildcard characters are allowed in the file specification.

/OBJECT[=file-spec]
/NOOBJECT

Controls whether an object module is created by the assembler. It also defines the file specification for the file.

By default, the assembler creates an object module with the same file name as the first input file and a file type of OBJ. When you specify /OBJECT, you can control the defaults applied to the output file specification by placing the qualifier in the command. The default file type provided for object files is OBJ.

No wildcard characters are allowed in the file specification.

/SHOW[=(function[,...])]
/NOSHOW[=(function[,...])]

Provides initial settings for the functions controlled by the assembler directives .SHOW and .NOSHOW. You can specify one or more of the functions listed. If you specify /SHOW without any functions, the listing level count is incremented. If you specify /NOSHOW without any functions, the listing level count is decremented.

The /SHOW qualifier requires that a listing file exist. The functions you may specify are as follows:

Function	Description
CONDITIONALS	Lists unsatisfied conditional code associated with .IF and .ENDC directives
CALLS	Lists macro calls and repeat range expansions
DEFINITIONS	Lists macro definitions
EXPANSIONS	Lists macro expansions
BINARY	Lists binary code generated by the expansion of macro calls

If you omit the /SHOW qualifier, the default is equivalent to /SHOW=(CONDITIONALS,CALLS,DEFINITIONS).

If you specify more than one function, separate them by commas and enclose the list in parentheses.

EXAMPLES

1

```
$ MACRO/RSX11  ORION
```

The MACRO–11 Assembler assembles the file ORION.MAC and creates an object file named ORION.OBJ. If this command is executed in a batch job, the assembler also creates a listing file named ORION.LST.

2

```
$ MACRO/RSX11/LIST  CYGNUS, LYRA/OBJECT=LYRAN
```

This MACRO/RSX11 command requests two separate assemblies. First, the MACRO–11 Assembler assembles CYGNUS.MAC to produce CYGNUS.LST and CYGNUS.OBJ. Then it assembles LYRA.MAC and creates a listing file named LYRA.LST and an object module named LYRAN.OBJ.

MCR

The MCR command provides a means of running RSX–11 components in a manner compatible with the RSX–11 operating system.

See Section 1 for information on using RSX–11 components under VAX–11 RSX and for a list of the RSX–11 components available.

FORMAT

MCR *[component [command-line]]*

Command Qualifiers	**Defaults**
None	*None*

restrictions

This command does **not** invoke the MCR command interpreter. It can only invoke the specified component.

prompt

MCR>

command parameters

component

The RSX-11 command used to invoke the desired component, for example, MAC or TKB. If you do not specify a component, you will be prompted for one.

The component name must be equivalent to the file name portion of the file specification for the component as it exists on the VAX/VMS system disk, that is, SYS$SYSTEM:filename. For example, you must type MAC, not MACRO, to invoke the MACRO-11 Assembler.

command-line

A valid command line for the component. If you do not specify a command line, the requested component prompts for a command string.

EXAMPLES

```
1   $ MCR
    MCR>PIP MYFILE.DAT/SP
    MCR>CTRL/Z Exit
    $
```

The MCR command prompts for a component and a command line for that component. After the PIP command executes, the MCR command continues prompting until you press CTRL/Z to return to DCL.

2

```
$ MCR PIP
PIP>MYFILE.DAT/SP
PIP>CTRL/Y Interrupt
$
```

The MCR command requests PIP, which in turn prompts for a command line. Pressing CTRL/Y returns control to DCL.

PART III Directives and I/O

7 System Directives

This section describes how VAX-11 RSX handles RSX-11 directives. Table 7-1 summarizes differences between the VAX-11 RSX and the RSX-11 handling of directives. All directives supported under VAX-11 RSX are described in 7.2. These directives are alphabetized by macro name. VAX-11 RSX handling that is different from RSX-11 handling is shaded in gray.

7.1 System Directive Summary

Table 7-1 lists the directives and briefly notes the differences between RSX-11 and VAX-11 RSX handling of directives. For each directive, the table gives the macro name and the principal differences in handling for VAX-11 RSX.

Table 7-1 VAX-11 RSX Handling of Directives

RSX-11 Macro	RSX-11 Directive Name and Differences in VAX-11 RSX Handling
ABRT$	Abort Task
	Process protected by group; group or world privilege required; process name required for target task image
ALTP$	Alter Priority
	No operation for VAX-11 RSX
ALUN$	Assign LUN
	Logical name translated; device assignment required; devices mapped
ASTX$S	AST Service Exit
	No differences for VAX-11 RSX
ATRG$	Attach Region
	Not supported by VAX-11 RSX
CINT$	Connect To Interrupt Vector
	Not supported by VAX-11 RSX
CLEF$	Clear Event Flag
	No differences for VAX-11 RSX
CMKT$	Cancel Mark Time Requests
	Process protected by group; GROUP or WORLD privilege required
CNCT$	Connect
	Not supported by VAX-11 RSX

Table 7–1 (Cont.) VAX–11 RSX Handling of Directives

RSX–11 Macro	RSX–11 Directive Name and Differences in VAX–11 RSX Handling
CPCR$	Checkpoint Common Region
	Not supported by VAX–11 RSX
CRAW$	Create Address Window
	Not supported by VAX–11 RSX
CRGF$	Create Group Global Event Flags
	Group global event flags protected by group
CRRG$	Create Region
	Not supported by VAX–11 RSX
CRVT$	Create Virtual Terminal
	Not supported by VAX–11 RSX
CSRQ$	Cancel Time Based Initiation Requests
	Process protected by group; GROUP or WORLD privilege required; process name required for target task image
DECL$S	Declare Significant Event
	Supported, but no operation performed by VAX–11 RSX
DSAR$S	Disable AST Recognition
	No differences for VAX–11 RSX
DSCP$S	Disable Checkpointing
	Set swap mode privilege (PSWAPM) required
DTRG$	Detach Region
	Not supported by VAX–11 RSX
ELAW$	Eliminate Address Window
	Not supported by VAX–11 RSX
ELGF$	Eliminate Group Global Event Flags
	Group global event flags protected by group
ELVT$	Eliminate Virtual Terminal
	Not supported by VAX–11 RSX
EMST$	Emit Status
	Not supported by VAX–11 RSX
ENAR$S	Enable AST Recognition
	No differences for VAX–11 RSX
ENCP$S	Enable Checkpointing
	Set swap mode privilege (PSWAPM) required
EXIF$	Exit If
	Task image termination
EXIT$S	Task Exit
	Task image termination

Table 7–1 (Cont.) VAX–11 RSX Handling of Directives

RSX–11 Macro	RSX–11 Directive Name and Differences in VAX–11 RSX Handling
EXST$	Exit With Status
	Task image termination
EXTK$	Extend Task
	No differences for VAX–11 RSX
FEAT$	Test For System Feature
	Limited support for Indirect Command Processor only
GCCI$	Get Command For Command Interpreter
	Not supported by VAX–11 RSX
GCII$	Get Command Interpreter Information
	Limited support for Indirect Command Processor only
GDIR$	Get Default Directory
	Limited support for Indirect Command Processor only
GDVI$	Get Device Information
	Not supported by RSX–11M Version 4.1 or RSX–11M–PLUS Version 2.1; extension of GLUN$ directive
GLUN$	Get LUN Information
	Words 3 and 4 have VAX/VMS device-dependent longword
GMCR$	Get MCR Command Line
	No differences under the MCR CLI; see notes for this directive
GMCX$	Get Mapping Context
	Not supported by VAX–11 RSX
GPRT$	Get Partition Parameters
	Parameters given for GEN partition
GREG$	Get Region Parameters
	Not supported by VAX–11 RSX
GSSW$S	Get Sense Switches
	Not supported by VAX–11 RSX
GTIM$	Get Time Parameters
	100-tick-per-second clock used
GTSK$	Get Task Parameters
	Parameters given for GEN partition
IHAR$S	Inhibit AST Recognition
	No differences for VAX–11 RSX
MAP$	Map Address Window
	Not supported by VAX–11 RSX
MRKT$	Mark Time
	100-tick-per-second clock used

Table 7–1 (Cont.) VAX–11 RSX Handling of Directives

RSX–11 Macro	RSX–11 Directive Name and Differences in VAX–11 RSX Handling
MSDS$	Map Supervisor D-Space Not supported by VAX–11 RSX
MVTS$	Move To/From User/Supervisor I/O Space Not supported by VAX–11 RSX
QIO$	Queue I/O Request Function codes mapped to VAX/VMS function codes
QIOW$	Queue I/O Request And Wait Function codes mapped to VAX/VMS function codes
RCST$	Receive Data Or Stop Mailbox used; protected by group; process name required for target task image; mailbox logical name is accessible only to processes in same group
RCVD$	Receive Data Mailbox used; protected by group; process name required for target task image; mailbox logical name is accessible only to processes in same group
RCVX$	Receive Data Or Exit Mailbox used; protected by group; process name required for target task image; mailbox logical name is accessible only to processes in same group
RDAF$	Read All Event Flags No differences for VAX–11 RSX
RDEF$	Read Event Flag Not supported by VAX–11 RSX
RDXF$	Read Extended Event Flags No differences for VAX–11 RSX
RMAF$	Remove Affinity Not supported by VAX–11 RSX
RPOI$	Request And Pass Offspring Information Not supported by VAX–11 RSX
RQST$	Request Task Active or hibernating target task image required; protected by group; privilege required; process name required for target task image
RREF$	Receive By Reference Not supported by VAX–11 RSX
RSUM$	Resume Task Protected by group; privilege required; process name required for target task image

Table 7–1 (Cont.) VAX–11 RSX Handling of Directives

RSX–11 Macro	RSX–11 Directive Name and Differences in VAX–11 RSX Handling
RUN$	Run Task
	Active or hibernating target task image required; protected by group; privilege required; process name required for target task image
SCAA$	Specify Command Arrival AST
	Not supported by VAX–11 RSX
SCAL$	Supervisor Call
	Not supported by VAX–11 RSX
SCLI$	Set Command Line Interpreter
	Not supported by VAX–11 RSX
SDAT$	Send Data
	Mailbox used; protected by group; privilege required; process name required for target task image
SDIR$	Set Default Directory
	Limited support for Indirect Command Processor only
SDRC$	Send, Request, And Connect
	Not supported by VAX–11 RSX
SDRP$	Send, Request, And Pass Offspring Information
	Not supported by VAX–11 RSX
SETF$	Set Event Flag
	No differences for VAX–11 RSX
SFPA$	Specify Floating Point Processor Exception AST
	Cannot be disabled by LDFPS instruction
SMSG$	Send Message
	Not supported by VAX–11 RSX
SNXC$	Send Next Command
	Not supported by VAX–11 RSX
SPEA$	Specify Parity Error AST
	Not supported by VAX–11 RSX
SPND$S	Suspend
	Process name required for caller
SPRA$	Specify Power Recovery AST
	No differences for VAX–11 RSX
SPWN$	Spawn
	For command line handling, mailbox used
SRDA$	Specify Receive Data AST
	No differences for VAX–11 RSX

Table 7–1 (Cont.) VAX–11 RSX Handling of Directives

RSX–11 Macro	RSX–11 Directive Name and Differences in VAX–11 RSX Handling
SREA$	Specify Requested AST
	Not supported by VAX–11 RSX
SREF$	Send By Reference
	Not supported by VAX–11 RSX
SREX$	Specify Requested Exit AST (Extended)
	Not supported by VAX–11 RSX
SRRA$	Specify Receive-By-Reference AST
	Not supported by VAX–11 RSX
STAF$	Set Affinity
	Not supported by VAX–11 RSX
STIM$	Set System Time
	Not supported by VAX–11 RSX
STLO$	Stop For Logical OR Of Event Flags
	No differences for VAX–11 RSX
STOP$S	Stop
	No differences for VAX–11 RSX
STSE$	Stop For Single Event Flag
	No differences for VAX–11 RSX
SVDB$	Specify SST Vector Table For Debugging Aid
	No differences for VAX–11 RSX
SVTK$	Specify SST Vector Table For Task
	No differences for VAX–11 RSX
ULGF$	Unlock Group Global Event Flags
	Not supported by VAX–11 RSX
UMAP$	Unmap Address Window
	Not supported by VAX–11 RSX
USTP$	Unstop Task
	Active or hibernating target task image required; protected by group; privilege required; process name required for target task image
VRCD$	Variable Receive Data
	Not supported by VAX–11 RSX
VRCS$	Variable Receive Data Or Stop
	Not supported by VAX–11 RSX
VRCX$	Variable Receive Data Or Exit
	Not supported by VAX–11 RSX

Table 7–1 (Cont.) VAX–11 RSX Handling of Directives

RSX–11 Macro	RSX–11 Directive Name and Differences in VAX–11 RSX Handling
VSDA$	Variable Send Data
	Not supported by VAX–11 RSX
VSRC$	Variable Send, Request And Connect
	Not supported by VAX–11 RSX
WSIG$S	Wait For Significant Event
	No operation performed by VAX–11 RSX
WTLO$	Wait For Logical OR Of Event Flags
	No differences for VAX–11 RSX
WTSE$	Wait For Single Event Flag
	No differences for VAX–11 RSX

7.2 System Directive Descriptions

Each directive description includes all or most of the following elements, as appropriate:

Name

This describes the function of the directive for VAX–11 RSX.

Macro Call

This shows the macro call, defines each parameter, and gives the defaults for optional parameters in parentheses following the definition of the parameter. Since zero is supplied for most defaulted parameters, only nonzero default values are shown. Parameters ignored by VAX–11 RSX and RSX–11 are required for compatibility with RSX–11D and IAS.

DSW Return Codes

This section lists and defines all Directive Status Word (DSW) return codes that are valid under VAX–11 RSX. In some cases, a VAX/VMS return status code in parentheses follows an RSX–11 status code. For example:

IE.RSU—Device allocated to another image (SS$_DEVALLOC)

The VAX/VMS code indicates the VAX/VMS error that caused the corresponding RSX–11 code to be returned.

Some RSX–11 codes reflect several VAX/VMS codes. In this case, VAX–11 RSX returns the RSX–11 code that it uses by default. Such codes are followed by the phrase "default error" . For example:

IE.IDU—Device or unit unknown (default error)

In some cases after a directive failure, VAX–11 RSX returns an error code that is more meaningful for an I/O operation. In these cases, the high-order byte of the DSW contains 0.

Note

The notes presented with some directive descriptions further explain the function, use, and/or consequences of using these directives under VAX–11 RSX. Read the notes carefully to ensure proper use of the directives.

ABRT$—Abort Task

The Abort Task directive instructs the system to terminate the execution of the indicated process's task image. The requester can abort itself or a task image executing in another process. Abort Task is intended for use as an emergency or fault exit.

MACRO CALL ABRT$ *tsk*

parameter

tsk
VAX/VMS process name

dsw return codes

IS.SUC	Successful completion
IE.INS	Process name unknown (default error)
IE.PRI	User not privileged (SS$_NOPRIV)
IE.UPN	Insufficient dynamic memory (SS$_INSFMEM)
IE.NOD	Image's quota exceeded (SS$_EXQUOTA)
IE.ADP	Part of the Directive Parameter Block (DPB) is out of the address space of the issuing task image
IE.SDP	Directive Identification Code (DIC) or DPB size is invalid

notes

1 VAX-11 RSX executes a Force Exit system service to terminate the specified process's task image on behalf of the task image issuing the Abort Task directive.

2 The task image issuing the Abort Task directive must be executing in a process that meets either of the following requirements:

- It is in the same group as the process to be aborted and has GROUP privilege.
- It has WORLD privilege.

3 The exit status is supplied by an exit handling routine (exit handler). It is assumed that the status returns a severe error.

ALTP$—Alter Priority

VAX-11 RSX does not emulate the Alter Priority directive. Instead it allows the process for which the directive was issued to continue execution at its present priority.

MACRO CALL ALTP$ *[tsk][,pri]*

parameters

tsk
Active task name

pri
New priority, a number from 1 to 250 (10)

dsw return codes

IS.SUC	Successful completion
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

ALUN$—Assign Lun

The Assign LUN directive instructs the system to assign a physical device unit to a logical unit number. An I/O channel is the VAX/VMS equivalent of an RSX-11 Logical Unit Number (LUN).

The reassignment of a LUN from one device to another causes VAX-11 RSX to cancel all I/O requests for the previous assignment.

MACRO CALL ALUN$ *lun,dev,unt*

parameters

lun
Logical unit number

dev
Device name (two characters)

unt
Device unit number

dsw return codes

IS.SUC	Successful completion
IE.IDU	Device or unit unknown (default error)
IE.ILU	Invalid LUN
IE.LNL	LUN is locked, file open on it
IE.NOD	I/O channel quota exceeded (SS$_EXQUOTA)
IE.RSV	Device allocated to another image (SS$_DEVALLOC)
IE.DUN	Device not mounted (SS$_DEVNOTMOUNT)
IE.ADP	Part of the DPB is out of the address space of the issuing task
IE.SDP	DIC or DPB size is invalid

notes

1 VAX-11 RSX executes an Assign I/O Channel system service for the task image issuing the Assign LUN directive.

2 The assignment of RSX-11 device names to VAX/VMS physical devices is described in Section 2.

3 If a LUN is reassigned, its previous assignment is deassigned. The deassignment causes I/O to be canceled on the old assignment. If the attempt to make a new assignment fails, the LUN remains deassigned.

4 If a file is open on a LUN and you attempt to reassign the LUN, the attempt to reassign fails.

ASTX$S—AST Service Exit

The AST Service Exit directive instructs the system to terminate execution of an Asynchronous System Trap (AST) service routine.

If another AST is queued and ASTs are not disabled, VAX–11 RSX immediately effects the next AST. Otherwise, the system restores the task image's state prior to the AST.

MACRO CALL ASTX$S *[err]*

parameter

err
Error routine address

dsw return codes

IS.SUC	Successful completion
IE.AST	Directive not issued from an AST service routine
IE.ADP	Part of the DPB or stack is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 When an AST occurs, VAX–11 RSX pushes, at a minimum, the following information onto the stack:

SP+06,12	0
SP+04	PS of process prior to AST
SP+02	PC of process prior to AST
SP+00	DSW of process prior to AST

The stack must be in this state when the AST Service Exit directive is executed.

2 Because this directive requires only a one-word DPB, the $S form of the macro is recommended. It requires less space and executes with the same speed as that of the DIR$ macro.

CLEF$—Clear Event Flag

The Clear Event Flag directive instructs the system to report an indicated event flag's polarity and then to clear the flag.

MACRO CALL CLEF$ *efn*

parameter

efn
Event flag number

dsw return codes

IS.CLR	Successful completion; flag was already clear
IS.SET	Successful completion; flag was set
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX-11 RSX executes a Clear Event Flag system service for the task image issuing the Clear Event Flag directive.

2 A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

CMKT$—Cancel Mark Time Requests

The Cancel Mark Time Requests directive instructs the system to cancel a specific mark time request or all mark time requests that were made by the issuing task image.

MACRO CALL CMKT$ *[[efn][,[ast][,err]]]*

parameters

efn
Event flag numer

ast
Mark time AST routine address

err
Error routine address

dsw return codes

IS.SUC	Successful completion
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX executes a Cancel Timer Request system service indicating that the specified or all timer requests be canceled for the task image issuing the Cancel Mark Time Requests directive.

2 If either one or both of the efn and ast arguments are specified, then only mark time requests made with these arguments will be canceled. If neither the efn nor the ast is specified, then all mark time requests issued by the task are canceled.

CRGF$—Create Group Global Event Flags

The Create Group Global Event Flags directive instructs the system to associate a group global event flag cluster with the process that issued the directive. If the group cluster does not exist, this directive instructs the system first to create the group cluster (with all flags initialized to 0) and then to associate it with the process that issued the directive.

If a Create Group Global Event Flags directive is issued for an event flag cluster that has been marked for deletion (by the Eliminate Group Global Event Flags directive) but has not yet been deleted, the order for deletion is canceled.

MACRO CALL CRGF$ *[group]*

parameter

group

Group number for the flags to be created. If a group number is specified in the macro call, it must match the group number specified in the task header of the task image that issued the call. If the group number is omitted from the macro call, the group number specified in the task header (H.CUIC) is used.

dsw return codes

IS.SUC	Successful completion
IE.UPN	Insufficient memory (SS$_INSFMEM)
IE.IUI	Invalid group; group must match UIC in task header (H.CUIC)
IE.APD	Part of the DPB is out of the address space of the issuing task image
IE.DIC	DIC or DPB size is invalid

notes

1. VAX–11 RSX issues the Associate Common Event Flag Cluster system service for the task image issuing the Create Group Global Event Flags directive. This has the effect of locking the flags in place. That is, the flags cannot be eliminated by another task while they are being used by this task.
2. A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).
3. All member numbers for a given group have access to the event flags that are created for that group.

CSRQ$—Cancel Time Based Initiation Requests

The Cancel Time Based Initiation Requests directive instructs the system to cancel all time-synchronized wake requests for a specified process's task image regardless of the source of each request.

MACRO CALL CSRQ$ *tsk*

parameter

tsk

VAX/VMS process name

dsw return codes

IS.SUC	Successful completion
IE.INS	Specified process name unknown (default error)
IE.PRI	Privilege violation (SS$_NOPRIV)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1. VAX–11 RSX executes a Cancel Wakeup system service for the task image issuing the Cancel Time Based Initiation Requests directive.
2. The task image issuing the Cancel Time Based Initiation Requests directive must be executing in a process that meets either of the following requirements:
   - It is in the same group as the process for which requests are to be canceled and it has GROUP privilege.
   - It has WORLD privilege.

DECL$S—Declare Significant Event

The Declare Significant Event directive instructs the system to declare a significant event. Declaration of a significant event has no effect on a VAX-11 RSX system; the concept of a significant event in the RSX-11 sense does not exist under VAX/VMS.

MACRO CALL DECL$S *[,err]*

parameter

err
Error routine address

dsw return codes

IS.SUC	Successful completion
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 No operation is performed and a success status is returned.

2 Because this directive requires only a one-word DPB, the $S form of the macro is recommended. It requires less space and executes with the same speed as that of the DIR$ macro.

DSAR$S (or IHAR$S)—Disable (or Inhibit) AST Recognition

The Disable AST Recognition directive instructs the system to disable recognition of user-level ASTs for the issuing task image. The ASTs are queued as they occur and are effected when the task image enables AST recognition. When an AST service routine is executing, AST recognition also is disabled. The initial state of a task image is to have recognition enabled.

MACRO CALL

DSAR$S *[err]*
IHAR$S *[err]*

parameter

err
Error routine address

dsw return codes

IS.SUC	Successful completion
IE.ITS	AST recognition is already disabled
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 While disabled, ASTs are queued in a first-in/first-out list.

2 Because this directive requires only a one-word DPB, the $S form of the macro is recommended. It requires less space and executes with the same speed as that of the DIR$ macro.

DSCP$S—Disable Checkpointing

The Disable Checkpointing directive instructs the system to disable swapping for the process.

MACRO CALL DSCP$S *[err]*

parameter

err
Error routine address

dsw return codes

IS.SUC	Successful completion
IE.ITS	Swapping already disabled
IE.PRI	Privilege violation (SS$_NOPRIV)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX executes a Set Swap Mode system service for the task image issuing the Disable Checkpointing directive.

2 The task image's initial state has swapping enabled.

3 The requesting task image's process must have the privilege to set its swap mode (PSWAPM).

4 Because this directive requires only a one-word DPB, the $S form of the macro is recommended. It requires less space and executes with the same speed as that of the DIR$ macro.

ELGF$—Eliminate Group Global Event Flags

The Eliminate Group Global Event Flags directive instructs the system to disassociate the calling process from its group global event flag cluster.

If no other processes in this group are associated with a cluster thus marked for deletion, the cluster is deleted immediately. If, however, the cluster is still associated with other processes, it is not deleted until all these processes are disassociated from the cluster.

MACRO CALL ELGF$ *[group]*

parameter

group

Group number of flags to be eliminated. If a group number is specified in the macro call, it must match the group number specified in the task header of the task image that issued the call. If the group number is omitted from the macro call, the group number specified in the task header (H.CUIC) is used.

dsw return codes

IS.SUC	Successful completion
IE.IUI	Invalid group; group must match UIC in task header (H.CUIC)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.DIC	DIC or DPB size is invalid

notes

1 VAX–11 RSX issues the Disassociate Common Event Flag Cluster system service for the task image issuing the Eliminate Group Global Event Flags directive.

2 A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

ENAR$S—Enable AST Recognition

The Enable AST Recognition directive instructs the system to recognize user-level ASTs for the issuing task image; that is, the directive nullifies a Disable AST Recognition directive. ASTs that were queued while recognition was disabled are affected when the Enable AST Recognition directive is issued. The initial state of a task image is to have AST recognition enabled.

MACRO CALL ENAR$S *[err]*

parameter

err
Error routine address

dsw return codes

IS.SUC	Successful completion
IE.ITS	AST recognition is not disabled
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

note

Because this directive requires only a one-word DPB, the $S form of the macro is recommended. It requires less space and executes with the same speed as that of the DIR$ macro.

ENCP$S—Enable Checkpointing

The Enable Checkpointing directive instructs the system to enable swapping for the process.

MACRO CALL ENCP$S *[err]*

parameter

err
Error routine address

dsw return codes

IS.SUC	Successful completion
IE.ITS	Swapping already enabled
IE.PRI	Privilege violation (SS$_NOPRIV)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1. VAX-11 RSX executes a Set Swap Mode system service for the task image issuing the Enable Checkpointing directive.
2. The task image's initial state has swapping enabled.
3. The requesting task image's process must have the PSWAPM privilege to set its swap mode.
4. Because this directive requires only a one-word DPB, the $S form of the macro is recommended. It requires less space and executes with the same speed as that of the DIR$ macro.

EXIF$—Exit If

The Exit If directive instructs the system to terminate the execution of the issuing task image if the specified event flag is not set. VAX–11 RSX returns control to the issuing task image if the specified event flag is set.

MACRO CALL EXIF$ *efn*

parameter

efn
Event flag number

dsw return codes

IS.SET	Indicated event flag is set; task image did not exit
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

2 This service is interlocked in RSX–11; it is not interlocked in VAX–11 RSX. That is, under VAX–11 RSX, the conditional Exit If function is accomplished using two system service calls: Read Event Flags and, if necessary, Exit.

EXIT$S—Task Exit

The Task Exit directive instructs the system to terminate execution of the issuing task image.

MACRO CALL EXIT$S *[err]*

parameter

err
Error routine address

dsw return codes

IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1. A return to the task image occurs only if the directive is rejected.
2. VAX–11 RSX executes an Exit system service for the issuing task image. The success status is returned.
3. Because this directive requires only a one-word DPB, the $S form of the macro is recommended. It requires less space and executes with the same speed as that of the DIR$ macro.

EXST$—Exit With Status

The Exit With Status directive instructs VAX–11 RSX to terminate the execution of the issuing task image and to accept the status code indicating whether the termination is normal or abnormal from the task image.

MACRO CALL EXST$ *sts[,err]*

parameters

sts

Exit status

EX$SUC	Normal termination (RSX$_EXITSTATUS)
EX$WAR	Warning (RSX$_EXITSTATUS)
EX$ERR	Abnormal termination (RSX$_EXITSTATUS)
EX$SEV	Severe error termination (RSX$_EXITSTATUS)

err

Error routine address

dsw return codes

IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 A return to the task image occurs only if the directive is rejected.

2 VAX–11 RSX executes an Exit system service specifying the exit status of the task image.

3 The VAX/VMS exit status code (RSX$_EXITSTATUS) is the same in each case. However, VAX–11 RSX inserts a severity code appropriate to the RSX–11 exit status.

EXTK$—Extend Task

The Extend Task directive instructs the system to modify the size of the issuing task by a positive or negative increment of 32-word blocks. If the directive does not specify an increment value, VAX–11 RSX makes the issuing task image's size equal to it: initial size.

MACRO CALL EXTK$ *[inc]*

parameter

inc

A positive or negative number equal to the number of 32-word blocks by which the task image size is to be extended or reduced.

dsw return codes

IS.SUC	Successful completion
IE.ALG	The issuing task image attempted to reduce its size to less than the size of its header; or the task image tried to increase its size beyond 32K words or beyond the base of the lowest mapped library or common block
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 A task image cannot extend itself past its 64K byte (32K words) address space or, if libraries or common areas are present, past the base of the lowest mapped library or common block.

2 A task image can extend itself to the base of its read-only section.

GDVI$—Get Device Information

The Get Device Information directive instructs the system to fill a buffer with information about a physical device unit to which a LUN is assigned. This directive is an extension of the GLUN$ directive.

MACRO CALL

GDVI$ *lun,buf,len*

parameters

lun
Logical unit number

buf
Address of a buffer that is to receive the device information

len
Length of LUN information buffer in words

buffer format

Word 0	Name of assigned device.
Word 1	Unit number of assigned device in the low-order byte. The high-order bit of the word is set to indicate that the driver is loaded.

Word 2 First device characteristics word:

- Bit 0 Record-oriented device (DV.REC,1=yes) [FD.REC] [1]
- Bit 1 Carriage-control device (DV.CCL,1=yes) [FD.CCL]
- Bit 2 Terminal device (DV.TTY,1=yes) [FD.TTY]
- Bit 3 Directory (file structured) device (DV.DIR,1=yes) [FD.DIR
- Bit 4 Single directory device (DV.SDI,1=yes) [FD.SDI]
- Bit 5 Sequential device (DV.SQD,1=yes) [FD.SQD]
- Bit 6 Mass storage device (DV.MSD,1=yes)
- Bit 7 User-mode diagnostics supported (DV.UMD,1=yes); always 0 under VAX–11 RSX
- Bit 8 Device supports extended 22-bit UNIBUS controller (DV.EXT,DV.MBC,1=yes); always 0 under VAX–11 RSX
- Bit 9 Unit software write-locked (DV.SWL,1=yes)
- Bit 10 Input spooled device (DV.ISP,1=yes)
- Bit 11 Output spooled device (DV.OSP,1=yes)
- Bit 12 Pseudo device (DV.PSE,1=yes); a mailbox under VAX–11 RSX
- Bit 13 Device mountable as a communications channel (DV.COM,1=yes)
- Bit 14 Device mountable as a Files–11 device (DV.F11,1=yes)
- Bit 15 Device mountable (DV.MNT,1=yes)

Word 3,4 VAX/VMS device-dependent longword. The contents of this longword are described in the *VAX/VMS I/O Reference Volume.* For a disk, the longword contains the maximum number of blocks; this number is compatible with RSX–11.

Word 5 Standard device buffer size.

Word 6 RSX–11 device mnemonic (two ASCII characters that specify the physical device type).

[1] Bits with associated symbols defined in File Control Services (FCS) have the symbols shown in square brackets. These symbols can be defined for use by a task image with the FCSBT$ macro. See the *IAS/RSX–11 I/O Operations Reference Manual.*

dsw return codes

IS.SUC Successful completion

IE.ULN Unassigned LUN

IE.ILU Invalid LUN

IE.ADP Part of the DPB or buffer is out of the address space of the issuing task image

IE.SDP DIC or DPB size is invalid

notes

1 VAX–11 RSX executes a VAX/VMS Get Device Information system service for the task image issuing the Get Device Information directive.

2 VAX–11 RSX converts the name and unit number of the VAX/VMS device to which the LUN is assigned to an RSX–11 device name and unit number before returning the LUN information. Device name conversion is described in Section 2 of this manual.

3 Mailboxes may have unit numbers that are greater than eight bits. To refer to mailboxes from a compatibility mode task, a logical name must be used. The mailbox can be the task's SYS$INPUT file (TI0 or $I0) or have a $$n logical name as described in Section 2.

4 If the specified device is a spooled device, the device information returned by VAX–11 RSX is as follows:

Word 0	Name of assigned (spooled) device
Word 1	Unit number of assigned spooled device and flags byte
Word 2	Logical OR of the first device characteristics word for the intermediate device and the output spool bit (bit 11 of the spooled device first characteristics word)
Word 3	Spooled device fourth device characteristics word
Word 4	High word of VAX/VMS device-dependent characteristics longword
Word 5	Intermediate device standard device buffer size

5 Six words is the minimum length accepted for the device information buffer. If more than seven words are specified, the remaining words are not modified by VAX–11 RSX. However, the remaining words may be altered in future releases.

6 The device mnemonic returned in (optional) Word 6 is for use only as device characteristic data. It cannot be used to refer to the device in any RSX–11 directive (for example, the Assign LUN directive).

Table 7–2 lists the RSX–11 device mnemonics and corresponding physical device types. The directive Get Device Information returns an RSX–11 mnemonic of binary 0 (two ASCII NUL characters) for physical devices not listed in Table 7–2.

Table 7–2 RSX–11 Device Mnemonics and Physical Device Correspondence

Media	RSX–11 Mnemonic	Physical Device
Disk	DM	RK06, RK07
	DB	RP04, RP05, RP06
	DR	RM03, RP07, RP07HT, (head-per-track), RM80, RB80 (11/730 IDC RM80), RM05
	DL	RL01, RL02, RB02 (11/730 IDC RL02)
	DY	RX02, RX04
	DD	TU58
	DX	RX01
	DU	RA80, RA81, RA60, RC25,RX50, RD51, RCF25, RZ01, RZF01

Table 7–2 (Cont.) RSX–11 Device Mnemonics and Physical Device Correspondence

Media	RSX–11 Mnemonic	Physical Device
Tape	MM[1]	TE16, TU45, TU77
	MS	TS11
	MF	TU78
	MU	TA78, TU80, TU81, TA81
Synchronous Communication	PA	CI750/CI780
	XE	ni, UNA11, DEUNA, DEQNA
	XM	DMC11, DMR11
	XK	XK_3271 emulator
	XJ	XJ_2780 emulator
	XP	PCL11R, PCL11T
Card Reader	CR	All card readers
Terminal	TT	All terminals
Line Printer	LP	All line printers
Workstation	VA	VAXstation 100 series
Mailbox	MB	All mailboxes

[1] The mnemonic MM is compatible with RSX–11, but not with VAX/VMS.

GLUN$—Get LUN Information

The Get LUN Information directive instructs the system to fill a six-word buffer with information about a physical device unit to which a LUN is assigned.

MACRO CALL GLUN$ *lun,buf*

parameters

lun
Logical unit number

buf
Address of a six-word buffer that is to receive the LUN information

buffer format

Word 0	Name of assigned device.
Word 1	Unit number of assigned device in the low-order byte. The high-order bit of the word is set to indicate that the driver is loaded.

Word 2 First device characteristics word:

- Bit 0 Record-oriented device (DV.REC,1=yes) [FD.REC][1]
- Bit 1 Carriage-control device (DV.CCL,1=yes) [FD.CCL]
- Bit 2 Terminal device (DV.TTY,1=yes) [FD.TTY]
- Bit 3 Directory (file-structured) device (DV.DIR,1=yes) [FD.DIR]
- Bit 4 Single directory device (DV.SDI,1=yes) [FD.SDI]
- Bit 5 Sequential device (DV.SQD,1=yes) [FD.SQD]
- Bit 6 Mass storage device (DV.MSD,1=yes)
- Bit 7 User-mode diagnostics supported (DV.UMD,1=yes); always 0 under VAX–11 RSX
- Bit 8 Device supports extended 22-bit UNIBUS controller (DV.EXT,DV.MBC,1=yes); always 0 under VAX–11 RSX
- Bit 9 Unit software write-locked (DV.SWL,1=yes)
- Bit 10 Input spooled device (DV.ISP,1=yes)
- Bit 11 Output spooled device (DV.OSP,1=yes)
- Bit 12 Pseudodevice (DV.PSE,1=yes); a mailbox under VAX–11 RSX
- Bit 13 Device mountable as a communications channel (DV.COM,1=yes)
- Bit 14 Device mountable as a Files–11 device (DV.F11,1=yes)
- Bit 15 Device mountable (DV.MNT,1=yes)

Word 3,4 VAX/VMS device-dependent longword. The contents of this longword are described in the *VAX/VMS I/O Reference Volume*. For a disk, the longword contains the maximum number of blocks; this number is compatible with RSX–11.

Word 5 Standard device buffer size.

[1] Bits with associated symbols defined in File Control Services (FCS) have the symbols shown in square brackets. These symbols can be defined for use by a task image with the FCSBT$ macro. See the *IAS/RSX–11 I/O Operations Reference Manual*.

dsw return codes

IS.SUC Successful completion

IE.ULN Unassigned LUN

IE.ILU Invalid LUN

IE.ADP Part of the DPB or buffer is out of the address space of the issuing task image

IE.SDP DIC or DPB size is invalid

notes

1 VAX–11 RSX executes a Get Device Information system service for the task image issuing the Get LUN Information directive.

2 VAX–11 RSX converts the name and unit number of the VAX/VMS device to which the LUN is assigned to an RSX–11 device name and unit number before returning the LUN information. Device name conversion is described in Section 2.

3 Mailboxes may have unit numbers that are greater than eight bits. To reference mailboxes from a compatibility mode task, a logical name must be used. The mailbox can be the task's SYS$INPUT file (TI0 or $I0) or have a $$n logical name as described in Section 2.

4 If the specified device is a spooled device, the LUN information returned by VAX–11 RSX is as follows:

Word 0	Name of assigned (spooled) device
Word 1	Unit number of assigned spooled device and flags byte
Word 2	Logical OR of the first device characteristics word for the intermediate device and the output spool bit (bit 11 of the spooled device first characteristics word)
Word 3	Spooled device fourth device characteristics word
Word 4	High word of VAX/VMS device-dependent characteristics longword
Word 5	Intermediate device standard device buffer size

GMCR$—Get MCR Command Line

The Get MCR Command Line directive instructs the system to transfer an 80-byte command line to the issuing task image. The command line is used to invoke the task image. It can be in either MCR or DCL format.

MACRO CALL GMCR$

dsw return codes

+n	Successful completion; n is the number of data bytes transferred, excluding the termination character (the termination character is, however, in the buffer)
IE.AST	No command line exists for the issuing task image; that is, the task image was not requested by a command other than RUN or the task image has already issued the Get MCR Command Line directive
IE.ADP	Part of the DPB is out of the address space of the issuing process
IE.SDP	DIC or DPB size is invalid

notes

1 This directive is fully compatible with RSX-11 when issued in a process containing the MCR command interpreter. Under DCL, blanks that precede characters recognizable as significant terminators by DCL (for example, commas) are removed.

2 The system processes all lines to:

- Convert tabs to a single space.
- Convert multiple spaces to a single space.
- Convert lowercase characters to uppercase.
- Remove all trailing blanks.
- Translate all symbols known to the CLI (DCL only).

3 The terminator <CR> is the last character in the line.

4 The command line can be the result of the following types of user-issued DCL commands:

Format	Example
$ MCR name command-string	*$ MCR PIP LP:=MYFILE*
$ MCR	*$ MCR*
MCR> name command-string	*MCR> PIP LP:=MYFILE*

Format	Example
$ name:="$name"	*$ PIP:="$PIP"*
$ name command-string	*$ PIP LP:=MYFILE*

Format	Example
$ MCR	*$ MCR*
MCR> name	MCR> *PIP*
followed by prompt	PIP> *LP:=MYFILE*

5 The command line can be the result of the following types of user-issued MCR commands:

Format	Example
> name command-string	*> PIP LP:=MYFILE*
> name	*> PIP*
followed by prompt	*PIP> LP:=MYFILE*

6 The command line received as a result of the Get MCR Command Line directive varies depending on the format of the command typed. If the command contains a command string, for example, LP:=MYFILE, the command containing that string and its length are available to the image.

Example:

```
$ MCR PIP LP:=MYFILE
> PIP LP:=MYFILE
```

Both of these commands result in the same string being made available to the image using the GMCR$ directive. The actual string that GMCR$ will return is:

```
PIP LP:=MYFILE
```

with a length of 14 (10).

If no command string is supplied, VAX–11 RSX returns a command string length of 0.

7 When a task image or native image executes as a result of a RUN command (either DCL or MCR), the command line length is 0.

GPRT$—Get Partition Parameters

The Get Partition Parameters directive instructs the system to fill an indicated three-word buffer with partition parameters. The VAX/VMS system does not have the concept of partitions. Therefore, the data returned is consistent with that returned by RSX–11 for a system-controlled partition named GEN starting at 40000 (8).

MACRO CALL GPRT$ *[prt],buf*

parameters

prt
Partition name

buf
Address of a three-word buffer

buffer format

Word 0	400 (8)
Word 1	Image size, including all libraries, expressed as a multiple of 64 bytes
Word 2	A value of 0 is returned to indicate a system-controlled partition

dsw return codes

IE.ADP	Part of the DPB or buffer is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

Successful completion is indicated by carry clear and $DSW equal to 0 indicating a mapped system.

GTIM$—Get Time Parameters

The Get Time Parameters directive instructs the system to fill an indicated eight-word buffer with the current time parameters. All time parameters are delivered as binary numbers. The value ranges are shown in decimal below.

MACRO CALL GTIM$ *buf*

parameter

buf
Address of an eight-word buffer

buffer format

Word 0	Year (since 1900)
Word 1	Month (1–12)
Word 2	Day (1–31)
Word 3	Hour (0–23)
Word 4	Minute (0–59)
Word 5	Second (0–59)
Word 6	Tick of second (0–99)
Word 7	Ticks per second (100 ticks occur per second)

dsw return codes

IS.SUC	Successful completion
IE.ADP	Part of the DPB or buffer is out of the address space of the issuing image
IE.SDP	DIC or DPB size is invalid

notes

1. VAX–11 RSX executes a Get Time system service for the task image issuing the Get Time Parameters directive.
2. VAX/VMS provides a 100-tick-per-second clock.

GTSK$—Get Task Parameters

The Get Task Parameters directive instructs the system to fill an indicated 16-word buffer with parameters relating to the issuing process.

MACRO CALL GTSK$ *buf*

parameter

buf
Address of a 16-word buffer

buffer format

Word 0	Process name in Radix–50 (first half), if any
Word 1	Process name in Radix–50 (second half), if any
Word 2	Partition name in Radix–50 (GEN)
Word 3	Partition name in Radix–50 (blanks)
Word 4	Undefined
Word 5	Undefined
Word 6	Run priority from RSX–11 task header
Word 7	Default directory if in UIC format; otherwise, equal to protection UIC
Word 8	Number of logical I/O units (LUNs)
Word 9	Undefined
Word 10	Undefined
Word 11	Address of task SST vector tables
Word 12	Size of task SST vector table in words
Word 13	Size in bytes of task image's address window excluding libraries and common areas
Word 14	System in which process is running; 5 for VAX–11 RSX
Word 15	Process (protection) UIC

dsw return codes

IS.SUC	Successful completion
IE.ADP	Part of the DPB or buffer is out of the address space of the issuing task image
IE.SDP	DIC or DPB is invalid

note

Under VAX/VMS, process UICs contain 16-bit group and member numbers. Under RSX–11, UICs contain 8-bit group and member numbers. VAX-11 RSX maps the entire UIC to [377,377] if either the process's group or member number exceeds the 8-bit maximum. See Section 2 for more information about UICs and default directories.)

MRKT$—Mark Time

The Mark Time directive instructs the system to set an event flag and/or declare an AST after an indicated time interval. The interval begins when the task image issues the directive. If an event flag is specified, the flag is cleared when the directive is issued and set when the interval elapses. If an AST entry point address is specified, an AST occurs when the interval elapses.

MACRO CALL MRKT$ *[efn],tmg,tnt[,ast]*

parameters

efn
Event flag number

tmg
Time interval magnitude

tnt
Time interval unit

ast
AST entry point address

dsw return codes

IS.SUC	Successful completion
IE.ITI	Invalid time parameter
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.UPN	Insufficient dynamic memory (SS$_INSFMEM)
IE.NOD	Task image's quota exceeded (SS$_EXQUOTA)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX executes a Set Timer system service for the process issuing the Mark Time directive.

2 If an AST entry point address is specified, the AST service routine is entered with the stack in the following state:

SP+08,14	0
SP+06	PS of process prior to AST
SP+04	PC of process prior to AST
SP+02	DSW of process prior to AST
SP+00	Event flag number or 0 (if none was specified in the Mark Time directive)

The event flag number must be removed from the stack before an AST Service Exit directive is executed.

VAX–11 RSX pushes four words of zeros in SP+14 through SP+08. RSX-11 pushes three words with undefined contents and a one-word event flag mask.

3 VAX–11 RSX returns the DSW code IE.ITI if the directive specifies an invalid time parameter. The time parameter consists of two components: the time interval magnitude (tmg) and the time interval unit (tnt).

A legal magnitude value (tmg) is related to the value assigned to the time interval unit (tnt). The unit values are encoded as follows:

1 = Ticks (1/100 of a second per tick)
2 = Seconds
3 = Minutes
4 = Hours

The magnitude (tmg) is the number of units to be clocked; it cannot exceed 24 hours. The following list describes the magnitude values that are valid for each type of unit:

- If tnt = 0, 1, or 2, tmg can be any positive value with a maximum of 15 bits.
- If tnt = 3, tmg can have a maximum value of 1440 (10).
- If tnt = 4, tmg can have a maximum value of 24 (10).

4 A task image must be associated with group global event flags to access flags in the group global cluster (refer to Section 3).

5 VAX/VMS enforces a quota on the number of ASTs a process can have pending.

QIO$—Queue I/O Request

The Queue I/O Request directive instructs the system to place an I/O request for an indicated physical device unit into a queue of priority-ordered requests for that device unit. The physical device unit is specified as a Logical Unit Number (LUN).

If the directive call specifies an event flag, VAX–11 RSX clears the flag when the request is queued and sets the flag when the request is completed.

The I/O status block is also cleared when the request is queued, and it is also set to the final I/O status when the I/O request is completed. If an entry point address is specified for an AST service routine, the AST occurs when the I/O request is completed, and the process's WAITFOR mask word, PS, PC, and DSW (directive status), and the address of the I/O status block are pushed onto the stack.

MACRO CALL QIO$ *fnc,lun,[efn],[pri],[isb],[ast][,prl]*

parameters

fnc
I/O function code

lun
Logical unit number

efn
Event flag number

pri
Priority; ignored, but must be present in the DPB

isb
Address of I/O status block

ast
Address of AST service routine entry point

prl
Parameter list of the form <P1,...,P6>

dsw return codes

IS.SUC	Successful completion
IE.ULN	Unassigned LUN
IE.ILU	Invalid LUN
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.NOD	Quota exceeded (SS$_EXQUOTA)
IE.UPN	Insufficient memory (SS$_INSFMEM)
IE.ADP	Part of the DPB or I/O status block is out of the address space of the task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX executes a Queue I/O Request system service for the task image issuing the Queue I/O Request directive.

2 Section 7 explains function codes, parameter meanings, and I/O status block return values.

3 If the directive call specifies an AST entry point address, the process enters the AST service routine with the stack in the following state:

SP+10,16	0
SP+06	PS of process prior to AST
SP+04	PC of process prior to AST
SP+02	DSW of process prior to AST
SP+00	Address of I/O status block, or 0 if none was specified in the QIO directive

The address of the I/O status block, which is a trap-dependent parameter, must be removed from the stack before an AST Service Exit directive is executed.

VAX–11 RSX pushes four words of zeros in SP+16 through SP+10. RSX-11 pushes three words with undefined contents and a one-word event flag mask.

4 If the directive is rejected, the specified event flag is not guaranteed to be cleared or set. Therefore a process that waits for a rejected Queue I/O Request directive may wait indefinitely. Care should always be taken to ensure that the directive completes successfully.

5 A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

6 VAX/VMS enforces a quota on the number of ASTs a process can have pending.

QIOW$—Queue I/O Request And Wait

The Queue I/O Request And Wait directive is identical to Queue I/O Request with one exception: when the wait variation of the directive specifies an event flag, VAX–11 RSX automatically effects a Wait For Single Event Flag directive.

MACRO CALL QIOW$ *fnc,lun,efn,[pri],[isb],[ast][,prl]*

parameters

fnc
I/O function code

lun
Logical unit number

efn
Event flag number

pri
Priority; ignored, but must be present in the DPB

isb
Address of I/O status block

ast
Address of AST service routine entry point

prl
Parameter list of the form <P1,...,P6>

dsw return codes

IS.SUC	Successful completion
IE.ULN	Unassigned LUN
IE.ILU	Invalid LUN
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.UPN	Insufficient memory (SS$_INSFMEM)
IE.NOD	Quota exceeded (SS$_EXQUOTA)
IE.ADP	Part of the DPB or I/O status block is out of the address space of the task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX-11 RSX executes a Queue I/O Request and Wait for Event Flag system service for the image issuing the Queue I/O Request And Wait directive.

2 See the notes for the Queue I/O Request directive.

RCST$—Receive Data Or Stop

The Receive Data Or Stop directive instructs the system to attempt to dequeue a message received by the mailbox created for the issuing task image when the task was loaded. If no data has been sent to the task by another process, the issuing task is stopped, or placed in hibernation. This directive cannot be issued from an AST service routine.

A 2-word sending process name in Radix–50 form and the 13-word data block are returned in a 15-word buffer.

MACRO CALL RCST$ *[tname],buf*

parameters

tname
Sending task name (if not specified, data may be received from any task)

buf
Address of a 15-word buffer

dsw return codes

IS.SUC	Successful completion
IE.NOD	Quota exceeded (SS$_EXQUOTA)
IE.UPN	Insufficient memory (SS$_INSFMEM)
IE.SET	No data obtained; process stopped
IE.AST	Directive was issued from an AST service routine and no data was obtained
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX dequeues the oldest message received for the task. If the sender task name is specified, the oldest message received from that task is dequeued.

2 The name of the mailbox is RCVD followed by the process name, that is, RCVDname.

3 The mailbox is not created until the task image actually begins to execute, so a task cannot send data to the mailbox before the task is running.

4 Because the logical name of the receive data mailbox is qualified with the group number of the owner (in the receiving process), this directive is useful only for passing data between processes within the same group.

5 If no data is obtained from the mailbox or if no mailbox exists, VAX–11 RSX executes a Hibernate system service for the process that issued the Receive Data Or Stop directive, and a status code of IS.SET is returned. Note that the status code IS.SET cannot be seen by the process issuing the directive until the process is restarted by an Unstop Task directive (a Wake system service).

6 If a process that has issued the Receive Data Or Stop directive stops because no data is available, the process is not automatically awakened later when data is received. A task sending data should also wake the stopped task using the Unstop Task directive; to receive the data, the awakened task must issue another receive-data directive (Receive Data, Receive Data Or Exit, or Receive Data Or Stop).

RCVD$—Receive Data

The Receive Data directive instructs the system to read a message from the mailbox created when the task image was loaded and to place that message in a buffer within the issuing task image.

A 2-word sending process name in Radix–50 form and the 13-word data block are returned in a 15-word buffer.

MACRO CALL RCVD$ *[tsk],buf*

parameters

tsk
Sending task name (if not specified, data may be received from any task)

buf
Address of a 15-word buffer

dsw return codes

IS.SUC	Successful completion
IE.NOD	Quota exceeded (SS$_EXQUOTA)
IE.UPN	Insufficient memory (SS$_INSFMEM)
IE.ITS	No data currently available
IE.ADP	Part of the DPB or buffer is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX dequeues the oldest message received for the task. If the sender task name is specified, the oldest message received from that task is dequeued.

2 The name of the mailbox is RCVD followed by the process name, that is, RCVDname.

3 The mailbox is not created until the task actually begins to execute, so another task cannot send data to the mailbox before the task is running.

4 Because the logical name of a temporary mailbox is implicitly qualified by VAX/VMS with the group number of the owner (in the receiving process), this directive is useful only for passing data among processes within the same group.

RCVX$—Receive Data Or Exit

The Receive Data Or Exit directive instructs the system to read a message from the mailbox created when the image was loaded. The message has been sent to the mailbox previously by another process. If no data has been sent, the task image exits.

A 2-word sending process name in Radix–50 form and the 13-word data block are returned in a 15-word buffer.

MACRO CALL RCVX$ *[tsk],buf*

parameters

tsk
Sending task name (if not specified, data may be received from any task)

buf
Address of a 15-word buffer

dsw return codes

IS.SUC	Successful completion
IE.NOD	Quota exceeded (SS$_EXQUOTA)
IE.UPN	Insufficient memory (SS$_INSFMEM)
IE.ADP	Part of the DPB or buffer is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX dequeues the oldest message received for the task. If the sender task name is specified, the oldest message received from that task is dequeued. If no data is obtained, VAX–11 RSX executes an Exit system service for the task image. The exit status is SS$_NORMAL.

2 The name of the mailbox is RCVD followed by the process name, that is, RCVDname.

3 The mailbox is not created until the task actually begins to execute, so another task cannot send data to the mailbox before the task is running.

4 Because the logical name of a temporary mailbox is implicitly qualified by VAX/VMS with the group number of the owner (in the receiving process), this directive is useful only for passing data among processes within the same group.

5 This directive does not provide the same interlock between the sender and the receiver as it does in RSX–11.

RDAF$—Read All Event Flags

The Read All Event Flags directive instructs the system to read local and common event flags for the issuing process and to record their values in a 64-bit (4-word) buffer.

MACRO CALL RDAF$ *buf*

parameter

buf
Address of a 4-word buffer

buffer format

Word 0	Local flags 1 through 16
Word 1	Local flags 17 through 32
Word 2	Common flags 33 through 48
Word 3	Common flags 49 through 64

dsw return codes

IS.SUC	Successful completion
IE.ADP	Part of the DPB or buffer is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX issues a Read Event Flags system service for the task image issuing the Read All Event Flags directive.

2 The Read All Event Flags directive does not read group global flags. The Read Extended Event Flags directive reads all 96 flags.

RDXF$—Read Extended Event Flags

The Read Extended Event Flags directive instructs the system to read all local, common, and group global event flags for the issuing process and to record their values in a 96-bit (6-word) buffer.

MACRO CALL RDXF$ *buf*

parameter

buf
Address of a 6-word buffer

buffer format

Word 0	Local flags 1 through 16
Word 1	Local flags 17 through 32
Word 2	Common flags 33 through 48
Word 3	Common flags 49 through 64
Word 4	Group global flags 65 through 80
Word 5	Group global flags 81 through 96

dsw return codes

IS.SUC	Successful completion
IS.CLR	Group global event flags do not exist; words 4 and 5 of the buffer contain zeros
IE.ADP	Part of the DPB or buffer is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX issues the Read Event Flags system service for the task image issuing the Read Extended Event Flags directive.

2 A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

3 If no group global event flag cluster is associated with the process, the group global event flags are returned as all zeros and IS.CLR is returned.

RQST$—Request Task

The Request Task directive instructs the system to activate a hibernating process. Section 3 describes the use of the Hibernate and Wake system services for real-time images.

Request Task is a frequently used subset of the Run directive.

MACRO CALL RQST$ *tsk,[prt],[pri][,ugc][,umc]*

parameters

tsk
VAX/VMS process name

prt
Partition name; ignored

pri
Priority; ignored

ugc
UIC group code; ignored

umc
UIC member code; ignored

dsw return codes

IS.SUC	Successful completion
IE.INS	Process name not known (default error)
IE.PRI	Privilege violation (SS$_NOPRIV)
IE.UPN	Insufficient dynamic memory (SS$_INSFMEM)
IE.NOD	Process quota exceeded (SS$_EXQUOTA)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1. VAX-11 RSX executes a Wake system service for the process issuing the Request Task directive.
2. The requested process must currently be present in the system; that is, either hibernating or active.
3. The task image issuing the Request Task directive must be executing in a process that meets either of the following requirements:
   - It is in the same group as the requested process and has GROUP privilege.
   - It has WORLD privilege.

4 VAX/VMS maintains an indicator to determine whether any wake requests have been issued for an active process. If the pending wake indicator is set and the process issues a hibernate request, the process remains active, and the pending wake indicator is cleared. A subsequent hibernate request causes the process to hibernate.

5 Hibernations caused by Stop directives (receive Data Or Stop, Stop, Stop For Logical OR Of Event Flags, and Stop For Single Event Flag) cannot be reactivated by the Request Task directive.

RSUM$—Resume Task

The Resume Task directive instructs the system to awaken a process that is in a state of hibernation.

MACRO CALL RSUM$ *tsk*

parameter

tsk
VAX/VMS process name

dsw return codes

IS.SUC	Successful completion
IE.INS	Process name unknown (default error)
IE.PRI	Privilege violation (SS$_NOPRIV)
IE.UPN	Insufficient dynamic memory (SS$_INSFMEM)
IE.NOD	Image's quota exceeded (SS$_EXQUOTA)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX executes a Wake system service for the task issuing the Resume Task directive.

2 The task image issuing the Resume Task directive must be executing in a process that meets either of the following requirements:

- The process is in the same group as the process to be resumed and has GROUP privilege.
- The process has WORLD privilege.

3 VAX/VMS maintains an indicator to determine whether any wake requests have been issued for an active process. If the indicator is set and the process issues a hibernate request, the process remains active, and the indicator is cleared. A subsequent hibernate (Suspend) request causes the process to hibernate.

4 If a Resume Task directive is issued for a task image that is active, the status returned is success. The process remains active.

5 Hibernations that are caused by Stop directives (Receive Data Or Stop, Stop, Stop For Logical OR Of Event Flags, and Stop For Single Event Flag) cannot be reactivated by the Resume Task directive.

RUN$—Run Task

The Run Task directive requests the system to activate a hibernating process at a specified future time and, optionally, to reactivate that process periodically. The schedule time is specified by delta time from issuance. If the smg, rmg, and rnt parameters are omitted, Run Task is the same as Request Task except that Run Task causes the process to become active one clock tick after the directive is issued.

MACRO CALL

RUN$ *tsk,[prt],[pri],[ugc],[umc],smg,snt,[,rmg][,rnt]*

parameters

tsk
VAX/VMS process name

prt
Partition name; ignored

pri
Priority; ignored

ugc
UIC group code; ignored

umc
UIC member code; ignored

smg
Schedule delta magnitude

snt
Schedule delta unit

rmg
Reschedule interval magnitude

rnt
Reschedule interval unit

dsw return codes

IS.SUC	Successful completion
IE.INS	Process name unknown
IE.PRI	Privilege violation (SS$_NOPRIV)
IE.UPN	Insufficient dynamic memory (SS$_INSFMEM)
IE.NOD	Process quota exceeded (SS$_EXQUOTA)
IE.ITI	Invalid time parameter
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX executes a Schedule Wake-up system service for the process issuing the Run Task directive.

2 The target process must be present in the system.

3 The task image issuing the Run Task directive must be executing in a process that meets either of the following requirements:

- It is in the same group as the process to be run and has GROUP privilege.
- It has WORLD privilege.

4 VAX/VMS maintains an indicator to determine whether any wake requests have been issued for an active process. If the wake pending indicator is set and the process issues a hibernate request, the process remains active, and the wake pending indicator is cleared. A subsequent hibernate (Suspend) request causes the process to hibernate.

5 VAX–11 RSX returns the DSW code IE.ITI if the directive specifies an invalid time parameter. A time parameter consists of two components: the time interval magnitude (smg or rmg) and the time interval unit (snt or rnt).

A legal magnitude value (smg or rmg) is related to the value assigned to the time interval unit snt or rnt. The unit values are encoded as follows:

1 = Ticks (1/100 of a second per tick)
2 = Seconds
3 = Minutes
4 = Hours

The magnitude is the number of units to be clocked. In no case can the magnitude exceed 24 hours. The following list describes the magnitude values that are valid for each type of unit:

- If unit equals 0, 1, or 2, the magnitude can be any positive value with a maximum of 15 bits.
- If unit equals 3, the magnitude can have a maximum value of 1440 (10).
- If unit equals 4, the magnitude can have a maximum value of 24 (10).

6 The schedule delta time is the difference in time from the issuance of the Run Task directive to the time the process is to be run. This time can be specified in the range from one-clock-tick to 24 hours.

7 Hibernations caused by Stop directives (Receive Data Or Stop, Stop, Stop For Logical OR Of Event Flags, and Stop For Single Event Flag) cannot be reactivated by the Run Task directive.

SDAT$—Send Data

The Send Data directive instructs the system to send a 13-word message to a mailbox.

When an event flag is specified in the Send Data directive, the indicated flag is set for the sending process.

MACRO CALL SDAT$ *tsk,buf[,efn]*

parameters

tsk
VAX/VMS process name to send to

buf
Address of a 13-word data buffer

efn
Event flag number

dsw return codes

IS.SUC	Successful completion
IE.INS	Receiver process name unknown (default error)
IE.NOD	Quota exceeded (SS$_EXQUOTA)
IE.UPN	Insufficient memory (SS$_INSFMEM)
IE.PRI	Privilege violation (SS$_NOPRIV)
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.ADP	Part of DPB or data block is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX-11 RSX executes a write Queue I/O Request system service for the process issuing the Send Data directive. The I/O operation writes to a mailbox named RCVD followed by the specified process name, that is, RCVDname.

2 The sending process must be in the same group as the receiving process because the logical name of the receive data mailbox is qualified by the owner's group number.

3 A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

SETF$—Set Event Flag

The Set Event Flag directive instructs the system to set an indicated event flag and to report the flag's previous value.

MACRO CALL SETF$ *efn*

parameter

efn
Event flag number

dsw return codes

IS.CLR	Flag was clear
IS.SET	Flag was already set
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1. VAX–11 RSX executes a Set Event Flag system service for the task image issuing the Set Event Flag directive.
2. A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

SFPA$—Specify Floating Point Processor Exception AST

The Specify Floating Point Processor Exception AST directive instructs the system either to enable or to disable delivery of floating point processor exception ASTs.

When an entry point address for an AST service routine is specified, future floating point processor exception ASTs occur for the issuing process, and control is transferred to the indicated location at the time of the AST's occurrence. When an entry point address for an AST service is not specified, future floating point processor exception ASTs do not occur until the task image issues a directive that specifies an AST entry point.

MACRO CALL SFPA$ *[ast]*

parameter

ast

AST service routine entry point address

dsw return codes

IS.SUC	Successful completion
IE.UPN	Insufficient dynamic memory (SS$_INSFMEM)
IE.ITS	AST entry point address is already unspecified
IE.AST	Directive was issued from an AST service routine or ASTs are disabled
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 The Specify Floating Point Processor Exception AST requires dynamic memory.

2 VAX-11 RSX queues floating point processor exception ASTs when a floating point processor exception trap occurs for the task. No future floating point processor exception ASTs are queued for the process until the first one queued has actually been effected.

3 The Floating Point Processor Exception AST service routine is entered with the task stack in the following state:

SP+12,18	Event flag mask word
SP+10	PS of task prior to AST
SP+06	PC of task prior to AST
SP+04	DSW of task prior to AST
SP+02	Floating exception code
SP+00	Floating exception address

The image must remove the floating point exception code and address from the stack before an AST SERVICE EXIT directive is executed.

VAX-11 RSX pushes four words of zeros in SP+18 through SP+12. RSX 11 pushes three words with undefined contents and a one-word event fla mask.

See Section 3 for more information concerning floating point exceptions.

4 This directive cannot be issued from an AST service routine or when AST are disabled.

SPND$S—Suspend

The Suspend directive instructs the system to place the process in a state of hibernation. A task image can suspend only the process in which it is executing. The suspended process can be restarted by another process that issues a Resume directive for it.

MACRO CALL SPND$S *[err]*

parameter

err
Error routine address

dsw return codes

IS.SUC	Successful completion
IE.INS	Process has no name
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX-11 RSX executes a Hibernate system service for the process issuing the Suspend directive.

2 A suspended process retains control of the system resources allocated to it. VAX/VMS makes no attempt to free these resources.

3 VAX/VMS maintains an indicator to determine whether any wake requests have been issued for an active process. If the indicator is set and the process issues a hibernate request, the process remains active, and the indicator is cleared. A subsequent hibernate request causes the process to hibernate.

4 If a Suspend directive is issued by an image that has pending resume requests, the following things occur:

- The status returned is success.
- The process remains active.
- The wake-pending indicator is cleared.

5 A process can be resumed only by specifying its process name. Therefore, a process is not allowed to suspend itself unless it has a process name.

6 A process hibernating because of a Suspend directive cannot be reactivated by the Unstop Task directive.

7 Because this directive requires only a one-word DPB, the $S form of the macro is recommended. It requires less space and executes with the same speed as that of the DIR$ macro.

SPRA$—Specify Power Recovery AST

The Specify Power Recovery AST directive instructs the system to record either of the following:

- Power-recovery ASTs for the issuing process and the address to which to transfer control when a powerfail-recovery AST occurs are required.
- Power-recovery ASTs for the issuing process are no longer required.

When an entry point address for an AST service routine is specified future power-recovery ASTs occur for the issuing process. VAX-11 RSX transfers control to the specified address whenever a powerfail recovery occurs. When an AST service entry point address is not specified, future power-recovery ASTs do not occur until an AST entry point is again specified.

MACRO CALL SPRA$ *[ast]*

parameter

ast
AST service routine entry point address

dsw return codes

IS.SUC	Successful completion
IE.ITS	AST entry point address is already unspecified
IE.AST	Directive was issued from an AST service routine or ASTs are disabled
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1. VAX-11 RSX executes a Set Power Recovery AST system service for the task image issuing the Specify Power Recovery AST directive.
2. ASTs are disabled while the AST service routine executes. They remain disabled until the service routine issues an AST Service Exit directive.
3. The process enters the powerfail AST service routine with the task stack in the following state:

SP+06,12	0
SP+04	PS of process prior to AST
SP+02	PC of process prior to AST
SP+00	DSW of process prior to AST

No trap-dependent parameters accompany a power-recovery AST; therefore, the AST Service Exit directive can be executed with the stack in the same state as when the AST was effected.

VAX-11 RSX pushes four words of zeros in SP+12 through SP+06. RSX-11 pushes three words with undefined contents and a one-word event flag mask.

4 This directive cannot be issued from an AST service routine or when ASTs are disabled.

SPWN$—Spawn

The Spawn directive instructs the system to create a subprocess to execute a specified task image or (optionally) to execute a command line.

The specified task image is invoked and executed by the subprocess. If the parameters unum and dnam are not specified, the task's TI device is the TI device of the parent process.

If a CLI task is specified, a mailbox is created and the command line is written to it. Because the input part of TI ($I), the SYS$INPUT for the subprocess, is set equivalent to the mailbox, the command interpreter reads the command from the mailbox. The output part of TI ($O) will be the parent process's SYS$OUTPUT, or the terminal that the unum and dnam parameters specify.

When the created subprocess terminates, its exit status is delivered to the termination AST routine. A termination mailbox is created for this purpose. VAX–11 RSX reports the termination status to the spawning task in the first word of the exit status block.

MACRO CALL

SPWN$ *tname,,,[ugc],[umc],[efn],[east],[esb],[cmdlin,* *[cmdlen],[unum],[dnam]*

parameters

tname
Name of task image to be executed in subprocess

ugc
Unused

umc
Unused

efn
Event flag to be set when subprocess terminates

east
Address of termination AST routine

esb
Address of exit status block, an eight-word buffer containing the exit status in the first word; the other words are unused

cmdlin
Address of command line to be executed

cmdlen

Length of command line

unum

Unit number of TI device for process; if omitted, parent's TI is used

dnam

Device name of TI device for process; if omitted, parent's TI is used

dsw return codes

IS.SUC	Successful completion
IE.UPN	Insufficient dynamic memory (SS$_INSFMEM)
IE.INS	Task name invalid, or task was a CLI, but no command line was specified
IE.ACT	Process name already exists
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.ADP	Part of the DPB, exit status block, or command, line is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 Task names "MCR" , "MCR..." , "CLI" , and "CLI..." request the MCR CLI. Task names "DCL" and "DCL..." request the DCL CLI.

2 When a command line that changes the process context (for example, SET DEFAULT or SET/UIC=[g,n]) is spawned to and executed by MCR (in a subprocess), it will change the context of the parent process as well as that of the subprocess. When an equivalent command line is spawned to and executed by DCL (in a subprocess), only the context of the subprocess is changed.

3 A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

4 The subprocess this directive creates will have a process name that is the same as the specified task name. However, if another process running in the same UIC group has the same process name, VAX–11 RSX modifies the subprocess name to make it unique.

VAX–11 RSX modifies the subprocess name by appending the string ".1" to the process name. Then, if necessary, VAX–11 RSX increments the number from 1 until a unique process name is achieved. When a CLI task is spawned, the ellipsis (...) is dropped from the task name. Spawning MCR results in a process name of "MCR" , "MCR.1" , "MCR.2" , and so forth.

5 VAX–11 RSX attempts to translate the specified task name (if it is not a CLI task) to a VAX/VMS logical name. The translation will be used to specify the task image file to be executed in the subprocess. The subprocess name will be the logical name, not the actual task image name to which it translates.

SRDA$—Specify Receive Data AST

The Specify Receive Data AST directive instructs the system to record either of the following conditions:

- Receive-data ASTs for the issuing task image are required as well as the address to which to transfer control when data has been placed in the task image's mailbox (RCVDprocessname).
- The Receive-data ASTs for the issuing task are no longer required.

When the directive specifies an entry point address for an AST service routine, receive-data ASTs for the task image occur whenever data has been placed in the task image's mailbox (RCVDprocessname). VAX–11 RSX transfers control to the specified address.

When the directive omits an entry point address, VAX–11 RSX disables receive-data ASTs for the issuing task image. Receive-data ASTs will not occur until the task image issues another Specify Receive Data AST directive indicating an entry point address.

MACRO CALL SRDA$ *[ast]*

parameter

ast
AST service routine entry point address

dsw return codes

IS.SUC	Successful completion
IE.ITS	AST entry point address is already unspecified
IE.AST	Directive was issued from an AST service routine or ASTs are disabled
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 The task enters the receive-data AST service routine with the task stack in the following state:

SP+06,12	0
SP+04	PS of process prior to AST
SP+02	PC of process prior to AST
SP+00	DSW of process prior to AST

No trap-dependent parameters accompany a receive-data AST. Therefore, the AST Service Exit directive must be executed with the stack in the same state as when the AST was effected.

VAX–11 RSX pushes four words of zeros in SP+12 through SP+06. RSX–11 pushes three words with undefined contents and a one-word event flag mask.

2 This directive cannot be issued from an AST service routine or when ASTs are disabled.

3 Refer to the description of the Receive Data directive for more information.

STLO$—Stop For Logical OR Of Event Flags

The Stop For Logical OR Of Event Flags directive instructs the system to stop the execution of the issuing task image until VAX–11 RSX sets one or more of the indicated event flags from one of the following groups:

GR0	Local flags 1 through 16
GR1	Local flags 17 through 32
GR2	Common flags 33 through 48
GR3	Common flags 49 through 64
GR4	Group global flags 65 through 80
GR5	Group global flags 81 through 96

The process does not stop if any of the indicated flags is already set when it issues the directive.

This directive cannot be issued from an AST service routine.

A process that is stopped because none of the indicated event flags is set can be restarted only when one or more of the specified event flags is set. Such a stopped process cannot become unstopped by an Unstop Task directive.

MACRO CALL STLO$ *grp,msk*

parameters

grp
Event flag group

msk
A 16-bit mask word

dsw return codes

IS.SUC	Successful completion
IE.AST	Directive was issued from an AST service routine
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX executes a Wait for Logical OR of Event Flags system service for the task image issuing the Stop For Logical OR Of Event Flags directive.

2 A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

3 There is a one-to-one correspondence between bits in the mask word and the event flags in the specified group. That is, if group 1 was specified, bit 0 in the mask word would correspond to event flag 17, bit 1 would correspond to event flag 18, and so forth.

4 VAX–11 RSX does not arbitrarily clear event flags when Stop For Logical OR Of Event Flags conditions are met. Some directives, (for example, Queue I/O Request) clear event flags; otherwise, the event flags must be explicitly cleared by the Clear Event Flag directive.

5 A task stopped for event flag conditions cannot be unstopped by issuing the Unstop directive. It can be unstopped only by meeting event flag conditions.

6 The grp operand must always be of the form n regardless of the macro form used. In all other macro calls, numeric or address values for the $S form macro have the following form:

```
#n
```

STOP$S—Stop

The Stop directive instructs the system to place a process in a state of hibernation. A Stop directive can stop only the process that issues the directive. This directive cannot be issued from an AST service routine.

A process stopped by the Stop directive can be restarted only by an Unstop Task directive issued by one of its own ASTs or by another process.

MACRO CALL STOP$S

dsw return codes

IE.SET	Successful completion
IE.INS	Process has no name
IE.AST	Directive was issued from an AST service routine
IE.ADP	Part of the DPB is out of address space of the the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX-11 RSX executes a Hibernate system service for the process issuing the Stop directive.

2 A stopped process retains control of the system resources allocated to it. VAX/VMS makes no attempt to free these resources.

3 VAX/VMS maintains an indicator to determine whether any wake requests have been issued for an active process. If the wake pending indicator is set and the process issues a hibernate request, the process remains active, and the wake pending indicator is cleared. A subsequent hibernate request causes the process to hibernate.

Thus, if a Stop directive is issued by a process that has pending unstop requests, the following things occur:

- The status returned is success.
- The process remains active.
- The wake pending indicator is cleared.

4 A process stopped by use of the Stop directive can be restarted (by use of the Unstop Task directive) only if its process name is specified. Therefore, a process is not allowed to stop itself unless it has a process name.

5 Because this directive requires only a one-word DPB, the $S form of the macro is recommended. It requires less space and executes with the same speed as that of the DIR$ macro.

STSE$—Stop For Single Event Flag

The Stop For Single Event Flag directive instructs the system to stop the execution of the issuing task image until the specified event flag is set. If the flag is set when the directive is issued, task image execution continues. This directive cannot be issued from an AST service routine.

A process stopped because a specified event flag is not set can be restarted only when that event flag is set. Such a stopped address space cannot be restarted by Ask directive.

MACRO CALL STSE$ *efn*

parameter

efn

Event flag number

dsw return codes

IS.SUC	Successful completion
IE.AST	Directive was issued from an AST service routine
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size invalid

notes

1 VAX–11 RSX executes a Wait for Logical OR of Event Flags system service for the image issuing the Stop For Single Event Flag directive.

2 A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

SVDB$—Specify SST Vector Table For Debugging Aid

The Specify SST Vector Table For Debugging Aid directive instructs the system to record the address of a table of SST service routine entry points.

To deassign the vector table, the parameters adr and len are omitted from the macro call.

When an SST service routine entry is specified in both the table used by the task image and the table used by a debugging aid, the trap occurs for the debugging aid, not for the task image.

MACRO CALL SVDB$ *[adr][,len]*

parameters

adr
Address of SST vector table

len
Length (in number of entries) of the table in words

vector table format

Word 0	Odd address or nonexistent memory error
Word 1	Memory protection violation
Word 2	T-bit trap or execution of a BPT instruction
Word 3	Execution of an IOT instruction
Word 4	Execution of an illegal or reserved instruction
Word 5	Execution of a non-RSX EMT instruction
Word 6	Execution of a TRAP instruction
Word 7	Not used

A table entry with a value of 0 indicates that the task image will not process the corresponding SST.

dsw return codes

IS.SUC	Successful completion
IE.ADP	Part of the DPB or table is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

SVTK$—Specify SST Vector Table For Task

The Specify SST Vector Table For Task directive instructs the system to record the address of a table of SST service routine entry points.

To deassign the vector table, the parameters adr and len are omitted from the macro call.

When an SST service routine entry is specified in both the table used by the task image and the table used by a debugging aid, the trap occurs for the debugging aid, not for the task image.

MACRO CALL SVTK$ *[adr][,len]*

parameters

adr

Address of SST vector table

len

Length (in number of entries) of the table in words

vector table format

Word 0	Odd address or nonexistent memory error
Word 1	Memory protection violation
Word 2	T-bit trap or execution of a BPT instruction
Word 3	Execution of an IOT instruction
Word 4	Execution of an illegal or reserved instruction
Word 5	Execution of a non-RSX EMT instruction
Word 6	Execution of a TRAP instruction
Word 7	Not used

A table entry with a value of 0 indicates that the task image will not process the corresponding SST.

dsw return codes

IS.SUC	Successful completion
IE.ADP	Part of the DPB or table is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

USTP$—Unstop Task

The Unstop Task directive instructs the system to awaken a process that is in a state of hibernation. An Unstop Task directive restarts a specified process that has stopped itself by either a Stop directive or a Receive Data Or Stop directive. The Unstop Task directive does not restart processes stopped for an event flag.

If the Unstop Task directive is issued to a process that is executing an AST service routine and if that process was previously stopped by either a Stop directive or a Receive Data Or Stop directive, the process becomes unstopped only when the execution of the AST service routine has been completed.

MACRO CALL USTP$ *[tname]*

parameter

tname

VAX/VMS process name

dsw return codes

IS.SUC	Successful completion
IE.INS	Process name unknown
IE.PRI	Privilege violation (SS$_NOPRIV)
IE.UPN	Insufficient dynamic memory (SS$_INSFMEM)
IE.NOD	Task image's quota exceeded (SS$_EXQUOTA)
IE.ADP	Part of the DPB is out of the address space of the issuing image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX-11 RSX executes a Wake system service for the process issuing the Unstop Task directive.

2 The process issuing an Unstop Task directive must meet one of the following requirements:

- It must have the same UIC as the process to be unstopped.
- It must be in the same group as the process to be unstopped, and it must have the GROUP privilege.

Otherwise, a process needs no privileges to issue an Unstop Task directive.

3 VAX/VMS maintains an indicator to determine whether any wake requests have been issued for an active process. If the indicator is set and the process issues a hibernate request, the process remains active, and the indicator is cleared. A subsequent hibernate request causes the process to hibernate.

Thus, if a Stop directive is issued by a process that has pending unstop requests, the following things occur:

- The status returned is success.
- The process remains active.
- The wake-pending indicator is cleared.

4 If an Unstop Task directive is issued for a task image that is active, the status returned is a success. The process remains active.

WSIG$S—Wait For Significant Event

Because significant events do not exist in VAX–11 RSX, the Wait For Significant Event directive does not correspond to any VAX/VMS system service request. No wait occurs when the direc tive is issued.

MACRO CALL WSIG$S *[err]*

parameter

err
Error routine address

dsw return codes

IS.SUC	Successful completion
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

note

Because this directive requires only a one-word DPB, the $S form of the macro is recommended. It requires less space and executes with the same speed as that of the DIR$ macro.

WTLO$—Wait For Logical OR Of Event Flags

The Wait For Logical OR Of Event Flags directive instructs the system to block the execution of the issuing task image until VAX–11 RSX sets an indicated event flag from one of the following groups:

GR 0	Flags 1 to 16
GR 1	Flags 17 to 32
GR 2	Flags 33 to 48
GR 3	Flags 49 to 64
GR 4	Flags 65 to 80
GR 5	Flags 81 to 96

The process does not wait if any of the indicated flags are already set when it issues the directive.

MACRO CALL WTLO$ *grp,msk*

parameters

grp
Event flag group

msk
A 16-bit flag mask word

dsw return codes

IS.SUC	Successful completion
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1 VAX–11 RSX executes a Wait for Logical OR of Event Flags system service for the task image issuing the Wait For Logical OR Of Event Flags directive.

2 A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

3 There is a one-to-one correspondence between bits in the mask word and the event flags in the specified group. That is, if group 1 was specified, bit 0 in the mask word would correspond to event flag 17, bit 1 would correspond to event flag 18, and so forth.

4 VAX-11 RSX does not arbitrarily clear event flags when Wait For Logical OR Of Event Flags conditions are met. Some directives, (for example, Queue I/O Request) clear event flags; otherwise, the event flags must be explicitly cleared by the Clear Event Flag directive.

5 The grp operand must always be of the form n regardless of the macro form used. In all other macro calls, numeric or address values for the $S form macro have the following form:

```
#n
```

WTSE$—Wait For Single Event Flag

The Wait For Single Event Flag directive instructs the system to block the execution of the issuing task image until the indicated event flag is set. If the flag is set when the directive is issued, execution of the task image continues.

MACRO CALL WTSE$ *efn*

parameter

efn
Event flag number

dsw return codes

IS.SUC	Successful completion
IE.IEF	Invalid event flag number because (1) in the range 65 to 96, but no associated group global event flags; or (2) not in the range 1 to 96 (EFN is less than 1 or greater than 96)
IE.ADP	Part of the DPB is out of the address space of the issuing task image
IE.SDP	DIC or DPB size is invalid

notes

1. VAX-11 RSX executes a Wait for Logical OR of Event Flags system service for the image issuing the Wait For Single Event Flag directive.
2. A task image must be associated with group global event flags to access flags in the group global cluster (see Section 3).

8 I/O Functions

When an RSX–11 task image running under VAX–11 RSX issues a Queue I/O Request directive, the I/O request is processed by the VAX/VMS I/O system and the function is performed by a standard VAX/VMS device driver or Ancillary Control Processor (ACP). Usually, RSX–11 I/O requests correspond to similar VAX/VMS requests. As a result, the RSX–11 task image I/O occurs as expected. However, if a task image issues an I/O request that depends on characteristics of the RSX–11 I/O system not present in the VAX/VMS I/O system, the requested I/O operation may not occur exactly as expected. This section describes what should be done in those cases.

Each RSX–11 I/O request consists of a function code, function-independent parameters, and function-dependent parameters. When VAX–11 RSX receives a Queue I/O Request directive, it forms the equivalent VAX/VMS arguments for each RSX–11 parameter specified in the directive. Because VAX–11 RSX issues QIOs using the VAX/VMS I/O system, it must convert RSX–11 QIOs to the native format for processing by the appropriate driver or ACP.

VAX–11 RSX handling of RSX–11 function-independent parameters, for example, efn, lun, and ast, is described in Section 3 and in the description of the Queue I/O Request directive in Section 7. This section describes how VAX–11 RSX handles I/O function codes and I/O function-dependent parameters.

8.1 Standard I/O Functions

The standard RSX–11 I/O functions are attach, detach, and cancel I/O; read and write virtual block; and read and write logical block. These functions are supported for all devices in VAX–11 RSX. The sections that follow provide additional information about attach, detach, and cancel I/O.

8.1.1 Attach and Detach I/O Device (IO.ATT and IO.DET)

VAX/VMS categorizes devices as shareable and nonshareable. A shareable device, such as a disk, can be accessed by many users without affecting the integrity of the data. A nonshareable device, such as a terminal, allows access from only one process at a time. When a task image assigns a channel to a nonshareable device, VAX/VMS implicitly allocates the device for exclusive use by the process. In the RSX–11 sense, the system attaches the device for the process. Because VAX/VMS performs implicit allocation, task images do not have to allocate and deallocate nonshareable devices explicitly during execution.

If a task image must have exclusive access to a shareable device, the device can be allocated in either of two ways:

- By a task image issuing an Allocate Device system service (native mode only)

- By the user typing an ALLOCATE command to the MCR or DCL command interpreter

Note that using the ALLOCATE command has one advantage over the Allocate Device system service. It eliminates the need for error recovery by the task image if the device is not available for allocation.

Tasks running in RSX–11 frequently attach terminals and other devices to prevent another task from using them. These devices are shareable in an RSX–11 system. When an RSX–11 task image running under VAX–11 RSX issues a QIO to attach a device, VAX–11 RSX performs no operation and returns a success status to the task image. If the target device is nonshareable, VAX–11 RSX allocates the device when the task image assigns a LUN to it. In effect, therefore, the device is attached. If the device is shareable, it remains unallocated after the directive status is returned. When an RSX–11 task image requires a shareable device to be allocated, the device must be allocated from a terminal or an indirect command file by using an ALLOCATE command.

NOTE: **The ALLOCATE command is useful in an indirect command file only if the task that requires use of the allocated device is invoked from the same command file, or by the original command file in a chain of command files.**

8.1.2 Cancel I/O Requests (IO.KIL)

When an RSX–11 task image issues IO.KIL for a VAX/VMS device, VAX–11 RSX executes a Cancel I/O on Channel system service and all I/O issued from the designated channel are canceled. Under RSX–11, on the other hand, all I/O from the issuing task to the device are canceled regardless of which LUN the I/O originated from. When a Cancel I/O on Channel system service is issued for a disk device under VAX–11 RSX, no operation is performed; VAX–11 RSX returns a success status to the task image.

When the Cancel I/O on Channel system service executes, it notifies the driver immediately. Queued I/O requests are canceled immediately; however, I/O that the driver is currently processing is not necessarily canceled.

8.2 I/O Status Block and Status Return Codes

When VAX/VMS completes an I/O operation, it returns a code indicating the status of the request in an I/O status block. When an RSX–11 task image issues a request, VAX–11 RSX returns the status information in an I/O status block that has the standard RSX–11 format, as illustrated in Figure 8–1.

The return code can be IS.SUC or any of the success error status codes listed in Table 8–1. The status code IS.SUC corresponds to the VAX/VMS status code SS$_NORMAL. See the *VAX/VMS System Messages and Recovery Procedures Reference Manual* for more detailed information on the VAX/VMS error codes listed in Table 8–1.

The high-order byte of word 0 always contains a 0 except in the case of terminal I/O read requests. For a terminal read request, that byte indicates the line terminator, as described in 8.6.14.

Figure 8–1 Format of RSX–11 I/O Status Block Under VAX–11 RSX

	Byte 1	Byte 0
Word 0	0 except for terminal read	Status code
Word 1	Number of bytes transferred	

ZK-851-82

The second word of the I/O status block contains the number of bytes read or written.

Table 8–1 I/O Status Return Codes

DSW Code	VAX/VMS Code	Meaning
IE.ABO	SS$_ABORT	An I/O request was canceled before the operation was completed, or a network link was broken.
	SS$_CANCEL	An I/O request was canceled before the operation was completed.
IE.ALN	SS$_FILALRACC	A DECnet–VAX logical link already existed or was pending.
IE.BAD	SS$_BADPARAM	A call to a network or file ACP contained invalid parameters.
IE.BCC	SS$_DATACHECK	A data check found a mismatch between disk data and memory data.
	SS$_FORMAT	The media format is invalid.
IE.BDR	SS$_BADIRECTORY	A file specified as a directory either was not a directory or contained invalid data.
IE.BDV	SS$_NOTFILEDEV	A file specification contained references to a directory or a file on a non-file-structured device.
IE.BHD	SS$_BADFILEHDR	A file header contained invalid data; for example, the structure was not consistent or the storage map indicated free blocks.
	SS$_FILESTRUCT	The file structure on an accessed volume was invalid for the called ACP.
	SS$_NOTLABELMT	A request to read a tape failed because the tape did not have standard labels.
IE.BLK	SS$_ILLBLKNUM	The logical block number specified for a file did not exist on disk.
	SS$_IVADDR	A disk address specified in a physical disk read or write operation was not valid.

Table 8–1 (Cont.) I/O Status Return Codes

DSW Code	VAX/VMS Code	Meaning
IE.BNM	SS$_BADFILENAME	A file name contained illegal characters or was longer than nine characters.
IE.BVR	SS$_BADFILEVER	A file version number was greater than 32,767.
	SS$_TOOMANYVER	The maximum number of versions for a file already existed and all had higher version numbers than the specified version number.
IE.BYT	SS$_BUFBYTALI	The accessed device does not support byte-aligned transfers.
	SS$_BUFNOTALIGN	The buffer is not properly aligned.
IE.CKS	SS$_BADCHKSUM	The checksum in a file header was invalid.
IE.CLO	SS$_FILELOCKED	A process attempted to access a locked file.
IE.CNR	SS$_REJECT	A request to connect to an object at a remote network node failed.
IE.DAA	SS$_DEVALLOC	An allocate request specified a device already allocated to another user.
IE.DAO	SS$_BUFFEROVF	A buffer was not large enough for a string output by a system service; the string was truncated.
	SS$_DATAOVERUN	A buffer was not large enough for data output by a system service.
	SS$_MBTOOSML	A mailbox was too small for data sent to it.
IE.DFU	SS$_DEVICEFULL	A device was full or did not have enough contiguous blocks to fill a request.
	SS$_DIRFULL	A file could not be created because the specified directory was full.
IE.DNA	SS$_DEVNOTALLOC	A deallocate request specified a device that was not allocated.
IE.DNR	SS$_DEVACTIVE	The device is currently busy.
	SS$_DEVFOREIGN	A Files–11 operation was attempted on a device that is foreign-mounted.
	SS$_DEVNOTMOUNT	A dismount request specified a device that was not mounted.
	SS$_VOLINV	A volume-valid bit for a requested volume was not set.
IE.DSQ	SS$_EXDISKQUOTA	A process exceeded its disk quota.
IE.DUP	SS$_DUPFILENAME	A specified file already existed in the specified directory.

Table 8–1 (Cont.) I/O Status Return Codes

DSW Code	VAX/VMS Code	Meaning
IE.EOF	SS$_ENDOFFILE	The end of a file was reached. Specifically, an EOF mark on a tape, an EOF card in a card reader, an empty mailbox, or the end of virtual memory was reached.
IE.EOT	SS$_BEGOFFILE	A backspace operation reached the beginning of a file.
	SS$_ENDOFTAPE	The end-of-tape mark was encountered on a tape.
IE.EOV	SS$_ENDOFVOLUME	The end of a volume was encountered.
IE.EXP	SS$_FILNOTEXP	A file could not be written or deleted because it had not reached its expiration date.
IE.FHE	SS$_BLOCKCNTERR	An error occurred while reading a record from a magnetic tape.
	SS$_CRTLERR	A hardware controller failed during an I/O operation.
	SS$_DRVERR	A device driver failed during an I/O operation.
	SS$_TAPEPOSLOST	The position of the magnetic tape was lost by the magnetic tape ACP.
	SS$_UNSAFE	A device driver was unusable.
IE.HFU	SS$_HEADERFULL	A file-header map was full and its extension was inhibited.
IE.IES	SS$_BADESCAPE	A terminal escape sequence was invalid.
IE.IFC	SS$_ILLCNTRFUNC	The control function specified for an ACP was invalid.
	SS$_ILLIOFUNC	The function code specified for an explicit I/O request was invalid.
IE.IFU	SS$_IDXFILEFUL	The index file on a volume is full.
IE.NDR	SS$_NOLINKS	A logical network link could not be created because no more slots were available.
IE.NLN	SS$_FILNOTACC	No file was accessed on a channel that was specified for an I/O operation.
IE.NNN	SS$_NOSUCHNODE	A specified network node did not exist.
IE.NOD	SS$_ACPVAFUL	A file ACP could not access a volume because it had no more virtual memory for volume service.
	SS$_EXQUOTA	A process attempted to exceed its limit or quota for a resource.
	SS$_INSFMEM	More system dynamic memory was required than was available.

Table 8–1 (Cont.) I/O Status Return Codes

DSW Code	VAX/VMS Code	Meaning
	SS$_INSFWSL	A process required more pages in its working set than it was allowed.
IE.NSF	SS$_NOMOREFILES	No more files matching a wildcard specification existed; at least one matching file was previously found.
	SS$_NOSUCHFILE	A specified file did not exist.
IE.OFL	SS$_DEVOFFLINE	A device went off line.
	SS$_MEDOFL	A requested device had no medium mounted on it (such as a tape or a disk).
IE.PES	SS$_PARTESCAPE	A terminal escape sequence was truncated at the end of its buffer; the remainder of the sequence was written in the type-ahead buffer.
IE.PRI	SS$_NOPRIV	A process requested initialization of a volume that it did not have the privilege to write.
IE.RER	SS$_FCPREADERR	An error occurred in reading file control data (such as a directory).
	SS$_FCPREPSTN	An I/O error occurred during an attempt to reposition a tape volume after the tape volume went off line.
	SS$_FCPREWNDERR	An error occurred in rewinding a volume.
	SS$_FCPSPACERR	An I/O error occurred while a file control primitive was skipping spaces within or among files.
IE.RSU	SS$_MBFULL	A mailbox could not accept another message because it was full.
	SS$_VECINUSE	The CTRL/C vector for a process was already in use by a controlling process.
IE.SNC	SS$_FILENUMCHK	The index file for a volume contained invalid data.
IE.SPC	SS$_ACCVIO	A process attempted to access memory outside its virtual address space; the program counter (PC) contains the address of the invalid instruction.
IE.SQC	SS$_FILESEQCHK	The file sequence number in a file header was invalid; the directory entry pointed to an obsolete or deleted file.
IE.TMO	SS$_TIMEOUT	An input operation was not completed within the specified timeout period.
IE.VER	SS$_PARITY	A device-dependent error occurred.
IE.WAC	SS$_ACCONFLICT	The access protection for a file did not allow a requested access.

Table 8–1 (Cont.) I/O Status Return Codes

DSW Code	VAX/VMS Code	Meaning
IE.WAT	SS$_BADATTRIB	An invalid attribute was specified for read or write access using a file ACP.
IE.WER	SS$_FCPWRITERR	An I/O error occurred while a file control primitive was writing.
IE.WLK	SS$_WRITLCK	The hardware write-lock switch was set on a requested disk.
IS.CHW	SS$_CHAINW	A chained write was completed.
IS.PND	None	An I/O request is pending.
IS.RDD	SS$_RDDELDATA	A read operation of a physical block that contains deleted data marks (see Table 8–2, Write Physical Block with Deleted Data Mark) succeeded.
IS.SUC	SS$_NORMAL	An operation succeeded.

8.3 Disk Driver

Table 8–2 shows the correspondence between RSX–11 disk function codes and VAX/VMS disk function codes.

Table 8–2 Disk Function Code Correspondence

Function	RSX–11 Code	VAX/VMS Code or Action
Attach Device	IO.ATT	No operation
Detach Device	IO.DET	No operation
Cancel I/O Requests	IO.KIL	Cancel I/O on Channel system service
Read Logical Block	IO.RLB	IO$_READLBLK
Sense Diskette Characteristics (only for RX02 diskettes)	IO.SEC	Get I/O Channel Information system service
Set Media Density (only for RX02 diskettes)	IO.SMD	IO$_FORMAT
Write Logical Block	IO.WLB	IO$_WRITELBLK
Read Virtual Block	IO.RVB	IO$_READVBLK
Write Virtual Block	IO.WVB	IO$_WRITEVBLK
Read Physical Block	IO.RPB	IO$_READPBLK
Write Physical Block	IO.WPB	IO$_WRITEPBLK
Write Physical Block with Deleted Data Mark (only for RX01 and RX02 diskettes)	IO.WDD	IO$_WRITEPBLK!IO$M_DELDATA

Table 8–2 (Cont.) Disk Function Code Correspondence

Function	RSX–11 Code	VAX/VMS Code or Action
Load Overlay	IO.LOV	Special form of IO.RLB performed only on OV (overlay device); an IO.RVB is performed on LUNs not assigned to OV
Pack Acknowledge	IO.STC	IO$_PACKACK

Table 8–3 shows the correspondence between RSX–11 function-dependent parameters and VAX/VMS arguments.

Table 8–3 Disk Parameter Correspondence

Parameter Function	RSX–11 Pn	VAX/VMS Pn
Starting buffer address (stadd)	P1	P1
Buffer size (size)	P2	P2
High-block number (bklh)	P4	P3 (high half of longword)
Low-block number (blkl)	P5	P3 (low half of longword)

8.4 Magnetic Tape Driver

Table 8–4 shows the correspondence between RSX–11 magnetic tape function codes and VAX/VMS magnetic tape function codes.

Table 8–4 Magnetic Tape Function Code Correspondence

Function	RSX–11 Code	VAX/VMS Code or Action
Attach Device	IO.ATT	No operation
Detach Device	IO.DET	No operation
Cancel I/O Requests	IO.KIL	Cancel I/O on Channel system service
Read Logical Block	IO.RLB	IO$_READLBLK
Write Logical Block	IO.WLB	IO$_WRITELBLK
Read Virtual Block	IO.RVB	IO$_READVBLK
Write Virtual Block	IO.WVB	IO$_WRITEVBLK
Write End-of-File Mark	IO.EOF	IO$_WRITEOF
Read Logical Block Reverse	IO.RLV	IO$_READPBLK!IO$M_REVERSE
Rewind Unit	IO.RWD	IO$_REWIND
Rewind and Turn Unit Off Line	IO.RWU	IO$_REWINDOFF

Table 8–4 (Cont.) Magnetic Tape Function Code Correspondence

Function	RSX–11 Code	VAX/VMS Code or Action
Mount Tape and Set Characteristics	IO.SMO	IO$_SETMODE (only parity and density can be set)
Sense Tape Characteristics	IO.SEC	IO$_SENSEMODE
Space Blocks	IO.SPB	IO$_SPACERECORD
Space Files	IO.SPF	IO$_SPACEFILE
Set Tape Characteristics	IO.STC	IO$_SETMODE (only parity and density can be set)

Table 8–5 shows the correspondence of RSX–11 function-dependent parameters and VAX/VMS arguments.

Table 8–5 Magnetic Tape Parameter Correspondence

Parameter Function	RSX–11 Pn	VAX/VMS Pn
Starting buffer address (stadd)	P1	P1
Buffer size (size)	P2	P2
Characteristic bits (cb) of IO.SMO and IO.STC	P1	P1
Number of blocks to space past (nbs) of IO.SPB	P1	P1
Number of EOFs to space past (nes) of IO.SPF	P1	P1

8.5 Line Printer Driver

Table 8–6 shows the correspondence between RSX–11 line printer function codes and VAX/VMS function codes or resultant action.

Table 8–6 Line Printer Function Code Correspondence

Function	RSX–11 Code	VAX/VMS Code or Action
Attach Device	IO.ATT	No operation
Detach Device	IO.DET	No operation
Cancel I/O Requests	IO.KIL	Cancel I/O on Channel system service
Write Logical Block	IO.WLB	IO$_WRITELBLK
Write Virtual Block	IO.WVB	IO$_WRITEVBLK
Write Physical Block	IO.WPB	IO$_WRITEPBLK

Table 8–7 shows the correspondence between RSX–11 function-dependent parameters and VAX/VMS arguments.

Table 8–7 Line Printer Parameter Correspondence

Parameter Function	RSX–11 Pn	VAX/VMS Pn
Starting buffer address (stadd)	P1	P1
Buffer size (size)	P2	P2
Vertical format control character (vfc)	P3	P4

When using VAX/VMS line printers, keep in mind the following points:

- VAX/VMS line printers are not shareable. VAX/VMS implicitly allocates a line printer when a channel is assigned.
- VAX/VMS line printers normally are spooled. A spooled printer is allocated to the print symbiont. VAX/VMS does not allow a process to allocate a spooled device unless it has the privilege to do so. An RSX–11 task image is not allowed exclusive use of a spooled device (for example, a printer) unless the process in which it is running has the necessary privilege and the ALLOCATE command has been issued to reserve the device prior to task image execution.
- If a printer is allocated or not spooled, the RSX–11 task image's IO.WLB and IO.WVB requests for it produce exactly the same results as in the RSX–11 operating system.
- See Section 2 for a discussion of the requirements for issuing IO.WLB and IO.WVB requests to a spooled device.
- If an RSX–11 task image issues a GLUN$ directive for a spooled device, the information returned is that for the intermediate device, including the bit DV.OSP (as in RSX–11M–PLUS).

8.6 Terminal Driver

Table 8–8 shows the correspondence between RSX–11 function codes and VAX/VMS functions. Table 8–9 shows the correspondence between RSX–11 function-dependent parameters P1 through P6 and their VAX/VMS equivalents for terminal devices. Table 8–10 lists the subfunction bits applicable for each RSX–11 function code and provides notes describing the VAX–11 RSX handling of these subfunctions for terminals.

VAX–11 RSX places restrictions on the I/O functions that can be performed on TI, CO, and CL, because they may be mapped to process-permanent files. It places the same restrictions on I/O to user-created process-permanent files. These restrictions are described in 8.6.14.

Table 8–8 Terminal Function Code Correspondence

Function	RSX–11 Code	VAX/VMS Code or Action
Attach Device	IO.ATT	Terminal not attached; forces Cancel CTRL/O on next write
Detach Device	IO.DET	Terminal not detached; forces Cancel CTRL/O on next write
Cancel I/O Requests	IO.KIL	Cancel I/O on Channel system service
Read Logical Block	IO.RLB	IO$_READLBLK
Write Logical Block	IO.WLB	IO$_WRITELBLK
Read Virtual Block	IO.RVB	IO$_READVBLK
Write Virtual Block	IO.WVB	IO$_WRITEVBLK
Read Physical Block	IO.RPB	IO$_READPBLK
Write Pass All	IO.WAL	IO$_WRITEPBLK
Read Logical Block after Prompt	IO.RPR	IO$_READPROMPT
Get Multiple Characteristics	SF.GMC	IO$SENSEMODE
Set Multiple Characteristics	SF.SMC	IO$_SETMODE
Get Terminal Support	IO.GTS	Standard data returned

Table 8–9 Terminal Parameter Correspondence

Parameter Function	RSX–11 Pn	VAX/VMS Pn
Starting buffer address (stadd)	P1	P1
Buffer size (size)	P2	P2
Vertical format control character (vfc) on write	P3	P4[1]
Timeout count (tmo) on read with prompt	P3	P3
Prompt address (pradd) for read with prompt	P4	P5
Prompt size (prsize) for read with prompt	P5	P6
Vertical format control character (vfc) for read with prompt	P6	none

[1] For all read functions except IO.RPB and IO.RST, the VAX/VMS P4 parameter specifies RETURN, ESCAPE, and CTRL/Z as terminators. For IO.RPB, no characters are terminators. For IO.RST, P4 is 0 specifying that all characters with a value less than an ASCII space are terminators except form feed, vertical tab, backspace, delete, and horizontal tab.

NOTE: The remaining device-specific function codes are the equivalent of the logical OR or of a subfunction bit and one of the standard function code IO.ATT, IO.RLB, or IO.WLB. See Table 8-10.

Table 8–10 Subfunction Bit Correspondence

APPLICABLE SUBFUNCTION BITS

Empty square = Indicates that subfunction in given column cannot be used with function in given row.

X = Corresponds directly to VAX/VMS function.

FUNCTION	EQUIVALENT WITH SUBFUNCTION BIT	TF.AST	TF.BIN	TF.CCO	TF.ESQ	TF.RAL	TF.RNE	TF.RST	TF.WAL	TF.WBT	TF.XOF
STANDARD FUNCTIONS:											
IO.ATT		1			2						
IO.DET											
IO.KIL											
IO.RLB						3	X	X			
IO.RVB						4	4	4			
IO.RPB							X				
IO.WLB				X					X	2	
IO.WVB				X					4	4	
IO.WPB											
DEVICE-SPECIFIC FUNCTIONS:											
IO.ATA	IO.ATT!TF.AST (see Note 1)				2						
IO.CCO	IO.WLB!TF.CCO								X		
SF.GMC											
IO.GTS											
IO.RAL	IO.RLB!TF.RAL (see Note 3)						X	X			
IO.RNE	IO.RLB!TF.RNE					3		X			
IO.RPR			2			3	X	X			2
IO.RST	IO.RLB!TF.RST					3	X				
SF.SMC											
IO.WAL	IO.WLB!TF.WAL			X						2	
IO.WBT	IO.WLB!TF.WBT (see Note 2)			X					X		

NOTES:

1. No attach performed. Enables for unsolicited character or CTRL/C ASTs only. See 8.6.1 of this section.
2. Subfunction bit ignored for one of the following reasons.

Function	**Reason**
TF.ESQ, TF.XOF	These are characteristics of the terminal line and cannot be controlled on a per-request basis.
TF.WBT	The write breakthrough function is not supported in VAX/VMS. See 8.6.8 of this section.
TF.BIN	Function is not supported in VAX/VMS.

3. Sets the VAX/VMS function modifier IO$M_NOFILTR. See 8.6.4.1 of this section.
4. RSX-11 virtual functions do not accept these subfunction bits.

ZK-1647-84

Two differences between the VAX–11 RSX terminal driver and the RSX–11 terminal driver can affect applications that display or request input from a terminal. These differences concern the RETURN and LINE FEED control characters sent to the terminal, as follows:

1 When an RSX–11 task image displays a message using a terminal write request, one difference occurs only if both of the following conditions occur:

   - The last character written or echoed to the terminal was a LINE FEED character.
   - The first character of the next write request is a LINE FEED character.

   If both conditions occur, VAX/VMS does not send the leading LINE FEED character in the next write request. The fact that VAX/VMS sends one, not two, LINE FEED characters changes the appearance of the output as seen by the terminal user. RSX–11 sends both LINE FEED characters under the same conditions.

2 When an RSX–11 task image requests input using a terminal read request, a difference always occurs when the terminal user ends the read request by pressing the RETURN key, which generates a RETURN character. VAX/VMS automatically echoes the RETURN character and a LINE FEED character to the terminal screen, whereas RSX–11 only echoes a RETURN character. Again, the terminal user may notice the difference on the output.

However, when the task image executes a terminal read request terminated by a RETURN character that is followed by a terminal write request with a leading LINE FEED character, the two differences cancel each other, thus giving the screen the same appearance under VAX–11 RSX as under RSX–11.

3.6.1 IO.ATT Function

When an RSX–11 task image uses the IO.ATT function for a terminal, VAX–11 RSX performs no operation to alter the attached/detached status of the terminal, as described in 8.1.1. VAX–11 RSX does, however, issue a request to the terminal driver to cancel CTRL/O on the next operation to the terminal if that operation is a write. The VAX/VMS terminal driver subfunction modifier to cancel CTRL/O is IO$M_CANCTRLO. The RSX–11 terminal driver also forces a Cancel CTRL/O (TF.CCO) on a write operation that follows an attach operation.

An IO.ATT function issued for TI, CO, or CL is ignored if the specified device is not a terminal.

8.6.1.1 IO.ATT!TF.AST and IO.ATA Functions

In VAX/VMS, a task image can enable itself to receive an Asynchronous System Trap (AST) if the user types an unsolicited character or a CTRL/C from the terminal. When the AST occurs, the task image can respond to the unsolicited character or CTRL/C.

The RSX-11 function codes IO.ATT!TF.AST and IO.ATA are equivalent. When an RSX-11 task image executing in VAX-11 RSX issues either of these codes, VAX-11 RSX issues a request to the VAX/VMS terminal driver to enable the task image for an unsolicited character or CTRL/C AST, depending on the values of parameters P1 and P3, respectively.

VAX-11 RSX also supports the TF.XCC subfunction bit. The function codes IO.ATT!TF.AST!TF.XCC and IO.ATA!TF.XCC are equivalent. When TF.XCC is specified in this way and if the user types a CTRL/C, the CTRL/C characte is ignored. Neither a CTRL/C nor an unsolicited input is declared.

8.6.1.2 IO.ATT!TF.ESQ Function

In VAX/VMS, certain features that are characteristic of a terminal line are set by issuing a Set Terminal Mode request (IO$_SETMODE) to the driver. Terminal characteristics cannot be altered for the duration of an I/O request by specifying a modifier to the request; nor can they be modified as a functioı of terminal allocation. The ability to recognize escape sequences on a termina line is a characteristic of the terminal and must be set with IO$_SETMODE or a SET TERMINAL command.

In RSX-11, the subfunction bit TF.ESQ is used with either of the attach function codes (IO.ATT or IO.ATA) to indicate that the task image recognizes any escape sequences generated at the designated terminal. When VAX-11 RSX receives an I/O request containing the TF.ESQ subfunction from an RSX-11 task image, it ignores that subfunction bit. The terminal characteristics remaiı unaltered.

To enable escape sequences, an RSX-11 task image should issue a Set Multiple Characteristics request (SF.SMC).

8.6.2 IO.DET Function

When an RSX-11 task image uses the IO.DET function for a terminal, VAX-11 RSX performs no operation to alter the attached/detached status of the terminal, as described in 8.1.1. If the terminal was attached for unsolicited character or CTRL/C AST reception, the AST enable will be canceled. VAX-11 RSX also issues a request to the terminal driver to cancel CTRL/O on the next operation to the terminal if that operation is a write. The VAX/VMS terminal driver subfunction modifier to cancel CTRL/O is IO$M_CANCTRLO. The RSX-11 terminal driver also forces a Cancel CTRL/O (TF.CCO) on a write operation that follows a detach operation.

An IO.DET function issued for TI, CO, or CL is ignored if the specified device is not a terminal.

8.6.3 IO.KIL Function

An IO.KIL function issued for TI, CO, or CL is ignored if the specified device is not a terminal.

.6.4 IO.RLB, IO.RAL, IO.RNE, IO.RST, and IO.RTT Functions

The function codes IO.RLB, IO.RAL, IO.RNE, and IO.RST all allow a task image to read a logical block from a terminal. When VAX–11 RSX receives a Read Logical Block request from an RSX–11 task image, it issues an IO$_READLBLK request on behalf of the task image. There is a direct correspondence between IO.RLB and IO$_READLBLK. The function codes IO.RAL, IO.RNE, and IO.RST are the equivalents of the logical OR of IO.RLB and a subfunction bit. The following sections describe VAX–11 RSX handling of subfunction bits used with Read Logical Block requests.

8.6.4.1 IO.RLB!TF.RAL and IO.RAL

In VAX/VMS, the default terminal driver operation on a read request is to intercept and interpret control characters, for example, TAB, CTRL/B, CTRL/R, CTRL/U, and DELETE. However, a VAX/VMS native image has two options for restricting the interception of control characters by the driver.

- It can specify the subfunction modifier IO$M_NOFILTR on a read function (either IO$_READLBLK, IO$_READVBLK, or IO$_READPROMPT) to prevent the driver from intercepting CTRL/U, CTRL/R, or DELETE.
- It can issue a Read Physical Block (IO$_READPBLK) request to prevent the driver from interpreting any characters.

When VAX–11 RSX receives an IO.RLB!TF.RAL or IO.RAL request from an RSX–11 task image, it issues a request specifying IO$_READLBLK!IO$M_NOFILTR for the task image.

The VAX/VMS equivalent of the RSX–11 Read-Passing-All-Data function is Read Physical Block (IO$_READPBLK). IO$_READPBLK corresponds directly to the RSX–11 function code IO.RPB. An RSX–11 task image executing a Read-Passing-All-Data function under VAX/VMS must be modified to issue an IO.RPB. A task image issuing IO.RPB under the RSX–11 operating system runs without receiving an error; that is, IO.RPB is a legal function.

NOTE: **In RSX–11, an IO.RPB request is equivalent to an IO.RLB request with a subfunction bit set. IO.RAL or IO.RPB work on both VAX/VMS and RSX–11 systems.**

VAX/VMS requires the task image to have the appropriate privilege to read a physical block.

8.6.4.2 IO.RLB!TF.RNE and IO.RNE Functions

The RSX–11 function codes IO.RLB!TF.RNE and IO.RNE are equivalent. Either one corresponds directly to the VAX/VMS function code IO$_READLBLK or IO$_READVBLK with a no echo function modifier (IO$M_NOECHO).

8.6.4.3 IO.RLB!TF.RST and IO.RST Functions

The RSX–11 function codes IO.RLB!TF.RST and IO.RST are equivalent. Either one corresponds directly to the VAX/VMS function code IO$_READLBLK with a function modifier of IO$M_TRMNOECHO and a record-termination parameter (P4) of 0. IO$M_TRMNOECHO prevents echoing of the line terminator. A record termination parameter of 0 causes all characters with a value less than an ASCII space to be terminators except form feed, vertical tab, backspace, and horizontal tab.

8.6.4.4 IO.RLB!TF.RTT and IO.RTT Functions

The RSX–11 function codes IO.RLB!TF.RTT and IO.RTT are equivalent. Eithe: one corresponds directly to the VAX/VMS function code IO$_READVBLK with a function modifier of IO$M_TRMNOECHO and IO$M_NOFILTR and a record-termination parameter P4.

8.6.5 IO.RPR Function

The IO.RPR function code corresponds directly to the VAX/VMS IO$_READPROMPT function code. However, the RSX–11 P6 parameter (vertical control character) is ignored. VAX–11 RSX handling of the subfunction bits TF.RAL, TF.RNE, and TF.RST with IO.RPR is exactly the same as it is for IO.RLB. VAX–11 RSX does not support use of the subfunction bit TF.BIN. VAX–11 RSX also ignores the subfunction bit TF.XOF if it is specified.

If an IO.RPR function is issued for TI, CO, or CL and the device is not a terminal, the prompt is ignored.

8.6.5.1 IO.RPR!TF.XOF Function

Under VAX/VMS, certain features that are characteristic of a terminal line are set by issuing a Set-Terminal-Mode request (IO$_SETMODE) to the driver. A subfunction modifier indicates the characteristic to be changed. Terminal characteristics cannot be altered for the duration of an I/O request by specifying a modifier to the request; nor can they be modified as a function of terminal allocation. The ability to control XON/XOFF on a terminal line is a characteristic of the terminal and must be set using IO$_SETMODE.

In RSX–11, the subfunction bit TF.XOF is used with IO.RPR to control XON /XOFF at the designated terminal. When VAX–11 RSX receives an I/O request containing the TF.XOF subfunction from an RSX–11 task image, it ignores that subfunction bit. The terminal characteristics remain unaltered.

To control XON/XOFF, an RSX–11 task image should issue a Set Multiple Characteristics request (SF.SMC). VAX/VMS allows a task to force XON; however, XOFF cannot be forced.

8.6.6 IO.RPB Function

See the discussion of IO.RAL in 8.6.4.1.

8.6.7 IO.RVB Function

The IO.RVB function code corresponds directly to the VAX/VMS IO$_READVBLK. No subfunction bits are supported in RSX–11 for IO.RVB.

8.6.8 IO.WLB, IO.CCO, and IO.WBT Functions

The function codes IO.WLB, IO.CCO, and IO.WBT all allow a task image to write a logical block to a terminal. When VAX–11 RSX receives IO.WLB request from an RSX–11 task image, it issues an IO$_WRITELBLK request. There is a direct correspondence between IO.WLB and IO$_WRITELBLK. The RSX–11 function codes IO.CCO, and IO.WBT are the equivalents of the logical OR of IO.WLB and a subfunction bit. The sections that follow describe VAX–11 RSX handling of subfunction bits on Write Logical Block requests.

8.6.8.1 IO.WLB!TF.CCO and IO.CCO Functions

The RSX–11 function codes IO.WLB!TF.CCO and IO.CCO are equivalent. Either one corresponds directly to the VAX/VMS IO$_WRITELBLK function with a function modifier of IO$M_CANCTRLO.

8.6.8.2 IO.WLB!WBT and IO.WBT Functions

Under VAX/VMS, the Write-Break-Through function is implemented using the Broadcast system service. As a result, neither of the RSX–11 function codes IO.WLB!WBT or IO.WBT corresponds directly to a VAX/VMS driver function. When an RSX–11 task image requests Write-Break-Through, VAX–11 RSX issues a IO$_WRITELBLK function to the driver. A normal Write Logical Block function occurs.

8.6.9 IO.WVB Function

The RSX–11 function code IO.WVB corresponds directly to the VAX/VMS function code IO$_WRITEVBLK. VAX–11 RSX handles the subfunction bits allowed with IO.WVB in the same manner as it handles the subfunction bits for IO.WLB. The resulting I/O operation is a Write Virtual Block, however.

8.6.9.1 IO.WLB!TF.WAL, IO.WAL, and IO.CCO!TF.WAL Functions

The RSX–11 function codes IO.WLB!TF.WAL and IO.WAL are equivalent. The RSX–11 function IO.CCO!TF.WAL adds the Cancel CTRL/O subfunction to an IO.WAL request. When an RSX–11 task image issues a Write-All-Data request, VAX–11 RSX issues an IO$_WRITELBLK!IO$M_NOFORMAT request to cause the data block to be transferred without interpretation to the specified buffer.

The appropriate VAX/VMS privilege is required to execute the IO.WAL function successfully.

8.6.10 IO.WPB Function

The RSX–11 function code IO.WPB corresponds directly to the VAX/VMS function code IO$_WRITEPBLK. No subfunction bits are applicable.

VAX/VMS requires a task image to have the appropriate privilege to write a physical block. An RSX–11 task image must have this privilege to issue an IO.WPB request successfully.

8.6.11 IO.GTS Function

VAX–11 RSX has no system generation options that control the features included in the terminal driver.

When an RSX–11 task image issues an IO.GTS request, VAX–11 RSX returns a four-word buffer of information describing the VAX/VMS terminal driver features. Because these features cannot be altered, the same information is always returned. Table 8–11 lists the terminal support information returned under VAX–11 RSX.

That information includes all the features that can be returned under RSX–11 with the following exceptions, which are always zero:

Word 0	Bit 1	F1.BTW	Write-break-through
	Bit 2	F1.BUF	Checkpointing during terminal input
	Bit 14	F1.UTP	Input characters buffered in task's address space
	Bit 15	F1.VBF	Variable-length terminal buffers

Table 8–11 Information Returned by Get Terminal Support (IO.GTS)

Word	Bit	Mnemonic	Meaning
0	0	F1.ACR	Automatic CRLF on long lines
0	3	F1.UIA	Unsolicited-character-input AST
0	4	F1.CCO	Cancel CTRL/O before writing
0	5	F1.ESQ	Recognize escape sequences in solicited input
0	6	F1.HLD	Hold screen mode
0	7	F1.LWC	Lowercase-to-uppercase conversion
0	8	F1.RNE	Read with no echo
0	9	F1.RPR	Read after prompting
0	10	F1.RST	Read with special terminators
0	11	F1.RUB	CRT rubout
0	12	F1.SYN	CTRL/R terminal synchronization
0	13	F1.TRW	Read all and write all
1	0	F2.SCH	Set characteristics QIO (SF.SMC)
1	1	F2.GCH	Get characteristics QIO (SF.GMC)
1	7	F2.FDX	Full duplex terminal driver
2			Not used in RSX–11
3			Not used in RSX–11

NOTE: **An IO.GTS function issued for TI, CO, or CL returns no information if the specified device is not a terminal.**

8.6.12 SF.GMC Function

When an RSX–11 task image issues an SF.GMC request, VAX–11 RSX issues an IO$SENSEMODE request. Table 8–12 lists the terminal characteristics that can be returned for SF.GMC requests. The RSX–11 characteristic TC.PRI is never returned by VAX–11 RSX, because VAX/VMS does not incorporate the concept of a privileged terminal.

An SF.GMC function issued for TI, CO, or CL performs no operation if the specified device is not a terminal.

3.6.13 SF.SMC Function

When an RSX-11 task image issues an SF.SMC function, VAX-11 RSX issues an IO$_SETMODE request. Table 8-12 shows the correspondence between RSX-11 terminal characteristics bit names and VAX/VMS subfunction modifiers used with the function code IO$_SETMODE.

The set terminal type subfunction (TC.TTP) supports the following terminals: LA12, LA36, LA100, LA120, LA34, LA38, LQP02, VT05, VT52, VT55, VT100, VT101, VT102, VT105, VT125, VT131, VT132, and the VT200 series. Other terminals considered "unknown," include ASR33, ASR35, KSR33, LA30S, LA30P, VT50, VT61, and LA180S.

The TC.TBF characteristic for the SF.SMC request clears the type-ahead buffer.

An SF.SMC function issued for TI, CO, or CL performs no operation if the specified device is not a terminal.

Table 8–12 Terminal Characteristics for SF.GMC and SF.SMC Requests

RSX–11 Bit Name	VAX/VMS Code	Meaning
TC.ABD	TT2$M_AUTOBAUD	Autobaud detection enabled
TC.ACR	TT$M_WRAP	Wraparound mode
TC.ANS	TT2$M_ANSICRT	ANSI CRT
TC.ASP	(not supported)	Remote line answer speed
TC.AVO	TT2$M_AVO	Advanced video VT1xx
TC.BIN	TT$M_PASSALL	Binary output
TC.BLK	TT2$M_BLOCK	Block mode terminal
TC.CTS	(not supported)	Suspend/resume I/O
TC.DEC	TT2$M_DECCRT	DEC CRT (VT100 compatible)
TC.DLU	(not supported)	Dialup line
TC.EDT	TT2$M_EDIT	Local edit terminal
TC.ESQ	TT$M_ESCAPE	Escape sequence recognition
TC.FDX	(not supported)	Full-duplex mode
TC.HFF	TT$M_MECHFORM	Hardware formfeed capability
TC.HFL	[1,2]	Fill on carriage return
TC.HHT	TT$M_MECHTAB	Hardware horizontal tab capability
TC.HLD	TT$M_HOLDSCREEN	Hold screen mode
TC.ISL	(not supported)	Subline on interface
TC.LPP	[2]	Page length

[1] The legal range of the carriage return fill value for VAX/VMS is different from the legal range under RSX–11. The latter range is enforced by VAX–11 RSX.
[2] This RSX–11 characteristic does not directly correspond to a VAX/VMS characteristic bit. Instead, VAX–11 RSX uses device-independent information fields for the VAX/VMS QIO functions IO$_SENSEMODE or IO$_SETMODE.

Table 8–12 (Cont.) Terminal Characteristics for SF.GMC and SF.SMC Requests

RSX–11 Bit Name	VAX/VMS Code	Meaning
TC.NEC	TT$M_NOECHO	Echo suppressed
TC.PRI	[3]	Terminal is privileged
TC.RAT	(not supported)	Type-ahead buffer
TC.RGS	TT2$M_REGIS	ReGIS graphics
TC.RSP	(not supported)	Receiver speed
TC.SCP	TT$M_SCOPE	Terminal is a scope
TC.SLV	TT$M_NOTYPEAHD	Terminal is a slave
TC.SMR	TT$M_LOWER	Uppercase conversion disabled
TC.TBF	[2]	Preview/flush type-ahead buffer
TC.TBS	(not supported)	Type-ahead buffer size
TC.TBM	(not supported)	Type-ahead buffer mode
TC.TTP	[2,4]	Terminal type
TC.VFL	[2,5]	Fill on line feed
TC.WID	[2]	Terminal width
TC.XSP	(not supported)	Transmitter speed
TC.8BC	(not supported)	Pass eight bits

[2] This RSX–11 characteristic does not directly correspond to a VAX/VMS characteristic bit. Instead, VAX–11 RSX uses device-independent information fields for the VAX/VMS QIO functions IO$_SENSEMODE or IO$_SETMODE.

[3] The bit TC.PRI cannot be set under VAX/VMS. When read, it will always return the value 0.

[4] In setting this characteristic, only terminal types supported by both VAX/VMS and RSX–11 are legal values.

[5] Under RSX–11, the bit TC.VFL is a Boolean value. The value 0 indicates no fill characters and the value 1 indicates four fill characters. Under VAX/VMS, the equivalent value is a byte count of fill characters (0–255). The value of this characteristic is returned as 0 only if the VAX/VMS characteristic is 0. If the VAX/VMS characteristic is not 0, the value is returned as 1 (the actual VAX/VMS fill count need not be 4). When this characteristic is being set, the fill count will be set to 4.

8.6.14 Terminal Read Status Returns

The contents of an I/O status block used for terminal requests is the same as that used for all QIO operations except for terminal read operations. For terminal read operations, the high-order byte of the first word contains a code indicating the character or sequence that terminated the read operation. Any one of the following codes can be returned:

Code	Meaning
IS.CR	Read terminated by RETURN
IS.ESC	Read terminated by ALTMODE
IS.ESQ	Read terminated by an escape sequence
–	Other terminator character
0	Read terminated by full buffer

When using VAX/VMS terminal function codes and parameters, keep in mind the following points:

- VAX/VMS terminals can be spooled.
- See Section 2 for a discussion of the requirements for issuing IO.WLB and IO.WVB requests to a spooled device.
- If an RSX–11 task image issues a Get LUN Information directive for a spooled device, the information returned is as follows:

Word 0,1	Name of spooled device (for example, LP0)
Word 2	Device-independent characteristics for intermediate device (disk), plus the output spooled device bit
Word 3	Spooled device (line printer) line size
Word 4	High word of VAX/VMS device-dependent longword
Word 5	Intermediate device buffer size (the size is usually 512 (10))

- TI, CO, and CL map to VAX/VMS process-permanent files as follows:

RSX–11 Pseudodevice	VAX/VMS Process-Permanent Files
TI	SYS$INPUT for read operations
	and for write operations SYS$OUTPUT
CO	SYS$COMMAND
CL	SYS$ERROR

- Process-permanent files that do not map to terminals are controlled by VAX–11 RMS. VAX/VMS, therefore, limits the I/O function codes that can be used to access these files to read and write functions only. All subfunction bits are ignored. Functions other than read and write are illegal and result in the return of the I/O status code IE.IFC (illegal function for this device).
- For RSX–11 task images, user-created process-permanent files appear as record-oriented terminal devices.
- When process-permanent files map to terminals, QIOs can be issued.
- The device characteristics for TI, CO, and CL are as follows if they are not terminals:

```
Unit record device
Terminal
132-byte buffer
Carriage control
No lowercase
```

8.7 Card Reader Driver

Table 8–13 shows the correspondence between RSX–11 card reader functions and VAX/VMS function codes or resultant actions.

Table 8–13 Card Reader Function Code Correspondence

Function	RSX–11 Code	VAX/VMS Code or Action
Attach Device	IO.ATT	No operation
Detach Device	IO.DET	No operation
Cancel I/O Request	IO.KIL	Cancel I/O on Channel system service
Read Virtual Block	IO.RVB	IO$_READVBLK
Read Logical Block	IO.RLB	IO$_READLBLK
Read Logical Block	IO.RBD	IO$_READLBLK!IO$M_BINARY

The two function-dependent parameters (P1 and P2) for RSX–11 card reader functions correspond directly to P1 and P2 of VAX/VMS card reader functions.

8.8 Null Device

VAX–11 RSX supports the use of a null device by RSX–11 task images. As under RSX–11, a read request to the null device results in an End-Of-File status return (IE.EOF), and a write request results in Success status return (IE.SUC).

I/O to the null device is treated like I/O to an unsupported device, as described in 8.9.

8.9 Unsupported Device

In most cases, the RSX–11 QIO function codes and subfunction modifiers are not identical to their VAX/VMS equivalents. Therefore, the VAX–11 RSX Applications Migration Executive (AME) translates RSX–11 QIO$ directive calls to appropriate VAX/VMS QIO system service calls. To do this, the AME must be able to translate for each device type.

The VAX–11 RSX AME supports the device types that are most common to both the RSX–11 and the VAX/VMS operating systems (for example, disk, tape, line printer, terminal, card reader, and file ACP QIO) and translates most RSX–11 QIO$ directive calls. The AME recognizes the "unsupported device" category for an RSX–11 task image that may perform I/O to a different type of a device (for example, a VAX/VMS mailbox).

RSX–11 QIO$ directives to unsupported devices are translated to VAX/VMS QIO system service calls by the AME. However, because the AME has no detailed knowledge of an unsupported device, the translation that it performs is minimal. No subfunction modifiers or device-dependent function codes may be used. Subfunction modifiers given with the RSX–11 QIO function code are ignored. Table 8–14 shows the correspondence between RSX–11 and VAX/VMS function codes.

Table 8–14 RSX–11 and VAX/VMS Function Code Correspondence

Function	RSX–11	VAX/VMS
Attach Device	IO.ATT	No operation
Detach Device	IO.DET	No operation
Cancel I/O Request	IO.KIL	Cancel I/O on Channel system service
Read Virtual Block	IO.RVB	IO$_READVBLK
Read Logical Block	IO.RLB	IO$_READLBLK
Write Virtual Block	IO.WVB	IO$_WRITEVBLK
Write Logical Block	IO.WLB	IO$_WRITELBLK

8.10 Disk and Magnetic Tape ACPs

I/O operations involving file-structured devices (disk and magnetic tape) often require ACP intervention. Normally, RSX–11 task images perform I/O using RMS–11 or FCS; they do not issue Queue I/O Request directives directly to an ACP. Any ACP intervention needed is requested by FCS and occurs transparently. However, task images can request ACP functions directly by issuing a QIO directive and specifying an ACP function code.

The information in this section is relevant only to RSX–11 task images that issue ACP functions directly, for example, task images that create files and enter file names. Other RSX–11 task images running under VAX/VMS can rely on FCS to request appropriate RSX–11 ACP functions during task image execution.

VAX/VMS ACP functions are expressed with six function codes and three function modifiers. The six function codes follow:

1. IO$_CREATE—Create file
2. IO$_ACCESS—Access file
3. IO$_DEACCESS—Deaccess file
4. IO$_MODIFY—Modify file
5. IO$_DELETE—Delete file
6. IO$_ACPCONTROL—ACP control

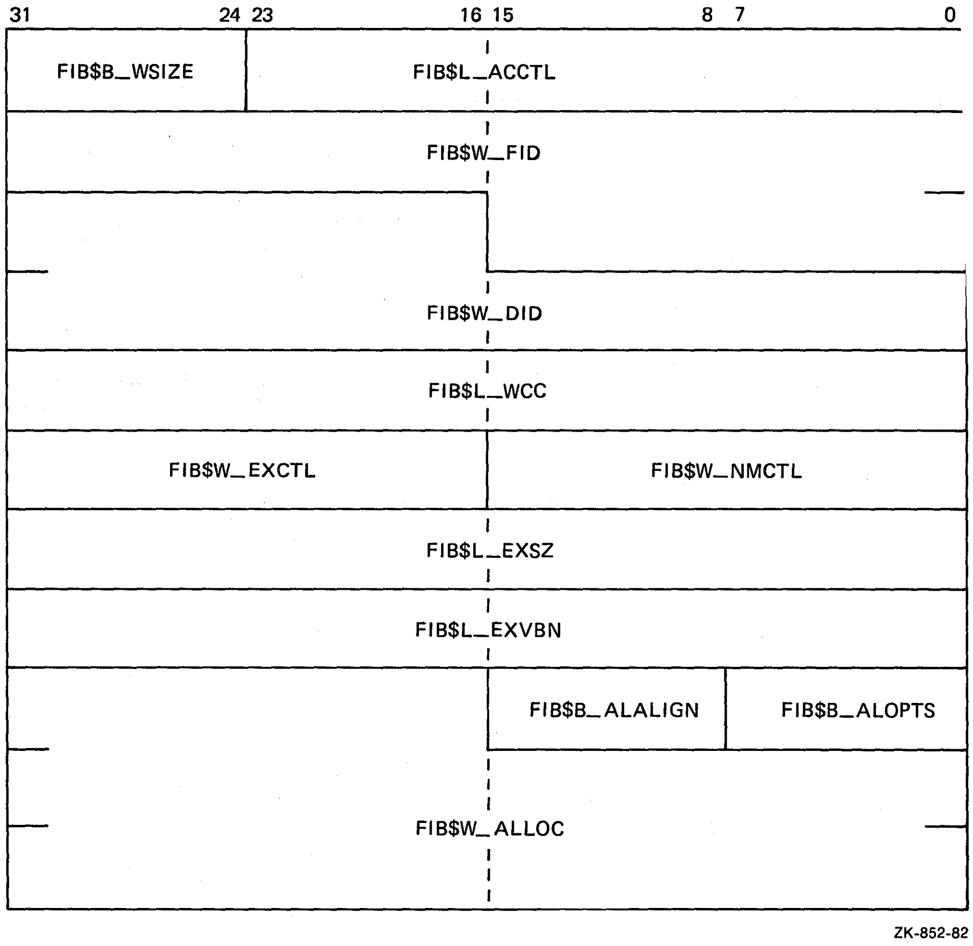
The three function modifiers, which can be applied to the create, access, and delete functions, follow:

1. IO$M_ACCESS—Open file on user's channel
2. IO$M_CREATE—Create a file identification
3. IO$M_DELETE—Delete file

By using a function code and a function modifier together, a task image can request multiple ACP operations in one I/O request. For example, IO$_CREATE!IO$M_ACCESS requests the ACP to create a file and to access the file on the specified channel. IO$_DELETE!IO$M_DELETE causes a file's directory entry and file header to be deleted; that is, the file is deleted. IO$_DELETE with no function modifier causes the file's directory entry to be deleted.

In addition to function codes and modifiers, VAX/VMS ACPs use a file identification block (FIB) for communication between the requester and the ACP. The function-dependent parameter P1 for all ACP requests is the address of a descriptor for the associated FIB. The FIB contains much of the information passed to an ACP by an RSX-11 task image in P1 through P6. Figure 8-2 illustrates a FIB.

Figure 8-2 File Identification Block Format



RSX-11 ACP functions are expressed with the following function codes:

- IO.CRE—Create file
- IO.ACR—Access for read
- IO.ACW—Access for write
- IO.ACE—Access for extend

- IO.EXT—Extend file
- IO.WAT—Write attributes
- IO.RAT—Read attributes
- IO.DAC—Deaccess file
- IO.DEL—Delete file
- IO.FNA—Find file name
- IO.RNA—Remove file name
- IO.ENA—Enter file name
- IO.APC—ACP control

When an RSX–11 task image issues an ACP request under VAX/VMS, VAX–11 RSX issues a Queue I/O Request system service to the ACP. It obtains the data to fill in the FIB and function-dependent parameters for the request from two sources:

- Function-dependent parameters supplied by the task image in the QIO$ directive
- Data structures pointed to by function-dependent parameters, for example, the file name block

Once the requested function is performed, VAX–11 RSX fills in the RSX–11 task image's data structures with the same information that is returned to the task image when it executes under the RSX–11 operating system.

8.10.1 ACP Function Parameter Correspondence

Table 8–15 identifies the relationship of RSX–11 function-dependent parameters to VAX/VMS function-dependent parameters and FIB fields.

Table 8–15 ACP Parameter Correspondence

Parameter Function	RSX–11 Pn	VAX/VMS Equivalent
File identification pointer	P1 (pointer)	FIB$W_FID (value)
Attribute list pointer	P2	P5 (reformatted)
Extend control	P3 (high byte)	FIB$W_EXCTL
Delta size in blocks	P3 (low byte) and P4	FIB$L_EXVBN for truncate only; FIB$L_EXSZ for extend
Window size	P5 (low byte)	FIB$B_WSIZE
Access control	P5 (high byte)	FIB$L_ACCTL
File name block pointer	P6	P2 (name string) and P4 (result string)

8.10.1.1 IO.CRE Function

Equivalent Function Code: IO$_CREATE!IO$M_CREATE

Notes on Standard IO.CRE Function

1. If the extend size is supplied in the low-order byte of P3 and in P4, it is stored in FIB$L_EXSZ.
2. The high-order byte of P3 (extend control) is used to set bits in FIB$W_EXCTL:

```
FIB$V_EXTEND  = EX.ENA
FIB$V_ALCON   = EX.AC1
FIB$V_ALCONB  = EX.AC2
FIB$V_FILCON  = EX.FCO
FIB$V_ALDEF   = EX.ADF
```

3. The file identification is copied from FIB$W_FID and returned in the address pointed to by P1 (FID pointer).
4. Information in the VAX/VMS attribute list is derived from the RSX-11 attribute list, if one is supplied.
5. The extend size in blocks is returned in bytes 1, 2, and 3 of the I/O status block.

When a file is created under VAX/VMS from VAX RMS, an Access Control List (ACL) is added to the file under one of the following conditions:

- If a previous version of the file exists and has an ACL, the ACL is propagated to the new file.
- If the new file does not have an ACL and no previous version of the file exists, then the default ACL for the directory is propagated to the new file.

If the default ACL is not available from the directory, an ACL is not added to the file.

Notes on Nonstandard IO.CRE Function

Directory context is required at the time the file is created to ensure correct ACL propagation to the new file. This is accomplished by using the version of the IO$_CREATE function that creates the file and enters it into the specified directory. The standard RSX-11 IO.CRE function described by the previous notes does not provide the required directory context. A nonstandard version of the IO.CRE function is available under VAX-11 RSX. The nonstandard version provides the necessary directory context required for correct ACL propagation. It creates the file and enters it into the specified directory in one operation.

You should convert any applications that create files on disk volumes so that the applications use the the nonstandard version. Programs that are not modified will create files that do not have ACLs. However, this recommendation does not apply either to programs that create temporary files or to programs that create files but never enter those files into a directory. In either situation, you may use the variant of the IO.CRE function provided by VAX-11 RSX to create and enter the file. The application may then explicitly remove the file from the specified directory.

The nonstandard version of the IO.CRE function is supported only under VAX-11 RSX. If a program issues the function under RSX-11, the operating system returns an error status. You can migrate FCS-linked programs to RSX-11, because FCS uses the nonstandard version of IO.CRE only when running under VAX–11 RSX. Under RSX–11, FCS creates and enters the file in two separate operations.

The following list provides notes on some features of the nonstandard version of the IO.CRE function:

- If you want the newly created file immediately entered into a directory (thus allowing propagation of an ACL), you must specify a file name block in P6. If you do not specify a file name block, IO.CRE will create the file without entering it in a directory and without an ACL.
- The directory identification is copied from the file name block into FIB$W_DID.
- The file name string supplied in the request is constructed from the Radix–50 file name in the file name block.
- The file name returned is the resultant string returned by the Queue I/O Request system service. The file name string is converted back to Radix–50 and returned to the file name block.

8.10.1.2 IO.DEL with EX.ENA=0

Equivalent Function Code: IO$_DELETE!IO$M_DELETE

Note

The file identification pointed to by P1 is copied into FIB$W_FID.

8.10.1.3 IO.DEL with EX.ENA=1

Equivalent Function Code: IO$_MODIFY

Notes

1. The file identification pointed to by P1 is copied into FIB$W_FID.
2. FIB$V_TRUNC is set in field FIB$W_EXCTL.
3. The extend size supplied in the low byte of P3 and in P4 is incremented by 1 and stored in FIB$L_EXVBN.
4. The file roundup in blocks is returned in bytes 2 and 3 of the I/O status block. **File roundup** is the number of blocks added to the specified file size to reach the next cluster boundary.

8.10.1.4 IO.ACR Function

Equivalent Function Code: IO$_ACCESS!IO$M_ACCESS

Notes

1. The file identification pointed to by P1 is copied into FIB$W_FID.
2. The high-order byte of P5 (access control) is used to set bits in FIB$L_ACCTL:

```
FIB$V_NOWRITE  = AC.LCK
FIB$V_REWIND   = AC.RWD
FIB$V_CURPOS   = AC.POS
FIB$V_UPDATE   = AC.UPD
```

3. The window size provided by the low-order byte of P5 is stored in FIB$B_WSIZE.

4 Information in the VAX/VMS attribute list is derived from the RSX–11 attribute list, if one is supplied.

8.10.1.5 IO.ACW and IO.ACE Functions

Equivalent Function Code: IO$_ACCESS!IO$M_ACCESS

Notes

1 The file identification pointed to by P1 is copied into FIB$W_FID.

2 The high-order byte of P5 (access control) is used to set bits in FIB$L_ACCTL:

```
FIB$V_DLOCK   = AC.DLK
FIB$V_NOWRITE = AC.LCK
FIB$V_REWIND  = AC.RWD
FIB$V_CURPOS  = AC.POS
FIB$V_UPDATE  = AC.UPD
```

In addition, VAX–11 RSX sets FIB$V_WRITE.

3 The window size provided by the low-order byte of P5 is stored in FIB$I WSIZE.

4 Information in the VAX/VMS attribute list is derived from the RSX–11 attribute list, if one is supplied.

8.10.1.6 IO.DAC Function

Equivalent Function Code: IO$_DEACCESS

Notes

1 The file identification pointed to by P1 is copied into FIB$W_FID.

2 Information in the VAX/VMS attribute list is derived from the RSX–11 attribute list, if one is supplied. An empty attribute list is treated by VAX/VMS as if no attribute list had been specified.

8.10.1.7 IO.EXT Function

Equivalent Function Code: IO$_MODIFY

Notes

1 The file identification pointed to by P1 is copied into FIB$W_FID.

2 The high-order byte of P3 (extend control) is used to set bits in FIB$W_EXCTL:

```
FIB$V_EXTEND  = EX.ENA
FIB$V_ALCON   = EX.AC1
FIB$V_ALCONB  = EX.AC2
FIB$V_FILCON  = EX.FCO
FIB$V_ALDEF   = EX.ADF
```

3 The extend size supplied in the low-order byte of P3 and in P4 is stored in FIB$L_EXSZ.

4 The amount by which the file is extended is returned in bytes 1, 2, and 3 of the I/O status block.

8.10.1.8 IO.WAT Function

Equivalent Function Code: IO$_MODIFY

Notes

1 The file identification pointed to by P1 is copied into FIB$W_FID.

2 Information in the VAX/VMS attribute list is derived from the RSX-11 attribute list, if one is supplied.

8.10.1.9 IO.RAT Function

Equivalent Function Code: IO$_ACCESS

Notes

1 The file identification pointed to by P1 is copied into FIB$W_FID.

2 Information in the VAX/VMS attribute list is derived from the RSX-11 attribute list, if one is supplied.

8.10.1.10 IO.FNA Function

Equivalent Function Code: IO$_ACCESS

Notes

1 The file identification is copied from FIB$W_FID and returned in the address pointed to by P1.

2 The directory identification is copied from the file name block into FIB$W_DID.

3 The file name string supplied in the request is constructed from the Radix-50 file name in the file name block.

4 If the bit NB.WLV is set in N.STAT of the file name block, a resultant string is constructed from the Radix-50 name and type. The version number is stored in N.FID+4 of the name block, and is supplied as input to the IO$_ACCESS call.

If NB.WLV is not set, a resultant string of zero length is supplied.

5 The file name string returned is the resultant string returned by the Queue I/O Request system service. It is converted back to Radix-50 and returned to the file name block.

6 Control bits in field N.STAT of the file name block are used to set bits of FIB$W_NMCTL:

```
FIB$V_ALLNAM  = NB.SNM
FIB$V_ALLTYP  = NB.STP
FIB$V_ALLVER  = NB.SVR
FIB$V_WILD    = NB.SNM!NB.STP!NB.SVR
```

7 The file name block field N.NEXT is used to set FIB$L_WCC. The resulting value of FIB$L_WCC is returned in N.NEXT.

8 Any file name that has more than nine characters, includes extended characters (for example, a dollar sign ($) or underscore (_)), or has a file type that has more than three characters is ignored for wildcard access.

8.10.1.11 IO.RNA Function

Equivalent Function Code: IO$_DELETE

Notes

1 The file identification is copied from FIB$W_FID and returned in the address pointed to by P1.

2 The directory identification is copied from the file name block into FIB$W_DID.

3 If the bit NB.WLV is set in N.STAT of the file name block, a resultant string is constructed from the Radix–50 name and type. The version number is stored in N.FID+4 of the name block, and is supplied as input to the IO$_ACCESS call.

If NB.WLV is not set, a resultant string of zero length is supplied.

4 The file name string supplied in the request is constructed from the Radix–50 file name in the file name block.

5 The file name returned is the resultant string returned by the Queue I/O Request system service. It is converted back to Radix–50 and returned to the file name block.

6 Control bits in field N.STAT of the file name block are used to set bits of FIB$W_NMCTL:

```
FIB$V_ALLNAM  = NB.SNM
FIB$V_ALLTYP  = NB.STP
FIB$V_ALLVER  = NB.SVR
FIB$V_WILD    = NB.SNM!NB.STP!NB.SVR
```

7 The file name block field N.NEXT is used to set FIB$L_WCC. The resulting value of FIB$L_WCC is returned in N.NEXT.

8.10.1.12 IO.ENA Function

Equivalent Function Code: IO$_CREATE

Notes

1 The file identification is copied from the file name block into FIB$W_FID.

2 The directory identification is copied from the file name block into FIB$W_DID.

3 The file name string supplied in the request is constructed from the Radix–50 file name in the file name block.

4 The file name returned is the resultant string returned by the Queue I/O Request system service. It is converted back to Radix–50 and returned to the file name block.

5 The file is entered into the specified directory if it was created with IO.CRE and if the directory context was provided using P6 for that function. Consequently, there is no need to enter the file specifically with the IO.ENA function. This only applies to VAX–11 RSX; the IO.CRE function does not allow directory context to be specified under RSX–11M or RSX–11M–PLUS.

8.10.1.13 IO.APC Function

Equivalent Function Code: IO$_ACPCONTROL

Notes

1 P3 contains the subfunction identification. The low-order byte of P3 is zero-extended and stored at FIB$W_CNTRLFUNC, which overlays FIB$W_EXCTL. The RSX-11 ACP subfunction codes have direct equivalents in VAX/VMS, as follows.

RSX–11 Subfunction	VAX/VMS Subfunction
FF.NV	FIB$C_NEXTVOL
FF.POE	FIB$C_POSEND
FF.RWD	FIB$C_REWINDVOL
FF.RWF	FIB$C_REWINDFIL
FF.SPC	FIB$C_SPACE

2 For the FF.SPC subfunction, P4 is sign-extended and stored at FIB$L_CNTRLVAL, which overlays FIB$L_EXSZ. A negative value for P4 specifies the number of blocks to space backward. A positive value indicates the number of blocks to space forward.

PART IV VAX–11 RSX Messages

9 System Messages

This section lists the error and information messages produced by VAX–11 RSX. System messages are accompanied by explanations and suggested actions for recovering from errors. Refer to 9.1 for messages that are produced by VAX–11 RSX and are in a VAX/VMS compatible format. Refer to 9.2 for Indirect Command Processor messages that are produced by VAX–11 RSX but are in an RSX–11 compatible format.

9.1 VAX–11 RSX Error Message Processing

The system displays or prints messages in both uppercase and lowercase letters (when the output device is capable of displaying or printing lowercase letters).

In this section the fixed and variable portions of the text of a message are distinguished by enclosing the variable portions in apostrophes; the distinction is obvious when a message is displayed.

Messages displayed by VAX–11 RSX have the following format:

```
%FACILITY-L-IDENT, text
[-FACILITY-L-IDENT, text]
```

where

% FACILITY or -FACILITY

is a VAX–11 RSX facility or component name. A percent sign (%) prefixes the first message issued, and a hyphen (-) prefixes each subsequent message. The VAX–11 RSX facilities are listed in 9.1.1.

L

is a severity level indicator with one of the following values:

Code	Meaning
S	Success
I	Information
W	Warning
E	Error
F	Fatal, or severe error

Severity levels are described in detail in 9.1.2.

IDENT

is an abbreviation of the message text; the message descriptions in 9.1.4 are alphabetized by this abbreviation.

TEXT

is the explanation of the message.

An example of the first message that might be issued is as follows:

```
%MCR-W-ABVERB, ambiguous command verb - supply more characters
```

where 'MCR' is the facility name, 'W' is the severity level, 'ABVERB' is the ident and 'ambiguous command verb - supply more characters' is the messaε text.

An example of the second type of message is as follows:

```
-RSX-F-TBIT, compatibility mode T-bit
```

where 'RSX' is the facility name, 'F' is the severity level, 'TBIT' is the ident, and 'compatibility mode T-bit' is the message text.

Indirect error messages have a different format. Refer to 9.2 for a discussion of Indirect error message processing.

9.1.1 VAX–11 RSX Facilities

This portion of Section 9 documents messages for all the facilities listed. Indirect messages are described in 9.2 and RSX–11 utility and component messages are described in the appropriate RSX–11M and RSX–11M–PLUS documentation. Table 9–1 notes the sections of this manual that contain mor information about each facility.

Table 9–1 Facilities and Associated Documentation

Facility Abbreviation	Full Name	Section
BTRAN	Back Translator (BACKTRANS)	Section 6
MCR	Monitor Console Routine (MCR)	Section 4
RSX	Applications Migration Executive (AME)	Sections 7 and 8

9.1.2 Severity Levels

The severity level of a message is included in the status message from the component. You can identify the type of message you have received by its severity level.

Success and information messages inform you that the system has performed your request. In some cases, the command processing continues after a success message is issued. In other cases, the message indicates that the command processing is complete.

Warning messages indicate that the command may have performed some, but not all, of your request, and that you may have to verify the command or program output.

Error messages indicate that the output or program result is incorrect, but that the system may attempt to continue execution.

Fatal messages indicate that the system cannot continue executing the request.

9.1.3 Linked Messages

Frequently, more than one system message will appear linked together. A series of messages can help you determine the location of an error. Linked messages usually display the return status of the different facilities called and provide a detailed explanation of the error.

9.1.4 VAX–11 RSX Messages

ABKEYW, ambiguous qualifier or keyword - supply more characters

Facility: MCR

Explanation: Too few characters were used to truncate a keyword or qualifier name to make the keyword or qualifier name unique.

User Action: The rejected portion of the command is displayed between backslashes (\ \). Reenter the command; specify at least four characters of the keyword or qualifier name.

ABSENT, entity or value absent from command string

Facility: MCR

Explanation: ABSENT is returned by the CLI$PRESENT CLI interface routine to indicate that the specified entity has not been indicated in the command line.

User Action: None.

ABVERB, ambiguous command verb - supply more characters

Facility: MCR

Explanation: Too few characters were used to truncate a command name to make the command name unique.

User Action: Reenter the command; specify at least four characters of the command name.

ACCVIO, compatibility mode access violation

Facility: RSX

Explanation: The RSX–11 task image attempted to access an area of memory outside its virtual address space. In VAX–11 RSX, the program counter (PC) contains the address of the instruction that caused the error. In RSX–11, the PC contains the address following the erroneous instruction. This message corresponds to the RSX–11 termination message MEMORY PROTECTION VIOLATION.

User Action: Determine which instruction or statement caused the error and correct the error.

ALLOC, 'device' allocated

Facility: MCR

Explanation: The specified device was allocated.

User Action: None.

ATTACHED, terminal now attached to process 'process name'

Facility: MCR

Explanation: This message is output by the MCR command ATTACH when it attaches to a process.

User Action: None.

BADCTLMSK, bad control character enable/disable mask

Facility: MCR

Explanation: This message is returned by two MCR callback routines if an invalid control character has been specified with the bit mask parameter.

User Action: Ensure that you have only specified a Y and then try again.

BADLIBREF, invalid library/common reference, 'library-name'

Facility: RSX

Explanation: The RSX–11 task image referred to a library or common area that was altered after the task image was built and is now inconsistent with the task image.

User Action: Rebuild the task image with the new library or common area.

BADSTACK, invalid compatibility mode stack pointer

Facility: RSX

Explanation: The RSX–11 task image's stack pointer (SP) points to an area of memory outside the task image's virtual address space. This message corresponds to the RSX–11 termination message BAD STACK.

User Action: Determine the instruction or statement where the error occurred and correct the error.

BADSTRLVL, unrecognized structure level for command tables

Facility: MCR

Explanation: Obsolete command tables (MCRTABLES) were used.

User Action: Use the correct version of tables in the SYS$LIBRARY:MCRTABLES file.

BADTEXT, \'text' \'text'

Facility: MCR

Explanation: An image request to MCR resulted in an error. The text supplies additional information for tracing the problem.

User Action: Examine the requests your image makes to MCR. In particular, check that qualifier lists include all qualifiers that MCR will allow with a command.

BREAK, compatibility mode breakpoint trap

Facility: RSX

Explanation: The RSX–11 task image executed a BPT instruction. This message corresponds to the RSX–11 termination message BPT EXECUTION.

User Action: Determine the instruction or statement where the error occurred and correct the error.

BUFOVF, command buffer overflow - shorten expression or command line

Facility: MCR

Explanation: The command string, including continuation lines, is longer than the buffer used by MCR to store a command.

User Action: Shorten and reenter the command; possibly enter more than one command.

BUGCHECK, internal consistency failure

Facility: RSX

Explanation: An internal error was detected in the RSX–11 facility.

User Action: Collect as much information about the problem as is possible. Submit a Software Performance Report (SPR) with a sample program that reproduces the problem. Notify your system manager.

CMDGONE, this is no longer a valid command - see documentation

Facility: MCR

Explanation: The command you entered is no longer available in MCR.

User Action: Read Section 4 to determine what commands will help you accomplish your task.

COMMA, requested value is terminated by a comma

Facility: MCR

Explanation: This message, returned by the CLI$GETVALUE CLI interface routine, indicates that the requested value was terminated by a comma. The value is part of a list.

User Action: None.

COMPLX, expression too complex - simplify

Facility: MCR

Explanation: MCR cannot analyze an arithmetic expression in a command; it contains too many operators and/or parentheses.

User Action: Shorten the expression and reenter the command. If necessary, execute more than one command.

CONCAT, requested value is terminated by a plus sign

Facility: MCR

Explanation: This message, returned by the CLI$GETVALUE CLI interface routine, indicates that the requested value was terminated by a plus sign. The value is part of a concatenated list.

User Action: None.

CONFLICT, illegal combination of command elements - check documentation

Facility: MCR

Explanation: Two qualifiers, keywords, or parameters that cannot be used in combination were used in the same command line.

User Action: Refer to Section 4 to determine which parameters, qualifiers, or keywords cannot be specified together. Reenter the correct command line.

CONFQUAL, conflicting qualifiers

Facility: MCR

Explanation: Two or more conflicting qualifiers were specified.

User Action: Refer to Section 4 to determine the qualifiers in conflict and reenter the command.

DEFAULTED, entity defaulted present in command string

Facility: MCR

Explanation: This status message, returned by the CLI$PRESENT CLI interface routine, indicates that the specified entity is not explicitly present in the command line but is present by default.

User Action: None.

DIRECT, invalid directory syntax - check brackets and other delimiters

Facility: MCR

Explanation: The directory name in a file specification that is entered for a command contains an illegal character. Or that is, a command that expects a directory name string did not find square brackets ([ ]) or angle brackets (< >) in the specified parameter. This error can occur if a physical device name is specified without a colon terminating it.

User Action: Verify the directory name (or logical name, or logical name translation, if a logical name was specified) for a spelling error. Correct the directory syntax and reenter the command.

DUPLMBX, receive data mailbox "!AS" already exists

Facility: RSX

Explanation: The specified logical name already exists. Therefore, VAX–11 RSX could not correctly initialize the Receive Data mailbox.

When the VAX–11 RSX AME activates a compatibility mode task, it creates a mailbox to be used in emulating the Receive Data directive. Other tasks using the Send Data directive send messages to this mailbox. VAX–11 RSX assigns a logical name to this mailbox in the form RCVDtask. The logical name is substituted for "!AS" in the message text.

User Action: Use the STOP command to stop the subprocess and release the mailbox if a compatibility mode task with the same task name was run and spawned a subprocess that used the Send Data directive. If the subprocess was not properly stopped at task termination, a VMS channel may still be assigned to the Receive Data mailbox, thus preventing VAX/VMS from deleting the mailbox.

If another task with the same task name is running in your UIC group, you cannot run your task until the other task completes.

EXITSTATUS, RSX–11M exit with status

Facility: RSX

Explanation: The RSX–11 task image terminated by issuing an Exit With Status directive that specified an exit status of normal, warning, error, or severe error termination. The symbols $STATUS and $SEVERITY contain the termination status and severity code, respectively.

User Action: This message does not necessarily indicate an abnormal termination of the task image. You can display the status with the command SHOW SYMBOL/GLOBAL, or you can test $STATUS or $SEVERITY with the DCL command IF or the MCR directive .IF.

EXPSYN, invalid expression syntax - check operators and operands

Facility: MCR

Explanation: The arithmetic expression in the command is not syntactically correct; the command interpreter cannot analyze the expression. An invalid expression can occur as a result of:

- Missing operators between values
- Unmatched parentheses

User Action: Examine the expression for any syntactic errors. Correct the expression and reenter the command.

EXTERNABRT, external abort

Facility: RSX

Explanation: The RSX–11 task image was aborted by another process issuing either a Force Exit ($FORCEX) system service, an Abort directive, or a STOP command.

User Action: None.

ILLEGOPT, option supported by another program

Facility: BTRAN

Explanation: The image invoked by a DCL command can invoke other images based on the qualifiers specified. An error occurs when you enter a command with a qualifier that is legal only when other qualifiers are enterec For example, the /OWNER_UIC qualifier for the DCL command CREATE applies only to the creation of a directory; if you specify this qualifier when you create a file, ILLEGOPT is issued.

User Action: Refer to Section 6 to determine which qualifiers are related an reenter the command.

ILLINST, compatibility mode illegal instruction

Facility: RSX

Explanation: The RSX–11 task image attempted to execute either a JMP or JSR instruction with a register destination. In VAX–11 RSX, the PC contains the address of the instruction that caused the error. In RSX–11, the PC contains the address following the erroneous instruction. This message corresponds to the RSX–11 termination message ILLEGAL INSTRUCTION.

User Action: Determine which instruction or statement caused the error and correct the error.

ILLVAL, illegal value is present

Facility: MCR

Explanation: The qualifier or parameter value is not within the legal range o values.

User Action: Refer to Section 4 for the legal values and reenter the command.

IMAGEFNF, image file not found 'file-name'

Facility: MCR

Explanation: You attempted to run a nonexistent task image.

User Action: Check for typing errors and retry.

IMAGETERM, 'task' terminated, 'current-time'

Facility: RSX

Explanation: The compatibility mode task image has terminated abnormally.

User Action: None.

IMGFNF, image file not found

Facility: MCR

Explanation: Your attempt to execute a task failed because the task image file could not be found.

User Action: Use the DIRECTORY command to verify the existence of the file. Ask the system manager to restore the file if necessary.

IMGNAME, image file 'file-name'

Facility: MCR

Explanation: This message is issued with an associated message indicating the problem.

User Action: Take corrective action based on the associated message.

INSFDYNMEM, insufficient virtual address space

Facility: RSX

Explanation: VAX–11 RSX required more virtual address space to emulate a requested RSX–11 service than a process is allowed.

User Action: Check for a programming error; the task could be in a loop. Correct the source program.

INSFPRM, missing command parameters - supply all parameters

Facility: MCR

Explanation: The command cannot be executed because one or more required parameters are missing from the command.

User Action: Refer to Section 4 to check the required parameters. Reenter the command specifying the required parameters.

INVFILSPE, input or output file specification too long - shorten

Facility: MCR

Explanation: A file specification was invalid in a PRINT or SUBMIT command. A node name is required in an operation with /REMOTE, and is invalid without /REMOTE.

User Action: Reenter the command, specifying a valid file specification.

INVKEY, invalid keyword

Facility: MCR

Explanation: An unknown keyword was specified for a qualifier value or parameter.

User Action: Refer to Section 4 for the legal keywords and reenter the command.

INVQUAL, invalid qualifier

Facility: MCR

Explanation: The task requested qualifier information from MCR, but the task images qualifier list did not contain an entry for one of the qualifiers typed in the command line.

User Action: Examine the qualifier list in the task image to determine whether the qualifier specified is one that MCR allows.

INVQUALNUM, invalid qualifier number

Facility: MCR

Explanation: This message is associated with an internal status code returnec from the MCR result parse routine. It indicates an erroneous qualifier.

User Action: Collect as much information as possible and submit a Software Performance Report (SPR).

INVREQTYP, invalid request type

Facility: MCR

Explanation: This message is associated with an internal status code returned from the MCR result parse routine. It indicates a request to perform an unimplemented function.

User Action: Collect as much information as possible and submit a Software Performance Report (SPR).

INVROUT, invalid routine specified in command tables - check .CLD file

Facility: MCR

Explanation: The command you typed contains an entry in the command tables, but the internal CLI routine named in the tables does not exist in MCR. The command may exist in DCL.

User Action: Check the *VAX/VMS DCL Dictionary* to determine whether the command exists in DCL. Also, check your .CLD files if you have added new commands to the MCRTABLES. If you added new command definitions, you may need to correct them.

INVTAB, command tables have invalid format - see SET COMMAND documentation

Facility: MCR

Explanation: The set of command tables available to MCR is not in the correct format. The file SYS$LIBRARY:MCRTABLES.EXE may be corrupt.

User Action: Obtain the original version of the MCRTABLES from your distribution kit. If you need to modify the table to add your own commands, see the SET COMMAND documentation in Section 4.

INVUIC, invalid UIC - must be of the form [group,member]

Facility: MCR

Explanation: The command contains an invalid UIC. The group and member portions of a UIC must both be specified and must be separated by a comma.

User Action: Correct the UIC and reenter the command.

IOT, compatibility mode IOT

Facility: RSX

Explanation: The RSX–11 task image executed an IOT instruction, and no SST vector was enabled to handle this condition. This message corresponds to the RSX–11 termination message IOT EXECUTION; it occurs only if the task executed an IOT instruction and no SST vector was enabled to handle this condition.

User Action: Determine the instruction or statement at which the error occurred and correct the error.

IVCHAR, invalid numeric value - replace character string with a number

Facility: MCR

Explanation: MCR encountered a character string where it expected a numeric value as the object of an arithmetic operation in an expression, or as the value for a qualifier.

User Action: Examine the command string for an undefined or incorrectly spelled symbol name, a missing radix operator (for example, %X) preceding a hexadecimal or octal value, an illegal value following a radix operator (for example, %O9), or a syntax error. Reenter the command.

IVKEYW, unrecognized keyword - check spelling

Facility: MCR

Explanation: The keyword specified in the command is not valid for the command.

User Action: The rejected portion of the command is displayed between backslashes. Refer to Section 4 for a list of valid keywords and options.

IVOPER, unrecognized operator in expression - check spelling and syntax

Facility: MCR

Explanation: The arithmetic expression contains an unrecognized operator. The rejected portion of the command is displayed between backslashes.

User Action: Refer to Section 4 for a list of the valid operators. Reenter the command.

IVPROT, invalid protection code - only use combinations of R,W,E, and D

Facility: MCR

Explanation: The protection code specified for a file or volume is not in the correct syntax.

User Action: Refer to Section 4 for the correct syntax and reenter the command.

IVQUAL, unrecognized qualifier - check validity, spelling, and placement

Facility: MCR

Explanation: The qualifier was spelled incorrectly or improperly placed in the command line.

User Action: Correct the command line.

IVQLOC, invalid qualifier location - place after a parameter

Facility: MCR

Explanation: A qualifier that can be used only to qualify a parameter in a command was placed after the command name.

User Action: Reenter the command; place the qualifier after the specification of the parameter it actually qualifies. Refer to Section 4 to determine the meaning and placement of the qualifier.

IVVALU, invalid value syntax - see command documentation

Facility: MCR

Explanation: The value for a qualifier is not specified correctly; or, the command is not being used in its proper context. The rejected portion of the command is displayed between backslashes.

User Action: Refer to Section 4 to determine the proper syntax or use of the command and reenter the command.

IVVERB, unrecognized command verb - check spelling

Facility: MCR

Explanation: The first word of the command is not a valid MCR command. The rejected portion of the command is displayed between backslashes.

User Action: Check the spelling of the command name and reenter the command.

KEYVALOVR, too many characters in qualifier keyword value

Facility: BTRAN

Explanation: The value specified for a keyword in a qualifier is too long.

User Action: Refer to Section 6 for the valid keywords and reenter the command.

LIBNOTFND, open failure on library/common file, 'library-name'

Facility: RSX

Explanation: The RSX–11 library or common area requested is not present in SYS$LIBRARY or does not have a file type of EXE.

User Action: Depending on the cause of the error, move the library to the device and directory identified as SYS$LIBRARY, rename the file type to EXE, or do both.

LOADERROR, error loading image file, execution aborted

Facility: RSX

Explanation: An error occurred during an attempt to load the RSX–11 task image into memory. Normally, this error is the result of a disk failure or a library error. This message can also result from an insufficient value for some quota. In the case of a library error, an accompanying message indicates the reason for the error.

User Action: Check the status of the device and reenter the command. Attempt to recreate the task image in another file. If the failure persists, notify the system manager.

LOCNEG, entity explicitly and locally negated in command string

Facility: MCR

Explanation: The specified qualifier was present in its negated form (with /NO) and was used as a local qualifier.

User Action: None.

LOCPRES, entity value is locally present in command string

Facility: MCR

Explanation: This message is returned by the CLI interface routine, CLI$PRESENT, and indicates that the specified qualifier is present and that you used it as a local qualifier.

User Action: None.

MAXPARM, too many parameters - reenter command with fewer parameters

Facility: MCR

Explanation: The command contained more than the maximum number of parameters allowed. This error can be caused by one of the following:

- Leaving blanks on a command line where a special character, for example, a comma (,) or a plus sign (+), is required
- Using logical names that, when substituted or translated, contain embedded blank characters
- Failure to place quotation marks around a character string

User Action: Determine the reason for the error and correct the syntax of the command. Refer to Section 4 to determine the correct number of parameters for the command.

NEGATED, entity explicitly negated in command string

Facility: MCR

Explanation: The specified qualifier was present in its negated form (with /NO) and was used as a local qualifier.

User Action: None.

NOCCAT, parameter concatenation not allowed - separate with commas

Facility: MCR

Explanation: The command accepts only a single parameter value rather than multiple parameter values separated by plus signs (+) for a particular parameter.

User Action: Reenter the command with a single parameter value for that parameter.

NOCHANGE, neither the directory nor the UIC can be changed as specified

Facility: MCR

Explanation: Under normal conditions, the MCR command SET /UIC changes both your default directory and your UIC, to make the VAX/VMS and RSX–11 systems compatible. However, because VAX/VMS does not implement directories that are greater than [377,377] and because the UIC you specified was greater than [377,377], your default directory was not changed.

You must have the privilege CMKRNL to change your UIC. However, since you do not have this privilege, your UIC was not changed.

User Action: If you need to change your process UIC, you will need the privilege CMKRNL. When you have this privilege, the command SET /UIC will change your UIC.

NOCOMD, no command on line - reenter with alphabetic first character

Facility: MCR

Explanation: The command string contains one or more nonalphanumeric characters, but no alphanumeric characters that can be evaluated. Or, a command continued on more than one line did not contain a blank space or tab required to delimit parameters.

User Action: Make sure that the first character of your command is alphabetic. Check the command spelling and syntax in Section 4.

If the error occurred in a command procedure, verify the syntax of the command lines and check for continued lines.

NOCOMM, unable to initialize process mailbox or common event flags

Facility: RSX

Explanation: The system encountered an error when you attempted to set up the process receive mailbox or common event flags used for communication with other processes.

User Action: Ensure that the duplicate process name does not exist, and that you have a quota large enough to create a mailbox and common event flag cluster.

NODIRCHG, directory cannot be changed to match long UIC format

Facility: MCR

Explanation: Under normal conditions, the MCR command SET /UIC changes both your default directory and your UIC, to make the VAX/VMS and RSX–11 systems compatible. However, because there is no VAX/VMS directory format corresponding to UICs greater than [377,377] and because the UIC that you specified was greater than [377,377], your default directory was not changed. Your process UIC was changed as you specified.

User Action: None.

NOKEYW, qualifier name is missing - append the name to the slash

Facility: MCR

Explanation: The slash character (/) was present on a command, but was not followed by a qualifier keyword name.

User Action: Reenter the command specifying the qualifier.

NOLIST, list of parameter values not allowed - supply a single value

Facility: MCR

Explanation: The command accepts only a single parameter value for a parameter that contains multiple parameter values separated by commas (,).

User Action: Reenter the command; specify only one file. If necessary, issue the command once for each file specified.

NONRSXEMT, non-RSX–11M EMT execution

Facility: RSX

Explanation: The RSX–11 task image attempted to execute an EMT instruction that is not a valid RSX–11 EMT. This message corresponds to the RSX–11 termination message NON-RSX EMT EXECUTION.

User Action: Determine the instruction or statement at which the error occurred and correct the error.

NOOPTPRS, no option present on command

Facility: MCR

Explanation: This message is associated with an internal status code returned by the MCR result parse routine. The operating system does not normally display this message.

User Action: Note the condition. Modify the source program, if necessary, to detect and respond to the condition.

NOPMD, post-mortem dump failure, 'file-name'

Facility: RSX

Explanation: VAX–11 RSX could not perform a postmortem dump as requested; that is, it could not create the indicated dump file in the current default directory.

User Action: Verify the current default directory and protection with the commands SHOW DEFAULT and SHOW PROTECTION. If necessary, use the command SET DEFAULT or the command SET PROTECTION to set the correct directory and protection; retry the postmortem dump.

NOQUAL, qualifiers not allowed - supply only verb and parameters

Facility: MCR

Explanation: The command has no qualifiers but was specified with a qualifier.

User Action: Reenter the command; do not specify any qualifiers.

NOSUCHDEV, TKB-assigned device not found, 'device-name':, lun=logical-unit

Facility: RSX

Explanation: VAX–11 RSX cannot preassign a device specified at task-build time. The indicated device does not map to a VAX/VMS physical device. The message indicates the device name causing the error and its Logical Unit Number (LUN).

User Action: Either rebuild the task image specifying a VAX/VMS physical or logical device name, or assign the indicated device name as the logical name for a VAX/VMS device prior to running the task image. Refer to Section 2 for more information on device name mapping.

NOTFR, no transfer address

Facility: MCR

Explanation: The task image file had no transfer address.

User Action: See the Link Utility in the *VAX/VMS Utilities Reference Volume* for details on how to specify transfer addresses.

NOTIFY, /NOTIFY not allowed - do not use with /WAIT or in batch jobs

Facility: MCR

Explanation: /NOTIFY should not be used unless you specify the /NOWAIT qualifier or the SPAWN command is executed from within a noninteractive process.

User Action: Do not use /NOTIFY.

NOTIMPLEM, not implemented

Facility: BTRAN

Explanation: An internal error was detected in the BACKTRANS program.

User Action: Collect as much information as possible, give examples of the command line, and submit a Software Performance Report (SPR). Notify the system manager.

NOTNEG, qualifier or keyword not negatable - remove 'NO' or omit

Facility: MCR

Explanation: The word 'NO' preceded a qualifier or keyword, but the qualifier or keyword cannot be specified as a negative.

User Action: Refer to Section 4 for a list of valid keywords.

NOTSKIMG, file is not a valid RSX–11M task image

Facility: RSX

Explanation: The task image that you requested for execution is not a valid RSX–11 task image. This can apply to the following:

- RUN commands
- Task names specified in response to MCR's prompt
- Task names specified in response to the DCL prompt MCR>

- Tasks requested by the compatibility mode task

User Action: Rebuild the image using the correct Task Builder. VAX–11 RSX provides the RSX–11M Version 4.1 Task Builder with the system.

NOVALU, value not allowed - remove value specification

Facility: MCR

Explanation: The qualifier does not accept a value specified with a keyword, a numeric value, or a file specification.

User Action: Refer to Section 4 for the correct syntax and reenter the command.

NOVALUE, value not present

Facility: MCR

Explanation: This message is associated with an internal status code returned by the MCR result parse routine. The operating system does not normally display this message.

User Action: Note the condition. Modify the source program, if necessary, to detect and respond to the condition.

NULFIL, null file specification - supply valid file specification

Facility: MCR

Explanation: MCR expected a file specification to follow a prompting message, a plus sign (+), or a comma (,), but you did not enter a file specification.

User Action: Reenter the command; put the file specification in the proper position.

NUMBER, invalid numeric value - supply an integer

Facility: MCR

Explanation: A numeric value was specified for an MCR command that expects values in certain radices, or interprets values within a particular context. For example, the number 999 was entered when an octal value was required. Or an alphabetic value was specified in a context that requires a numeric value, for example, a file version number.

User Action: Use the MCR command HELP or refer to the command description in Section 4.

ODDADDR, compatibility mode odd address

Facility: RSX

Explanation: The RSX–11 task image attempted a word reference on a byte boundary. In VAX–11 RSX, the PC contains the address of the instruction that caused the error. In RSX–11, the PC contains the address of the instruction following the erroneous instruction. This message corresponds to the RSX–11 termination message ODD ADDRESS ERROR.

User Action: Determine which instruction or statement caused the error and correct it.

ONEVAL, list of values not allowed - supply a single value

Facility: MCR

Explanation: A qualifier that accepts only a single value was specified with multiple values.

User Action: Reenter the command specifying only one value for the qualifier.

OPTVALOVR, too many characters in qualifier value

Facility: BTRAN

Explanation: Too many characters were entered in a numeric value specified as a qualifier value.

User Action: Refer to Section 6 for the legal values and reenter the command.

OVRFLW, too much data - modify size of field or size of data

Facility: MCR

Explanation: The data value specified is larger than the area defined to receive it. For example, the command DEPOSIT requested that a value larger than a byte be deposited in a byte.

User Action: Verify the value and the area to receive it; correct the value or the receiving area and reenter the command.

PARMDEL, invalid parameter delimiter - use a space

Facility: MCR

Explanation: The command contains an invalid character either in the file specification or following the specification of a parameter.

User Action: Check the command string for a spelling or grammatical error. Reenter the command.

PMD, post-mortem dump, 'file-name'

Facility: RSX

Explanation: The RSX-11 task built with the postmortem dump option aborted. The postmortem dump followed the abort. The 'file-name' is the name of the postmortem dump file created by VAX-11 RSX.

User Action: None.

PRESENT, entity value is present in command string

Facility: MCR

Explanation: PRESENT is returned by the CLI$PRESENT CLI interface routine to indicate that the specified entity is explicitly present in the command line.

User Action: None.

PROC_ID, identification of created process is process-id

Facility: MCR

Explanation: The MCR command RUN successfully created a process. This message does not indicate whether the requested task image was started or completed successfully when the process began executing.

This message displays the process identification number of the created process.

User Action: None.

PWDLOCKED, password was locked to prevent change

Facility: MCR

Explanation: This message is returned for the command SET PASSWORD. The system manager has protected the account against attempts to change its password. Typically the password is locked in demonstration accounts intended for access by a large number of users.

User Action: If you need to make a change in such an account, consult the system manager.

PWDNOTVAL, old password validation error - password unchanged

Facility: MCR

Explanation: This message is returned for the command SET PASSWORD. MCR detected an old password validation error. You are required to enter the old password to help protect an unattended process. You may have mistyped the old password.

User Action: Retype the old password. If you cannot remember the old password, consult the system manager.

PWDNOTVER, new password verification error - password unchanged

Facility: MCR

Explanation: This message is returned for the command SET PASSWORD. The new password was not entered correctly twice. Entering the password the same way twice ensures that the password you intended is properly entered.

User Action: Repeat the entire sequence correctly.

PWDSYNTAX, illegal password syntax

Facility: MCR

Explanation: You specified an illegal password. Passwords may contain from 1 through 31 characters. The valid characters are:

A to Z
a to z
0 to 9
$ (dollar sign)
_ (underscore)

User Action: Reenter the password, using valid characters.

REFUSED, you cannot attach to that process

Facility: MCR

Explanation: Either you have attempted to attach to a process that is your own process or that is not part of your process tree, or your process and the process to which you are attempting to attach are not getting input from the same terminal.

User Action: None. You cannot perform the attempted operation.

REGMSG, 'register' = 'octal-value' 'hex-value'

Facility: RSX

Explanation: When an RSX–11 task image terminates abnormally, this message is used to display the value of each compatibility mode register at the time of task termination. The 'register' is the register name, and 'octal-value' and 'hex-value' are the values of the register in octal and hexadecimal radix. This message will follow the RSX-F-IMAGETERM message for each register (R0 through PC).

User Action: None.

REQPRMABS, required parameter is absent

Facility: MCR

Explanation: This message is associated with an internal status code returned by the MCR result parse routine. It indicates to the caller that not all parameters are present. The operating system does not normally display this message.

User Action: Note the condition. Modify your source program, if necessary, to detect and respond to the condition.

RESERVED, compatibility mode reserved instruction

Facility: RSX

Explanation: The RSX–11 task image attempted to execute one of the following instructions that are not allowed in compatibility mode:

```
HALT, WAIT, RESET, SPL, MARK
```

In VAX–11 RSX, the PC contains the address of the instruction that caused the error. In RSX–11, the PC contains the address of the instruction following the erroneous instruction. This message corresponds to the RSX–11 termination message RESERVED INSTRUCTION.

User Action: Determine the instruction or statement at which the error occurred and conect the error.

RETURNED, control returned to process 'process-name'

Facility: MCR

Explanation: This message is output by MCR when you log out of a spawned process.

User Action: None.

RSLOVF, result parse buffer overflow

Facility: MCR

Explanation: The buffer used by the command interpreter to store a command string during parsing has overflowed.

User Action: Simplify the command.

SPAWNED, process 'process-name' spawned

Facility: MCR

Explanation: This message is output by the MCR command SPAWN when it spawns a subprocess.

User Action: None.

SPWNIO, shared input or output files are not allowed - respecify

Facility: MCR

Explanation: You used a file that was already opened by MCR as input or output to SPAWN.

User Action: If input or output to MCR is SYS$INPUT or SYS$OUTPUT, do not specify the input or output qualifier. See the SPAWN command in Section 4 for details on input and output interpretation.

STR, user structure became invalid during an operation

Facility: RSX

Explanation: RSX–11 is attempting to return information into a user structure that was corrupted or unmapped during the operation.

User Action: Determine the instruction or statement at which the error occurred and correct the error.

STRINGOVR, result string

Facility: BTRAN

Explanation: The command string was backtranslated to an RSX–11 MCR command which made the resulting string greater than 80 characters (the maximum command string length for MCR).

User Action: Recast and reenter the command. If the command contains many file specifications, reorder them to take advantage of temporary file specification defaults. Or, if possible, issue several commands to process all files.

SUPERSEDE, previous value of 'value' has been superseded

Facility: MCR

Explanation: You redefined the specified logical name.

User Action: None.

SYMOVF, no room for symbol definitions - delete some symbols

Facility: MCR

Explanation: MCR cannot hold any more symbol definitions. This error is also issued if an attempt is made to define the explicit value for the reserved global symbols $STATUS and $SEVERITY.

User Action: Collect as much information as possible and submit a Software Performance Report (SPR).

TBIT, compatibility mode T-bit

Facility: RSX

Explanation: The RSX–11 task image executed an instruction requesting a T-bit trap. This message corresponds to the RSX–11 termination message T-BIT EXECUTION.

User Action: Determine which instruction or statement caused the error and correct the error.

TOOBIG, task exceeds compatibility mode address limits

Facility: RSX

Explanation: The VAX–11 RSX AME was requested to activate an RSX–11 task image with a root segment larger than 64K bytes.

User Action: Rebuild the task and either reduce the EXTTSK option (if you previously specified it), or reorganize the task's overlay structure (so the root segment does not exceed 64K bytes in size).

TRAP, compatibility mode TRAP execution

Facility: RSX

Explanation: The RSX–11 task image executed a TRAP instruction. This message corresponds to the RSX–11 termination message TRAP EXECUTION.

User Action: Determine the instruction or statement that caused the error and correct it.

UNPROPARM, unprocessed parameters

Facility: MCR

Explanation: The command contained more parameters than are currently supported.

User Action: Reenter the command; specify only the maximum number of parameters.

UNPROQUAL, unprocessed qualifiers

Facility: MCR

Explanation: This message is associated with a status code returned from the MCR result parse routine. The message indicates that qualifiers present with the command were ignored. The operating system does not normally display this message.

If this error occurred as a result of a MACRO–11 assembler operation, either the /SHOW or the /CROSS_REFERENCE qualifier was specified without the /LIST qualifier. The MACRO–11 assembler does not process the source file or produce an object module.

User Action: Note the condition. Modify the source program, if necessary, to detect and respond to the condition.

If either the /SHOW or the /CROSS_REFERENCE qualifier was specified without /LIST, reenter the command with the /LIST qualifier.

VALCNVERR, error converting value

Facility: MCR

Explanation: The keyword or keyword value was not converted correctly because, for example, a string was supplied when only a number is valid.

User Action: Retype the command and give the correct value.

VALREQ, missing qualifier or keyword value - supply all required values

Facility: MCR

Explanation: A value must be specified for the qualifier or keyword.

User Action: Retype the command and supply a value to the qualifier or keyword.

9.2 Indirect Message Processing

When Indirect encounters an error, it prints the appropriate error message and the command line in which the error occurred. If the line contained a substitution, the line is also displayed as it appeared before the substitution took place. Messages are displayed in both uppercase and lowercase letters (when the device is capable of displaying or printing lowercase letters).

Indirect error messages are divided into classes, depending on the level of severity. Class 2 errors can be handled with the <ERRCTL> symbol (see 5.4 in Section 5) and class 1 errors can be handled with the .ONERR directive (see 5.6.21 in Section 5). Class 0 errors must be corrected outside of Indirect.

The message descriptions that follow are alphabetized according to the message text. Information-only messages are explained in 9.2.1, and error messages are explained in 9.2.2.

9.2.1 Indirect Information-Only Messages

@ <EOF>

Facility: IND

Explanation: Indirect has reached the end-of-file for the outermost commaı file and is terminating execution.

User Action: None.

IND—Continuing

Facility: IND

Explanation: Indirect is resuming execution after a .PAUSE or .DELAY directive.

User Action: None.

IND—Delaying

Facility: IND

Explanation: A .DELAY directive was just executed, halting the processing (an indirect command file for a specified period of time.

User Action: None.

IND—Invalid answer or terminator

Facility: IND

Explanation: Any of the following occurred:

- In response to an .ASK question, you entered something other than 7, N, or null, followed by a RETURN.
- In response to an .ASK question, you did not enter a numeric value.
- You pressed the ESCAPE Key either without escape recognition enabled or as a character other than the first one following the question. The question will be repeated.

User Action: Enter an acceptable answer in response to the question.

IND—Pausing. To continue type "RES"

Facility: IND

Explanation: Indirect just executed a .PAUSE directive, which interrupted processing of an indirect command file to wait for user action.

User Action: Type the necessary command and task name as indicated in thε message.

IND—Value not in range

Facility: IND

Explanation: The response to an .ASKN or .ASKS question was not within the specified range. Indirect repeats the question.

User Action: Enter a value within the specified range.

).2.2 Indirect Error Messages

IND—Bad range or default specification

Facility: IND

Explanation: This is a class 1 error. One of the following actions caused this error:

- An illegal character was specified as a range or default argument. (Only numeric expressions are permitted.)
- The end of the range is greater than the beginning.
- The default is outside the range.
- The first value is missing in the range.
- The string length exceeds the maximum string length allowed.

User Action: Correct the range specification in the command file.

IND—Command file open error

Facility: IND

Explanation: This is a class 2 error. The file being invoked in an @file or @file/LB:module command line cannot be found or opened.

User Action: Verify that the file exists and that it is not locked.

IND—Data file error, code x.

Facility: IND

Explanation: This is a class 1 error. Indirect encountered an error while processing an .OPEN, .OPENA, .CLOSE, or .DATA directive or a data mode access to the secondary file.

User Action: See the *IAS/RSX-11 I/O Operations Reference Manual* for a definition of the numeric code x, or the description of the <FILERR> symbol in Section 5 of this manual. Respond to the error accordingly.

IND—File already open

Facility: IND

Explanation: This is a class 1 error. An .OPEN or .OPENA directive specified a file that was already open.

User Action: Remove the directive that specified the open file and execute the file again.

IND—File not found

Facility: IND

Explanation: This is a class 2 error. An @filename or .CHAIN directive specified an incorrect file name or nonexistent file.

User Action: Verify that the file exists; then enter the correct file name and execute the file.

IND—File not open

Facility: IND

Explanation: This is a class 1 error. Indirect encountered a .DATA or .CLOS directive that did not reference an open file.

User Action: Correct the file and verify that the referenced file is open.

IND—File read error

Facility: IND

Explanation: This is a class 2 error. An error was detected in reading the indirect command file. This error is usually caused by records that are more than 132 (10) bytes long.

User Action: Verify that the file you are reading is in the correct format.

IND—Illegal file number

Facility: IND

Explanation: This is a class 1 error. The file number in an .OPEN, .OPENA, .OPENR, .DATA, .ENABLE DATA, .READ, or .CLOSE directive is not in the range of 0 through 3.

User Action: Rewrite the file and use a legal file number.

IND—Illegal nesting

Facility: IND

Explanation: This is a class 1 error. Too many Begin-End blocks are nested in the indirect command file. The maximum nesting depth is limited to the size of the symbol table.

User Action: Rewrite the file and specify fewer Begin-End blocks.

IND—Initialization error, code x.

Facility: IND

Explanation: This is a class 0 error. Indirect failed to complete initialization when you invoked it. The following list gives the meaning of the displayed code number:

- Unable to acquire system information such as the UIC or device name
- Impure area setup failed
- Unable to acquire task-specific information
- Unable to acquire terminal-type information
- Unable to acquire the disk name and other information about the system device (SY:)
- Unable to allocate enough space for command and data I/O buffers; the EXTEND TASK directive failed to return sufficient space for Indirect to allocate the buffers
- Initialization of allocated buffers failed
- Initialization of the DATA file structures failed
- Allocation of FCS-11 buffers for data and command lines failed

- Symbol table initialization failed
- Initialization cleanup failed
- Unable to obtain initial command line

Error codes greater than 12 are returned by FMS–11 and other special purpose initialization modules.

User Action: If any of these errors persist, submit a Software Performance Report (SPR) with any other pertinent information.

IND—Invalid keyword

Facility: IND

Explanation: This is a class 1 error. An unrecognized keyword (preceded by a period) was specified.

User Action: Rewrite the file and specify a valid keyword.

IND—Label not at beginning of line

Facility: IND

Explanation: This is a class 1 error. The specified label does not start in the first column of the line. All labels must do so.

User Action: Rewrite the file so that the label is in the first column.

IND—Maximum indirect file depth exceeded

Facility: IND

Explanation: This is a class 2 error. An attempt was made to reference an indirect command file at a nested depth greater than the maximum specified in the build file for the Indirect task.

User Action: Restructure your files.

IND—No pool space

Facility: IND

Explanation: This is a class 2 error. The dynamic memory allocation has been exhausted.

User Action: Wait for more pool space to become available or see your system manager.

IND—Null control string

Facility: IND

Explanation: This is a class 1 error. The control string specified with the .PARSE directive was null (there were no characters between the quotation marks).

User Action: Rewrite the file and place the characters between the quotation marks.

IND—Numeric under- or overflow

Facility: IND

Explanation: This is a class 2 error. The evaluation of a numeric expression yielded a value outside the range 0 through 177777 (8).

User Action: Reevaluate your program to determine the cause of the error. It may be necessary to rewrite the program.

IND—Redefining a read-only symbol

Facility: IND

Explanation: This is a class 2 error. An attempt was made to assign a new value to a read-only symbol. Read-only symbols cannot be overwritten.

User Action: Specify the correct symbol.

IND—Redefining symbol to different type ssssss

Facility: IND

Explanation: This is a class 1 error. An .ASK, .ASKN, .ASKS, .READ, .SETF, .SETL, .SETN, .SETS, or .SETT directive was used in an attempt to set the specified, already defined symbol to a different type. The first definition of a symbol determines its type (logical, numeric, or string); subsequent value assignments must conform to the original type.

User Action: Create a new symbol and verify that you refer to the correct symbol.

IND—.RETURN without .GOSUB

Facility: IND

Explanation: This is a class 1 error. A .RETURN directive was specified without a previous call to a subroutine (.GOSUB).

User Action: Rewrite your program to include the .GOSUB call.

IND—Spawn failure

Facility: IND

Explanation: This is a class 1 error. Indirect could not initiate the execution of a user command task.

User Action: Try executing the file again. If you get the same error, see your system manager.

IND—String expression larger than 132 bytes

Facility: IND

Explanation: This is a class 2 error. An attempt was made to generate a string expression longer than 132 (10) characters.

User Action: Rewrite the file to use shorter string symbols.

IND—String substitution error

Facility: IND

Explanation: This is a class 1 error. Indirect encountered an error during a substitution operation. A probable cause for the error is either the omission of a second apostrophe or pound sign, or the specification of an undefined symbol.

User Action: Correct the error according to the problem indicated in the message.

IND—Subroutine nesting too deep

Facility: IND

Explanation: This is a class 1 error. The maximum subroutine nesting level was exceeded. The maximum level is specified in the build file for the Indirect task.

User Action: Rewrite your program.

IND—Symbol table overflow ssssss

Facility: IND

Explanation: This is a class 2 error. The symbol table was full and there was no space for symbol ssssss.

User Action: Rewrite your program to use fewer symbols or to reuse existing symbols (see the description of the .ERASE directive in Section 5).

IND—Symbol type error ssssss

Facility: IND

Explanation: This is a class 1 error. The symbol ssssss was used out of context for its type; for example, a numeric expression referred to a logical symbol. Only symbols of the same type can be compared.

User Action: Rewrite your program to refer to the correct symbol.

IND—Syntax error

Facility: IND

Explanation: This is a class 2 error. The format of the specified command line is incorrect.

User Action: Rewrite the specified command line in your program.

IND—Undefined label .label:

Facility: IND

Explanation: This is a class 1 error. The label .label: specified in a .GOTO, .GOSUB, or .ONERR directive could not be found.

User Action: Check for the specified label in your program.

IND—Undefined symbol ssssss

Facility: IND

Explanation: This is a class 1 error. The symbol ssssss was referred to, but i had not been defined.

User Action: Check your program and make sure that you refer to the correc symbol.

10 Error Codes

This section provides information on RSX–11 error codes. The RSX–11 I/O error codes are listed in 10.1, and the RSX–11 directive error codes are listed in 10.2.

10.1 I/O Error Codes

The following table lists RSX–11 I/O error codes. I/O error codes are found in the low-order byte of the first word of the I/O status block. The octal number listed is the 2's complement of the decimal number.

Table 10–1 RSX–11 I/O Error Codes

Error Number		Code	
Octal	Decimal	Name	Meaning
377	–01	IE.BAD	Bad parameters
376	–02	IE.IFC	Invalid function code
375	–03	IE.DNR	Device not ready
374	–04	IE.VER	Parity error on device
373	–05	IE.ONP	Hardware option not present
372	–06	IE.SPC	Illegal user buffer
371	–07	IE.DNA	Device not attachable
370	–08	IE.DAA	Device already attached
367	–09	IE.DUN	Device not attached
366	–10	IE.EOF	End-of-file detected
365	–11	IE.EOV	End-of-volume detected
364	–12	IE.WLK	Write attempted to locked unit
363	–13	IE.DAO	Data overrun
361	–15	IE.ABO	Request terminated
360	–16	IE.PRI	Privilege violation
357	–17	IE.RSU	Shareable resource in use
356	–18	IE.OVR	Illegal overlay request
355	–19	IE.BYT	Odd byte count (or virtual address)
354	–20	IE.BLK	Logical block number too large
353	–21	IE.MOD	Invalid UDC module number
352	–22	IE.CON	UDC connect error
351	–23	IE.NOD	Caller's nodes exhausted

Table 10–1 (Cont.) RSX–11 I/O Error Codes

Error Number		Code	
Octal	Decimal	Name	Meaning
350	–24	IE.DFU	Device full
347	–25	IE.IFU	Index file full
346	–26	IE.NSF	No such file
345	–27	IE.LCK	Locked from read/write access
344	–28	IE.HFU	File header full
342	–30	IE.CKS	File header checksum failure
341	–31	IE.WAT	Attribute control list format error
340	–32	IE.RER	File processor device read error
337	–33	IE.WER	File processor device write error
336	–34	IE.ALN	File already accessed on LUN
335	–35	IE.SNC	File ID, file number check
334	–36	IE.SQC	File ID, sequence number check
333	–37	IE.NLN	No file accessed on LUN
332	–38	IE.CLO	File not properly closed
331	–39	IE.NBF	No buffer space available for file
330	–40	IE.RBG	Illegal record size
327	–41	IE.NBK	File exceeds space allocated, no blocks
326	–42	IE.ILL	Illegal operation on File Descriptor Block
325	–43	IE.BTP	Bad record type
324	–44	IE.RAC	Illegal record access bits set
323	–45	IE.RAT	Illegal record attribute bits set
322	–46	IE.RCN	Illegal record number—too large
320	–48	IE.2DV	Rename—two different devices
317	–49	IE.FEX	Rename—new file name already in use
316	–50	IE.BDR	Bad directory file
315	–51	IE.RNM	Cannot rename old file system
314	–52	IE.BDI	Bad directory syntax
313	–53	IE.FOP	File already open
312	–54	IE.BNM	Bad file name
311	–55	IE.BDV	Bad device name
310	–56	IE.BBE	Bad block on device
307	–57	IE.DUP	Enter—duplicate entry in directory

Table 10–1 (Cont.) RSX–11 I/O Error Codes

Error Number		Code	
Octal	Decimal	Name	Meaning
305	–59	IE.FHE	Fatal hardware error on device
304	–60	IE.NFI	File ID not specified
303	–61	IE.ISQ	Illegal sequential operation
302	–62	IE.EOT	End-of-tape detected
301	–63	IE.BVR	Bad version number
300	–64	IE.BHD	Bad file header
277	–65	IE.OFL	Device offline
276	–66	IE.BCC	Block check, CRC, or framing error
265	–75	IE.EXP	File expiration date not reached
264	–76	IE.BTF	Bad tape format
263	–77	IE.NNC	Not ANSI 'D' format byte count
256	–82	IE.IES	Invalid escape sequence
255	–83	IE.PES	Partial escape sequence
254	–84	IE.ALC	Allocation failure
253	–85	IE.ULK	Unlock error
252	–86	IE.WCK	Write check failure

For more information, see the *IAS/RSX-11 I/O Operations Reference Manual.*

10.2 Directive Error Codes

Directives in the Directive Status Word (DSW) return the error codes listed in the following table. The octal error number listed is the 2's complement of the decimal number.

Table 10–2 RSX–11 Directive Error Codes (Standard System Codes)

Error Number		Code	
Octal	Decimal	Name	Meaning
377	–01	IE.UPN	Insufficient dynamic storage
376	–02	IE.INS	Specified task not installed
375	–03	IE.PTS	Partition too small for task
374	–04	IE.UNS	Insufficient dynamic storage for send
373	–05	IE.ULN	Unassigned LUN
372	–06	IE.HWR	Device handler not resident
371	–07	IE.ACT	Task already active

Table 10–2 (Cont.) RSX–11 Directive Error Codes (Standard System Codes)

Error Number		Code	
Octal	Decimal	Name	Meaning
370	–08	IE.ITS	Directive inconsistent with task state
367	–09	IE.FIX	Task already fixed/unfixed
366	–10	IE.CKP	Issuing task not checkpointable
365	–11	IE.TCH	Task checkpointable
	–12		(reserved)
	...		...
	–14		(reserved)
361	–15	IE.RBS	Receive buffer too small
360	–16	IE.PRI	Privilege violation
357	–17	IE.RSU	Shareable resource in use
356	–18	IE.NSW	No swap space available
355	–19	IE.ILV	Illegal table vector specified
	...		...
	–79		(reserved)
260	–80	IE.AST	Directive issued/not issued from AST
257	–81	IE.MAP	Illegal mapping specified
256	–82		(reserved)
255	–83	IE.IOP	Window has I/O in progress
254	–84	IE.ALG	Alignment error
253	–85	IE.WOV	Address window allocation overflow
252	–86	IE.NVR	Invalid region ID
251	–87	IE.NVW	Invalid address window ID
250	–88	IE.ITP	Invalid TI parameter
247	–89	IE.IBS	Invalid send buffer size (greater than 255)
246	–90	IE.LNL	LUN locked in use
245	–91	IE.IUI	Invalid UIC
244	–92	IE.IDU	Invalid device or unit
243	–93	IE.ITI	Invalid time parameters
242	–94	IE.PNS	Partition/region not in system
241	–95	IE.IPR	Invalid priority (greater than 250)
240	–96	IE.ILU	Invalid LUN

Table 10–2 (Cont.) RSX–11 Directive Error Codes (Standard System Codes)

Error Number		Code	
Octal	Decimal	Name	Meaning
237	–97	IE.IEF	Invalid event flag (greater than 64)
236	–98	IE.ADP	Part of DPB out of user's space
235	–99	IE.SDP	DIC or DPB size invalid

For more information, see the *RSX-11M/M-PLUS and Micro/RSX Executive Reference Manual* and the *IAS/RSX-11 I/O Operations Reference Manual.*

Index

Index

VAX-11 RSX
Compatibility Mode
Reference Manual
AA-CJ40A-TE

READER'S COMMENTS

Note: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

Did you find errors in this manual? If so, specify the error and the page number.

Please indicate the type of user/reader that you most nearly represent:

- □ Assembly language programmer
- □ Higher-level language programmer
- □ Occasional programmer (experienced)
- □ User with little programming experience
- □ Student programmer
- □ Other (please specify) ______________________

Name ______________________ Date ______________________

Organization ______________________

Street ______________________

City ______________________ State ______________________ Zip Code ______________________
or Country

— — Do Not Tear - Fold Here and Tape — — — — — — — — — — — — — — — — — — —

digital

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

SSG PUBLICATIONS ZK1-3/J35
DIGITAL EQUIPMENT CORPORATION
110 SPIT BROOK ROAD
NASHUA, NEW HAMPSHIRE 03062-2698

— — Do Not Tear - Fold Here —