VAX–11 RSX Version 2.5 Installation Guide and Release Notes

Order Number: AA–CJ39D–TE

This manual describes the installation procedures and new features of VAX–11 RSX Version 2.5

Revision/Update Information:	This manual supersedes previous VAX–11 RSX Release Notes.
Software Version:	VAX–11 RSX Version 2.5 VMS Version 5.0 and later

digital equipment corporation
maynard, massachusetts

First Printing, June 1990

The postpaid Reader's Comments forms at the end of this document request your critical evaluation to assist in preparing future documentation.

Production Note

This book was produced with the VAX DOCUMENT electronic publishing system, a software tool developed and sold by Digital. In this system, writers use an ASCII text editor to create source files containing text and English-like code; this code labels the structural elements of the document, such as chapters, paragraphs, and tables. The VAX DOCUMENT software, which runs on the VMS operating system, interprets the code to format the text, generate a table of contents and index, and paginate the entire document. Writers can print the document on the terminal or line printer, or they can use Digital-supported devices, such as the LN03 laser printer and PostScript printers (PrintServer 40 or LN03R ScriptPrinter), to produce a typeset-quality copy containing integrated graphics.

Contents

PREFACE xiii

CHAPTER 1 VAX–11 RSX INSTALLATION GUIDE 1–1

1.1 VAX–11 RSX AND VMS VERSIONS 1–1

1.2 VAX–11 RSX KIT 1–1

1.3 ONLINE RELEASE NOTES 1–2

1.4 GETTING STARTED 1–2

1.5 INSTALLING VAX–11 RSX 1–6

1.6 AFTER INSTALLATION 1–9

1.7 UPGRADING FROM PRIOR VERSIONS OF VAX–11 RSX 1–11
1.7.1 Removing the VAX–11 RSX Software 1–11
1.7.2 Installation and Startup Changes 1–11
1.7.3 Synonym Directories 1–11
1.7.4 Moving Layered Product Files 1–12

1.8 INSTALLING LAYERED PRODUCTS 1–12

1.9 VAX–11 RSX INSTALLATION EXAMPLE 1–13

1.10 VAX–11 RSX MESSAGES 1–17
1.10.1 Installation and Startup Messages 1–18
1.10.2 IVP Messages 1–21
1.10.3 RSX$REMOVE_AND_DELETE Messages 1–22

Contents

CHAPTER 2 NEW FEATURES FOR VAX–11 RSX 2–1

2.1 NEW SYSTEM DIRECTIVES—FSS$, PFCS$, AND PRMS$ 2–1

2.2 COMMON FILE CONTROL SERVICES 2–1

2.3 MCR SUPPORT FOR MOST DCL COMMANDS 2–1

2.4 R.MLB MACRO LIBRARY 2–2

2.5 DATA TERMINAL EMULATOR 2–2

2.6 PROFESSIONAL HOST COMMUNICATIONS 2–3

2.7 HELP ON MCR 2–3

2.8 RMS–11 ON VAX–11 RSX 2–3
2.8.1 Files and Placement on the Distribution Kit 2–4
2.8.2 Building RMS–11 Applications 2–4
2.8.3 RMS–11 Memory-Resident Libraries 2–6

CHAPTER 3 MODIFICATIONS TO EXISTING VAX–11 FEATURES 3–1

3.1 INSTALLATION AND STARTUP 3–1
3.1.1 New and Renamed Files 3–2
3.1.1.1 RSX$DEVICE_NAMES.COM File • **3–2**
3.1.1.2 Renamed Startup Files • **3–2**
3.1.2 Defining Device Logical Names During Installation 3–2
3.1.3 RSX$DEVICE_CACHE Logical Name 3–3
3.1.4 Contiguous Space Requirements 3–3
3.1.5 Running the IVP After Installation 3–3
3.1.6 Remove and Delete Procedure 3–4

3.2 SYNONYM DIRECTORIES 3–4

3.3	LMF SUPPORT	3–5
3.4	ACCESSING MULTIVOLUME TAPES	3–5
3.5	APPLICATIONS MIGRATION EXECUTIVE	3–5
3.5.1	Task Overlay Loading Correction	3–5
3.5.2	Asynchronous System Traps	3–6
3.5.3	Task Names and Process Names	3–6
3.5.4	Batch Process Names	3–7
3.5.5	Compatibility Mode	3–7
3.5.6	GRPNAM Privilege Required for Group-Wide Logical Names	3–8
3.5.7	SYS$ERROR Definition	3–8
3.5.8	Privileged Images	3–8
3.6	BACKUP AND RESTORE UTILITY	3–8
3.6.1	Multi-Volume Restore Correction	3–8
3.6.2	BRU /APPEND Qualifier	3–8
3.6.3	BRU /IDENTIFICATION Qualifier	3–9
3.6.4	BRU Messages	3–9
3.6.5	BRU Multiple Disk Operations	3–9
3.6.6	BRU Performance Enhancement	3–9
3.6.7	Creating User File Directories	3–9
3.7	DISK SAVE AND COMPRESS UTILITY	3–10
3.8	DEVICE AND FILE HANDLING	3–10
3.8.1	Device Mnemonics and Physical Devices	3–10
3.8.2	Device Name Logicals	3–10
3.8.3	IE.NSF Error Code	3–11
3.8.4	FCS Create and Open Operations	3–11
3.8.5	FCS Created Files	3–11
3.8.6	FCS .PARSE Routine	3–11
3.9	PERIPHERAL INTERCHANGE PROGRAM (PIP)	3–11
3.9.1	Used and Free File Header Count	3–12
3.9.2	PIP /PU Command	3–12
3.10	EXECUTIVE DIRECTIVES	3–12
3.10.1	ALUN$ Directive	3–12
3.10.2	ALUN$ and ACHN$ Directive	3–13
3.10.3	ATRG$ Directive	3–13

3.10.4	**BOM$ Directive**	**3–14**
3.10.5	**Task Size**	**3–14**
3.10.6	**GPRT$ Directive**	**3–14**
3.10.7	**GREG$ Directive**	**3–15**
3.10.8	**QIO$ Directive**	**3–15**
3.10.8.1	Logical Block Number • **3–15**	
3.10.9	**SPWN$ Directive**	**3–15**
3.11	**INDIRECT**	**3–15**
3.11.1	**Indirect Directives**	**3–16**
3.11.2	**Indirect Error Messages**	**3–17**
3.11.3	**Multiple Indirect Command Files Using MCR Commands**	**3–17**
3.11.4	**Ownership of Files and Indirect**	**3–17**
3.11.5	**Storing String Symbol Values**	**3–18**
3.11.6	**Error Deleting Global Section When Indirect Exits**	**3–18**
3.12	**LOGIN PROCEDURE**	**3–18**
3.13	**MACRO–11**	**3–18**
3.13.1	**MACRO–11 Version 5.5 Update**	**3–19**
3.13.2	**MACRO–11 Version 5.4 Update**	**3–19**
3.13.3	**MACRO–11 Version 5.3 Update**	**3–20**
3.13.4	**MACRO–11 Version 5.2 Update**	**3–20**
3.13.5	**MACRO–11 Version 5.1 Update**	**3–21**
3.13.6	**MACRO–11 Version 5.0 Update**	**3–21**
3.14	**MONITOR CONSOLE ROUTINE**	**3–23**
3.14.1	**MCR Commands**	**3–23**
3.14.2	**MCR Mode Error Messages**	**3–24**
3.14.3	**MCR Mode Informational Messages**	**3–24**
3.14.4	**SUBMIT /LOG_FILE Keyword**	**3–25**
3.15	**PDP–11 EMULATOR**	**3–25**
3.16	**RMS–11**	**3–26**
3.16.1	**Access Methods**	**3–26**
3.16.2	**RMSBCK Utility**	**3–26**
3.16.2.1	RMSBCK Corrections • **3–26**	
3.16.2.2	New RMSBCK Switches • **3–27**	
3.16.3	**RMSCNV Utility**	**3–27**
3.16.4	**RMSDEF Utility**	**3–28**
3.16.5	**Invoking RMSDEF From MCR**	**3–28**

3.16.6	**RMSDES Utility**	**3–28**
3.16.6.1	RMSDES Enhancements • **3–28**	
3.16.6.2	RMSDES Corrections • **3–29**	
3.16.7	**RMSDSP Utility**	**3–29**
3.16.8	**RMSIFL Utility**	**3–29**
3.16.9	**RMSRST Utility**	**3–30**
3.16.9.1	RMSRST Switches Added • **3–31**	
3.16.9.2	RMSRST Corrections • **3–31**	
3.16.10	**Additional Corrections to Utilities**	**3–32**
3.16.11	**Wildcard Characters in File Specifications**	**3–32**
3.16.12	**Documentation Corrections**	**3–32**
3.16.12.1	RSX–11M/M–PLUS RMS–11 User's Guide • **3–32**	
3.16.12.2	RSX–11M/M–PLUS RMS–11 Macro Programmer's Guide • **3–33**	
3.16.12.3	RSX–11M/M–PLUS RMS–11 Utilities Manual • **3–34**	
3.17	**VAX–11 RSX SYSTEM LIBRARY**	**3–38**
3.17.1	**High-Level Language Interface Routines**	**3–38**
3.17.2	**Named Directory Support**	**3–38**
3.17.3	**A.JUMP and A.MODE Symbols**	**3–38**
3.18	**TASK BUILDER**	**3–38**
3.18.1	**Building Drivers**	**3–39**
3.18.2	**Contiguous Space Requirements for TKB**	**3–39**
3.18.3	**Linking to Libraries**	**3–39**
3.18.4	**New Error Messages**	**3–39**
3.18.5	**TKB /FO Switch**	**3–39**
3.18.6	**OTS Fast Map Routine**	**3–39**
3.18.7	**RNDSEG Option**	**3–40**
3.18.8	**Specifying a Task Partition**	**3–40**
3.19	**TERMINAL DRIVER**	**3–40**
3.19.1	**IO.ATA Function**	**3–40**
3.19.2	**TC.RAT Function Correction**	**3–41**
3.19.3	**Terminal Characteristics for SF.GMC and SF.SMC Requests**	**3–42**
3.20	**NEW VAX–11 RSX ERROR MESSAGES**	**3–42**

Contents

CHAPTER 4 VAX–11 RSX RESTRICTIONS 4–1

4.1	MCR SET UIC	4–1
4.2	TASK BUILDER	4–1
4.2.1	Building Memory-Resident Overlays in a Co-Tree	4–1
4.2.2	OTS Fast Map Restrictions	4–2
4.3	BACKUP AND RESTORE UTILITY	4–2
4.3.1	Multidisk Restore Operation	4–2
4.3.2	APPEND Qualifier	4–2
4.3.3	LENGTH Qualifier	4–3
4.4	INDIRECT	4–3
4.4.1	Stopping Indirect Command Procedures	4–3
4.4.2	Logical Name Handling	4–3
4.4.3	Process Context	4–4
4.4.4	Differences Between VAX–11 RSX and RSX–11M–PLUS Indirect	4–4
4.5	LOGIN COMMAND FILES	4–5
4.5.1	LOGIN.CMD File Nesting	4–6
4.5.2	System Login Command File	4–6
4.6	MEMORY MANAGEMENT ON VAX–11 RSX AND RSX–11	4–6
4.6.1	Compatibility Between VAX–11 RSX and RSX–11 Memory Management	4–6
4.6.2	Creating Regions	4–6
4.6.3	Deleting Regions	4–7
4.6.4	Displaying Region Information	4–7
4.6.5	Region Checkpointing	4–8
4.7	DEVICE NAMES	4–8
4.7.1	Passing Device Names to Other Tasks	4–8
4.7.2	RSX–11 Device Names	4–9
4.8	PIP WILDCARD CHARACTERS	4–10

4.9	RMS–11	4–10
4.9.1	RMSCNV Utility	4–10
4.9.2	RMSDES Utility	4–10
4.9.3	RMSIFL Utility	4–10
4.9.4	Using QIOs on Assigned LUNs	4–11
4.10	SHARING DATA BETWEEN TASKS	4–11
4.11	SPAWN DIRECTIVE	4–12
4.11.1	Spawning DCL With a Command Line	4–12
4.11.2	Spawning MCR With a Command Line	4–12
4.11.3	Spawning MCR With DEFCLI Set	4–12
4.12	SYSCOMMON SYSTEM DISKS	4–13
4.13	RSX–11M–PLUS VERSION 4.3 SYSTEM GENERATION AND THE TASK BUILDER	4–13
4.14	RSX–11 SYSGEN AND NETGEN	4–14
4.15	REDEFINING THE DEFAULT DEVICE	4–14
4.16	UNDOCUMENTED ERROR CODES	4–14
APPENDIX A	REPORTING PROBLEMS	A–1
APPENDIX B	THE DIGITAL EQUIPMENT COMPUTER USERS SOCIETY (DECUS)	B–1
B.1	DECUS GOALS	B–1
B.2	DECUS ACTIVITIES	B–1

INDEX

TABLES

1–1	VMSTAILOR Classes	1–2
1–2	VAX–11 RSX Global Page/Section Requirements	1–4
1–3	Disk Block Requirements	1–5
1–4	Layered Products Requiring Special Installation Procedures	1–13
3–1	RSX–11 Device Mnemonics	3–10
3–2	TC.RAT Functions	3–41
3–3	Terminal Characteristics for SF.GMC and SF.SMC Requests	3–42

Preface

Manual Objectives

The *VAX–11 RSX Version 2.5 Installation Guide and Release Notes* describes the installation, new features, and changes for VAX–11 RSX Version 2.5. In addition, this manual provides cumulative information from previous release notes for VAX–11 RSX Version 2.0 to Version 2.5.

Intended Audience

This manual is intended for system users familiar with both VMS and RSX–11 operating systems.

Structure of This Manual

This manual contains the following chapters:

- Chapter 1 describes how to install the VAX–11 RSX Version 2.5 software.
- Chapter 2 describes major new features of VAX–11 RSX Version 2.0 to Version 2.5.
- Chapter 3 describes corrections and modifications to existing VAX–11 RSX software features from Version 2.0 to Version 2.5.
- Chapter 4 describes restrictions on the use of VAX–11 RSX Version 2.5.
- Appendix A describes when and how to report software problems to DIGITAL by submitting Software Performance Reports (SPRs).
- Appendix B briefly describes the goals and activities of the DIGITAL Equipment Computer Users Society (DECUS).

Technical Changes for VAX–11 RSX Version 2.5

The information in Chapter 1 of this manual is specific to this release of VAX–11 RSX. Chapters 2, 3, and 4 describe cumulative software changes from VAX–11 RSX Version 2.0 to Version 2.5.

New Features (Chapter 2)

In addition to the features added cumulatively to VAX–11 RSX since Version 2.0, Chapter 2 describes the following major new features for Version 2.5:

- New system directives—The FSS$, PFCS$, and PRMS$ system directives available on RSX–11M–PLUS are now supported by the Application Migration Executive (AME) in VAX–11 RSX.
- Common FCS—A File Control Services (FCS) common with RSX–11M–PLUS and VAX CoProcessor/RSX. This lets you run tasks built on any of these systems without rebuilding them.

- MCR support for most DCL commands—The MCR CLI now uses the same command table used by DCL (DCLTABLES). Previously, MCR used a separate command table (MCRTABLES) and supported a small subset of DCL commands that were updated only with new releases of VAX–11 RSX.

Modifications (Chapter 3)

In addition to cumulative changes to VAX–11 RSX since Version 2.0, Chapter 3 desribes the following modifications for Version 2.5:

- AST delivery—A window during which ASTs would not be delivered has been closed.
- Task and process names—A clarification of how the system generates task and process names is given.
- The GRPNAM privilege is required to create or delete logical names in the group logical name table
- Leading zeros are dropped from logical device names
- UFD creation by the Backup and Restore Utility—The Backup and Restore Utility (BRU) has been modified to allow it to create UFDs on Files–11 Structure Level 2 disk volumes.
- Restoring multi-volume disk savesets—This BRU operation now works correctly with all devices.
- Multivolume tapes to be accessed by compatibility mode utilities must be mounted with the /MULTI_VOLUME qualifier
- ATRG$ directive—The ATRG$ directive now functions correctly with memory-resident overlays in shared task regions.
- Task overlay loading correction—Tasks which are built with both memory-resident and disk-resident overlays now load correctly.
- The GPRT$, GREG$, and GTSK$ directives return the task size in terms of 32-word blocks.
- MCR DEBUG command—The MCR DEBUG command now works correctly on VAX systems without compatibility mode hardware.
- Indirect—An error deleting a global section when Indirect exits has been corrected.
- RMSDEF—The MCR DEF command is now the equivalent of the DCL DEFINE command. You may run the RMSDEF Utility from MCR by entering >RUN SYS$SYSTEM:DEF from MCR.
- RMSIFL—The RMS Utility RMSIFL has several new corrections. These are described in Chapter 3.
- RMS–11 $CREATE, $OPEN, and $CONNECT directives—These directives no longer fail if the IFI field in the FAB or ISI field in the RAB contain a non-zero value. These are described in Chapter 3.
- The Disk Save and Compress Utility (DSC1) is no longer included with VAX–11 RSX.

Restrictions (Chapter 4)

Chapter 4 describes the following new restrictions on the use of VAX–11 RSX for Version 2.5:

- Installing over previous versions of VAX–11 RSX—If you plan to upgrade an existing version of VAX–11 RSX, Version 2.5 can only be installed over VAX–11 RSX Version 2.4.

  Versions prior to 2.4 must be removed before the new installation. A procedure to remove VAX–11 RSX files is described in Chapter 1.

- MCR SET /UIC—Use of the MCR syntax for the SET /UIC=[g,m] command requires a VMS V5.4 or later system. RSX–11M, RSX–11S, and RSX–11M–PLUS system and network generations also require a VMS V5.4 or later system because they use the MCR SET /UIC=[g,m] command.

- RSX logical names SY and WK—The RSX logical names SY and WK are defined to be SYS$DISK (the device to which your default is currently set). Although you can redefine WK to be a valid device, SY is defined internally in the RSX AME and cannot be changed.

  This is not a new restriction, but a clarification of VAX–11 RSX behavior.

- RSX–11M–PLUS Version 4.3 System Generation —

  Before performing an RSX–11M–PLUS Version 4.3 System Generation on VAX–11 RSX, a modified version of the Task Builder (TKB) supplied with VAX–11 RSX Version 2.5 must be copied to the System Generation disk. Please refer to Section 4.13 for information about this procedure.

Associated Manuals

The following manuals provide more information on VAX–11 RSX installation and system operation procedures:

- The *VAX–11 RSX Compatibility Mode Reference Manual*
- The *VMS System Manager's Manual*

The following conventions are observed in this manual:

Convention	Meaning
>	A right angle bracket is the explicit prompt of the Monitor Console Routine (MCR), which is the command interpreter provided by VAX–11 RSX. The MCR prompting character appears whenever control is returned to the compatibility mode terminal and you can type input.

Convention	Meaning
$	A dollar sign followed by a space is the explicit prompt of the VMS DIGITAL Command Language (DCL), the default command interface on VMS systems. Whenever control is returned to the native mode terminal and you can type input, this prompt appears.
UPPERCASE	Uppercase letters in a command line indicate letters that must be entered as they are shown. For example, utility switches must always be entered as they are shown in format specifications.
command abbreviations	Where short forms of commands are allowed, the shortest form acceptable is represented by uppercase letters. The following example shows the minimum abbreviation allowed for the DCL command DIRECTORY: `$ DIR`
LOWERCASE	Any command in lowercase must be substituted for. Usually the lowercase word identifies the kind of substitution expected, such as a filespec, which indicates that you should fill in a file specification. For example: `filename.filetype;version` This command indicates the values that comprise a file specification; values are substituted for each of these variables as appropriate.
/keyword, /qualifier, or /switch	A command element preceded by a slash (/) is an MCR keyword; a DCL qualifier; or a task, utility, or program switch. Keywords, qualifiers, and switches alter the action of the command they follow.
parameter	Required command fields are generally called parameters. The most common parameters are file specifications.
[option]	Square brackets indicate optional entries in a command line or a file specification. If the brackets include syntactical elements, such as periods (.) or slashes (/), those elements are required for the field. If the field appears in lowercase, you are to substitute a valid command element if you include the field. Note that when an option is entered, the brackets are not included in the command line.
[, . . .]	Square brackets around a comma and a horizontal ellipsis indicate that you can use a series of optional elements separated by commas. For example, (argument [, . . .]) means that you can specify a series of optional arguments by enclosing the arguments in parentheses and by separating them with commas.
{ }	Braces indicate a choice of required options. You are to choose from one of the options listed.
:argument	Some parameters and qualifiers can be altered by the inclusion of arguments preceded by a colon. An argument can be either numerical (COPIES:3) or alphabetical (NAME:QIX). In DCL, the equal sign (=) can be substituted for the colon to introduce arguments. COPIES=3 and COPIES:3 are the same.

Convention	Meaning
( )	Parentheses are used to enclose more than one argument in a command line. For example: `SET PROT = (S:RWED,O:RWED)`
,	Commas are used as separators for command line parameters and to indicate positional entries on a command line. Positional entries are those elements that must be in a certain place in the command line. Although you might omit elements that come before the desired element, the commas that separate them must still be included.
[g,m] [directory]	The convention [g,m] signifies a User Identification Code (UIC). The g is a group number and the m is a member number. The UIC identifies a user and is used mainly for controlling access to files and privileged system functions. This may also signify a User File Directory (UFD), commonly called a directory. A directory is the location of files. Other notations for directories are: [ggg,mmm], [gggmmm], [ufd], [name], and [directory]. The convention [directory] signifies a directory. Most directories have 1- to 9-character names, but some are in the same [g,m] form as the UIC. Where a UIC, UFD, or directory is required, only one set of brackets is shown (for example, [g,m]). Where the UIC, UFD, or directory is optional, two sets of brackets are shown (for example, [[g,m]]).
filespec	A full file specification includes device, directory, file name, file type, and version number, as shown in the following example: `DL2:[46,63]INDIRECT.TXT;3` Full file specifications are rarely needed. If you do not provide a version number, the highest numbered version is used. If you do not provide a directory, the default directory is used. Some system functions default to particular file types. Many commands accept a wildcard character (*) in place of the file name, file type, or version number. Some commands accept a filespec with a DECnet node name.
.	A period in a file specification separates the file name and file type. When the file type is not specified, the period may be omitted from the file specification.
;	A semicolon in a file specification separates the file type from the file version. If the version is not specified, the semicolon may be omitted from the file specification.
@	The at sign invokes an indirect command file. The at sign immediately precedes the file specification for the indirect command file, as follows: `@filename[.filetype;version]`

Convention	Meaning
. . .	A horizontal ellipsis indicates the following: • Additional, optional arguments in a statement have been omitted. • The preceding item or items can be repeated one or more times. • Additional parameters, values, or other information can be entered.
. . .	A vertical ellipsis shows where elements of command input or statements in an example or figure have been omitted because they are irrelevant to the point being discussed.
KEYNAME	This typeface denotes one of the keys on the terminal keyboard, for example, the RETURN key.
CTRL/x	The symbol CTRL/x means that you are to press the key marked CTRL while pressing another key. Thus, CTRL/Z indicates that you are to press the CTRL key and the Z key together in this fashion. CTRL/Z is echoed on some terminals as ^Z. However, not all control characters echo.
n	A lowercase n; indicates a variable for a number.
black ink	In examples, what the system prints or displays is printed in black.
red ink	In interactive examples, what the user types is printed in red. System responses appear in black.
xxx	A symbol with a 1- to 3-character abbreviation, such as x or RET, indicates that you press a key on the terminal. For example, RET indicates the RETURN key, LF indicates the LINE FEED key, and DEL indicates the DELETE key.

1 VAX–11 RSX Installation Guide

VAX–11 RSX is a VMS layered product that allows you to execute RSX–11 tasks on VAX processors. This chapter describes how to install, verify, and maintain the VAX–11 RSX Version 2.5 software on the VMS Version 5.0 or later operating system. It also documents changes to the installation and startup of VAX–11 RSX from Versions 2.0 to 2.5.

1.1 VAX–11 RSX and VMS Versions

VAX–11 RSX Version 2.5 will run only on Version 5.0 and later versions of the VMS operating system.

VMS Version 5.4 is required for the use of the MCR SET /UIC command. RSX–11M, RSX–11S, and RSX–11M–PLUS system and network generations also require a VMS V5.4 or later system because they use the MCR SET /UIC command.

The current installation kit lets you upgrade VAX–11 RSX Version 2.4 software to Version 2.5. However, if your current VAX–11 RSX software is Version 2.3 or earlier, you must delete your VAX–11 RSX files before performing a new VAX–11 RSX Version 2.5 installation. Section 1.7 describes how to upgrade from VAX–11 RSX Version 2.3 (or earlier) to Version 2.5.

1.2 VAX–11 RSX Kit

VAX–11 RSX Version 2.5 is distributed on the following types of media:

- CDROM (as part of the VMS Consolidated Software Distribution)
- Magnetic tape (1600 bpi)
- TK50 tape cartridges

Each distribution kit contains three files that are backup save sets. The following table describes each save set and its contents:

Save Set	Contents
RSX025.A	Contains the startup command procedures and the command files needed to perform the software installation.
RSX025.B	Contains the task images, libraries, other files that constitute VAX–11 RSX Version 2.5, and the Installation Verification Procedure (IVP).
RSX025.C	Contains the VAX–11 RSX PDP–11 Instruction Set Emulator. This save set is installed on systems that do not have compatibility mode hardware or microcode.

1.3 Online Release Notes

The distribution kit contains a machine-readable copy of this manual, the *VAX–11 RSX Version 2.5 Installation Guide and Release Notes*, which is copied to directory SYS$HELP during the installation process.

You can display or print the release notes before performing the installation by entering the following command:

```
$ @SYS$UPDATE:VMSINSTAL RSX025 ddcu: OPTIONS N
```

In the preceding command line, ddcu: is the drive in which you have placed the VAX–11 RSX distribution kit.

1.4 Getting Started

Before you install VAX–11 Version 2.5, take the time to read this section and perform the following preliminary steps:

1 If you have used the VMSTAILOR utility to tailor off any of the VMS classes, be sure your system has the VMS classes you need to use the VAX–11 RSX product.

You can run compatibility mode tasks under VAX–11 RSX without any of the tailorable VMS classes. However, certain features of VAX–11 RSX require some of the tailorable VMS classes, as described in Table 1–1.

Table 1–1 VMSTAILOR Classes

Class	Subclass	Description
System Programming Support	Files–11 ODS1 ACP	Required in order to access RSX-compatible disks (Files–11 On-Disk Structure Level 1)
	Print and Batch Queue Utilities	Required in order to spool listing files from the VAX–11 RSX utilities (by using the /SP switch).

2 Be sure that the values in the SYSTEM account authorization record are equal to or greater than the default values in the following list:

Buffered byte count (BYTLM)	=	20480
Enqueued quota (ENQLM)	=	20
Direct I/O limit (DIOLM)	=	12
Buffered I/O limit (BIOLM)	=	12
Open file quota (FILLM)	=	20
Asynchronous System Trap (AST) limit (ASTLM)	=	20

Also, make sure that the SYSTEM account does not specify the DEFCLI option. When set, this option causes the IVP to fail, because it prevents the user of the SYSTEM account from logging in under any command interpreter (CLI) except the DIGITAL Command Language (DCL).

To check these limits, run the VMS Authorize Utility (AUTHORIZE) by entering the following commands:

```
$  SET DEFAULT SYS$SYSTEM
$  RUN AUTHORIZE
```

The system responds with the User Authorization File (UAF) prompt UAF>. At the prompt, enter the following command:

```
UAF> SHOW SYSTEM
```

AUTHORIZE displays the current values of the SYSTEM account's authorization record.

You can change these values by entering the MODIFY command in response to the UAF> prompt as shown in the following command line:

```
UAF> MODIFY SYSTEM/limit=new-value
```

For example:

```
UAF> MODIFY SYSTEM/DIOLM=12
```

To modify the DEFCLI option, enter the following command in response to the UAF> prompt:

```
UAF> MODIFY SYSTEM/FLAGS=NODEFCLI
```

Reenter the SHOW command to verify that the new values have been set, as follows:

```
UAF> SHOW SYSTEM
```

To exit from AUTHORIZE, enter the EXIT command at the UAF> prompt. You must then log out and log in again for the changes to take effect.

3 If you are installing VAX–11 RSX Version 2.5 on a VAX processor that is not part of a VAXcluster, be sure that your processor has enough unused global pages and global sections. Table 1–2 specifies the number of global pages and global section required for your installation.

If you are installing VAX–11 RSX Version 2.5 on a VAX processor that is part of a VAXcluster, be sure that each node in the cluster that will be running VAX–11 RSX has enough unused global pages and global sections. Table 1–2 displays these requirements.

The following VAX processors have compatibility mode hardware. When VAX–11 RSX runs on these processors it does not need or use the VAX–11 RSX PDP–11 Instruction Set Emulator. Other VAX processors do not have compatibility mode hardware; they use the emulator when running VAX–11 RSX.

VAX–11/725	VAX–11/785
VAX–11/730	VAX 8600
VAX–11/750	VAX 8650

VAX–11/780

Note: If your VAX processor does not support compatibility mode hardware, refer to the "With Emulator" column in Table 1–2. If your VAX processor supports compatibility mode hardware, refer to the "Without Emulator" column in Table 1–2.

Table 1–2 VAX–11 RSX Global Page/Section Requirements

Global Configuration Requirements	With Emulator	Without Emulator
Unused global pages	3800	100
Unused global sections	12	10

To determine the number of unused global pages on your system, invoke the Install Utility (INSTALL) by entering the following VMS DCL commands:

```
$  INSTALL
INSTALL> LIST/GLOBAL/SUMMARY
```

Read the summary line of the screen display and note the following:

- The number of global sections *used*
- The number of global pages *unused*

To determine the unused number of global sections, invoke the VMS System Generation Utility (SYSGEN) as follows:

```
$  RUN SYS$SYSTEM:SYSGEN
SYSGEN> SHOW GBLSECTIONS
```

The number in the first column of the SYSGEN display is the current number of global sections. Subtract the number of used global sections (displayed by the INSTALL listing) from the current number of global sections (shown by the SYSGEN display). The difference is the unused number of global sections.

Once you know how many unused global pages and global sections are available, you can modify the necessary parameters through AUTOGEN.COM (a DIGITAL-supplied command procedure located in the SYS$UPDATE directory) by using the following procedure :

a. If the number of unused global sections is fewer than 10 (12 on VAX processors without compatibility mode hardware) and the number of unused global pages is fewer than 100 (3800 on VAX processors without compatibility mode hardware), you will have to increase the GBLPAGES and GBLSECTIONS parameters.

To modify the GBLPAGES and GBLSECTIONS parameter, edit the file SYS$SYSTEM:MODPARAMS.DAT to include the following:

```
ADD_GBLPAGES=100
ADD_GBLSECTIONS=10
```

b. If you have modified either parameter (GBLPAGES or GBLSECTIONS) you must run AUTOGEN for the new value to take effect. Enter the following command:

```
$ @SYS$UPDATE:AUTOGEN SAVPARAMS REBOOT NOFEEDBACK
```

These parameters are not dynamic, so your system will need to be rebooted. AUTOGEN will perform an automatic shutdown and will reboot when the shutdown has completed.

4 Remove all unwanted or redundant files from the appropriate media to make sure there are enough free blocks to install VAX–11 RSX.

The following summary in Table 1–3 indicates the approximate disk block utilization requirements for VAX–11 RSX Version 2.5:

Table 1–3 Disk Block Requirements

Disk Block Utilization	With Emulator	Without Emulator
Peak disk block utilization:	12,200 blocks	8500 blocks
Net disk block utilization:	10,700 blocks	7000 blocks

A significant number of system disk blocks will contain the work files required for the installation and the files to be read from the distribution volume. The number of blocks used during the installation is called the *peak disk block utilization*. DIGITAL recommends that you make enough blocks available on the system disk to cover peak utilization. If you choose not to do so, the VMSINSTAL procedure will operate in an alternate mode that reduces the peak utilization.

If a system failure occurs in the alternate mode, the installation will fail. To recover from this system failure, provide more free blocks and begin the installation again.

When the installation is complete, the system disk returns to its normal state. However, many disk blocks are depleted as a result of the installation process. The number of used disk blocks will be greater than the number in use before you began the installation. This number is called the *net disk block utilization*.

After you rebootstrap the system, you can recover additional disk blocks by purging system files that the VMSINSTAL procedure cannot purge during the installation.

Confirm the free block count by entering the following DCL command:

```
$ SHOW DEVICE SYS$SYSDEVICE
```

5 Use the VMS Backup Utility (BACKUP) to back up your system disk.

A system failure at a critical point during the installation may result in unusable files. Therefore, you should always back up your system disk before you attempt any software installation. You should use the copy created from the backup operation for the installation. For more information on BACKUP, see the *VMS Backup Utility Manual*.

6 Log in at the console terminal under the system manager (SYSTEM) account.

7 Be sure that all users are logged out and that all batch jobs are completed by entering the following commands:

```
$  SHOW USERS
   .
   .
   .
$  SHOW QUEUE/BATCH
   .
   .
   .
```

8 Enter the following command to prevent users from gaining access to the system:

```
$  SET LOGINS/INTERACTIVE=0
```

9 Make sure DECnet-VAX is turned off. If you are not sure whether DECnet-VAX is running on your system, enter the following command:

```
$  SHOW NETWORK
```

If the message "Network unavailable" appears, proceed to Step 10. If DECnet-VAX is running, turn it off by entering the following command:

```
$  RUN SYS$SYSTEM:NCP
```

The system responds with the Network Control Program (NCP) prompt NCP>. At the prompt, enter the following commands:

```
NCP> SET EXECUTOR STATE SHUT
NCP> EXIT
$
```

10 Register the Product Authorization Key (PAK) for VAX–11 RSX by entering the following command from the system manager account:

```
$  @SYS$UPDATE:VMSLICENSE
```

The VMSLICENSE command procedure will prompt you for information from your PAK and register your PAK with the VMS License Management Facility (LMF).

See the *VMS License Management Utility Manual* for more information about license registration and management.

1.5 Installing VAX–11 RSX

If you successfully completed all the steps in Section 1.4, you are ready to install VAX–11 RSX Version 2.5. The installation takes approximately 30 minutes to complete. At any time during the execution of the VMSINSTAL procedure, you can type a question mark (?) for help. Perform the installation as follows:

1 Log in at the console terminal under the system manager (SYSTEM) account.

2 Enter the following command line to initiate the installation:

```
$  @SYS$UPDATE:VMSINSTAL RSX025 ddcu:
```

In this command line, ddcu is the drive in which you have placed the VAX–11 RSX distribution kit.

3 After you respond to a series of questions, the installation procedure copies the files and then verifies that the VAX–11 RSX utilities are working.

Before copying the files, answer the following question:

```
* Are you satisfied with the backup of your system disk [YES]?
```

Assuming that you have performed the backup as recommended in Section 1.4, step 5, press the RETURN key (YES is the default response).

Next, the VMSINSTAL procedure prompts you as follows:

```
Please mount the first volume of the set on ddcu:

* Are you ready?
```

Physically mount the distribution software in the appropriate drive (if you have not already done so) and then answer YES.

A prompt will appear to request each additional volume. Enter YES after you insert each volume. When all the volumes have been copied, remove the last volume from the drive.

The VMSINSTAL procedure prompts you as follows:

```
Please select one of the following actions:


   1)  Perform the installation
   2)  Create a file with a description of the installation
   3)  Both of the above


* What would you like to do [3]:
```

Choose option 1 if you want to perform the installation without creating a file that contains a description of the installation.

Choose option 2 if you do not want to perform the installation, but want the description file, SYS$UPDATE:RSX025.TXT.

Choose option 3 (the default) if you want both to perform the installation and create a description file.

If you choose either option 1 or 3, the VMSINSTAL procedure responds with the following questions:

```
* Do you want to install the MCR help library [YES]?
```

This library supplies help for VAX–11 RSX Monitor Console Routine (MCR) commands and for the RSX–11 utilities supplied with VAX–11 RSX. The library requires 2000 additional disk blocks. (These blocks

are included in the disk block utilization requirements shown in Table 1–3.) If you want the library, answer YES.

```
* Do you want to purge files replaced by this installation [YES]?
```

If you do not have sufficient free disk space or you do not want the old copies of the files, answer YES.

```
* Does this product have an authorization key registered and
  loaded?
```

If you have not registered and loaded an authorization key for VAX–11 RSX, as described in Section 1.4, Step 10, you should answer NO to this question. You will not be asked whether you want to run the Installation Verification Procedure (IVP), since the IVP cannot be run if the authorization key is not registered and loaded.

If you have registered and loaded an authorization key for VAX–11 RSX, as described in Section 1.4, Step 10, you should answer YES to this question.

Next, the VMSINSTAL procedure asks a question about the IVP:

```
* Do you want to run the IVP after the installation [YES]?
```

The IVP tests the various VAX–11 RSX components to make sure that the components were installed correctly and are working properly. The IVP is not meant to be a comprehensive test of all components, but does test some of the features of the components. For example, it tests the Librarian Utility Program (LBR) by having it create an object library, extract a module from that library, delete the module in the library, and then reinsert the extracted module into the library.

At this point, you have answered all the questions necessary to install VAX–11 RSX successfully. The VMSINSTAL procedure will complete your VAX–11 RSX Version 2.5 installation. For more information about the VMSINSTAL procedure, see the *Guide to VMS Software Installation*.

4 When the IVP is testing the Source Language Input Program (SLP), the following messages appear:

```
%RSX-I-TESTSTART, SLP testing started at 10:17:18
%RSX-S-TESTSUCCESS, SLP testing completed successfully at
 10:17:27
%RSX-S-IVPSUCCESS, the installation verification of VAX--11 RSX
 Version 2.5 succeeded
```

Various compatibility mode images tested during the IVP may produce status messages that are not in standard VMS format. However, if an error occurs, the IVP will detect it and produce the error message in the standard VMS format.

Refer to Section 1.10 of this document if an error message is displayed during the IVP. Section 1.10 describes each error message and how to correct the error.

5 For a complete description of the new images and files that VAX–11 RSX uses (if you chose to create a description file during installation), please read the SYS$UPDATE:RSX025.TXT file.

1.6 After Installation

This section lists the steps that you should follow after the installation of VAX–11 RSX.

1 Read the console listing carefully. Look for warning or error messages that indicate steps you must perform manually. Many informational messages will also be displayed. Be sure to read them.

During the installation, a machine-readable copy of the VAX–11 RSX Release Notes (which you can display or print) is put in directory SYS$HELP.

2 RSX$STARTUP.COM, which is placed in the SYS$STARTUP directory during installation, installs VAX–11 RSX each time your system is bootstrapped and establishes the environment in which VAX–11 RSX works.

You should modify your system startup procedure to invoke the RSX$STARTUP procedure when the system boots, as follows:

a. Examine the file SYS$MANAGER:SYSTARTUP_V5.COM, and insert the following command:

```
$ @SYS$STARTUP:RSX$STARTUP.COM [spool_device]
```

In the above command, the optional spool device parameter defines the logical device names SP and SP0. If this parameter is not specified, these logical device names are defined as LPA0.

b. Remove the following line if it exists (from a prior installation of VAX–11 RSX):

```
$ @SYS$MANAGER:VAX11RSX.COM [spool_device]
```

3 The installation procedure also creates the command file RSX$DEVICE_NAMES.COM in the SYS$STARTUP directory (if the file did not already exist from a previous installation of VAX–11 RSX).

The RSX$DEVICE_NAMES.COM file defines the following logical names:

- $$n logical names for the disk and magnetic tape devices that were configured at the time of installation
- $$n names for SYS$SPECIFIC and SYS$COMMON to allow references to them
- RSX$DEVICE_CACHE, which restricts the usable range of $$n names to improve performance

RSX$DEVICE_NAMES.COM is invoked by the RSX$STARTUP.COM command procedure. You can modify it to do any of the following:

- Add additional $$n names for devices or rooted directories

- Delete $$n names that are not needed
- Modify the $$n name unit numbers to improve clarity or efficiency (by giving the lowest unit numbers to the most frequently used devices)
- Modify the definition of the RSX$DEVICE_CACHE logical name (See the *VAX–11 RSX Compatibility Mode Reference Manual* and Section 3.1.3 of this manual for more information about the RSX$DEVICE_CACHE logical name.)

4 You can include the systemwide page-length value that RSX–11 utilities (for example, the Task Builder [TKB] and the PDP–11 MACRO–11 Assembler [MAC]) use to determine the number of lines per page for a printed listing. If you want to specify this value, include the following line in the definition table of the SYSTARTUP_V5.COM file:

```
$ DEFINE/SYSTEM/EXECUTIVE_MODE SYS$LP_LINES [page_length_value]
```

SYS$LP_LINES is used by the $PGLEN run-time library routine, which computes the default number of lines on a printer page. This routine can be used by any task that paginates and produces listing files.

The new page-length value range for SYS$LP_LINES is 30 to 99 lines per page. If you do not specify a value, the page-length value will be 66 lines per page by default.

5 If you are installing VAX–11 RSX on a cluster, log in to the system manager's account and perform the following steps on each of the other processors in the cluster before you attempt to use VAX–11 RSX on those processors:

a. Enter the following command on each processor to invoke the VAX–11 RSX startup command procedure:

```
$  @SYS$STARTUP:RSX$STARTUP.COM
```

b. Enter the following command on each processor to load the license for VAX–11 RSX:

```
$  LICENSE LOAD RSX
```

6 Establish minimum user account quotas for individual users (for example, set quotas for open file, paging file, working set size, and subprocess creation).

7 Perform an incremental backup of the system disk.

8 If you did not run the Installation Verification Program (IVP) for the VAX–11 RSX product during the installation, you can do so after the installation.

DIGITAL strongly recommends that you run the IVP on all the nodes in your cluster. You can only run the IVP on one node at a time. Run the IVP by entering the following command from each node's system manager account :

```
$  @SYS$TEST:RSX$IVP
```

1.7 Upgrading from Prior Versions of VAX–11 RSX

If you are installing VAX–11 RSX for the first time or are upgrading from Version 2.4, these considerations do not apply to you and you may skip the remainder of this section.

The current installation kit lets you upgrade VAX–11 RSX Version 2.4 software to Version 2.5. However, if your current VAX–11 RSX software is Version 2.3 or earlier, you must delete your VAX–11 RSX files, as described in the Section 1.7.1. You can then then perform a new VAX–11 RSX Version 2.5 installation.

In addition to describing how to remove your old VAX–11 RSX files, this section discusses other important issues you need to consider when upgrading from versions of VAX–11 RSX prior to Version 2.4.

1.7.1 Removing the VAX–11 RSX Software

VAX–11 RSX includes the RSX$REMOVE_AND_DELETE.COM procedure to help you remove VAX–11 RSX files easily. This procedure removes installed images, synonym directories, and logical names assigned by the VAX–11 RSX startup procedure and deletes all files created by the installation of VAX–11 RSX.

If the remove and delete procedure is not in SYS$MANAGER, you can copy it from the VAX–11 RSX distribution kit by typing the following command:

```
$  BACKUP ddcu:RSX025.A/SAVE_SET/SELECT=RSX$REMOVE_AND_DELETE.COM -
   SYS$COMMON:[SYSMGR]
```

In the preceding command line, ddcu is the drive in which you have placed the VAX–11 RSX distribution kit.

Run the remove and delete procedure by typing the following command:

```
$  @SYS$MANAGER:RSX$REMOVE_AND_DELETE
```

1.7.2 Installation and Startup Changes

See Section 3.1 for information about the changes that have been made to the VAX-11 RSX installation and startup procedures. These changes include new names for the startup and $$n device name files.

1.7.3 Synonym Directories

If you are upgrading from a VAX–11 RSX Version 2.3 or earlier system, you should be aware that VAX–11 RSX no longer creates synonym directories for the VMS directories [SYSLIB], [SYSMSG], and [SYSEXE].

Section 3.2 in Chapter 3 discusses this change in detail, and describes how to modify code and command procedures that refer to files in these directories.

Note that you should move your compatibility mode libraries, files, and tasks from the old VMS directories to the new VAX–11 RSX directories.

1.7.4 Moving Layered Product Files

If you are upgrading from a VAX–11 RSX Versions 2.3 or earlier system, and you have layered products that use VAX–11 RSX installed on your system, run the RSX$MOVE_LP_FILES.COM procedure after installing VAX–11 RSX Version 2.5 to rename the layered product files from the VMS directories where they were located in the past to the new VAX–11 RSX directories.

You can copy the RSX$MOVE_LP_FILES.COM procedure from the VAX–11 RSX distribution kit by typing the following command:

```
$  BACKUP ddcu:RSX025.A/SAVE_SET -
   /SELECT=(RSX$MOVE_LP_FILES.COM,RSX$LP_FILES.DAT) SYS$COMMON:[SYSMGR]
```

In the preceding command line, ddcu is the drive in which you have placed the VAX–11 RSX distribution kit.

Run the RSX$MOVE_LP_FILES.COM procedure by typing the following command:

```
$  @SYS$MANAGER:RSX$MOVE_LP_FILES
```

1.8 Installing Layered Products

Synonym directories were removed in VAX-11 RSX Version 2.4. Hence, the installation of older versions of some layered products will fail. Table 1–4 lists the layered products and versions affected by the removal of synonym directories.

If you plan to install one of the layered products listed in the table after you have installed VAX-11 RSX Version 2.5, you must follow special instructions described in the RSX$MOVE_LP_FILES.COM procedure.

Copy the RSX$MOVE_LP_FILES.COM procedure from the VAX-11 RSX distribution kit by entering the following command:

```
$  BACKUP ddcu:RSX025.A/SAVE_SET -
_$  /SELECT=(RSX$MOVE_LP_FILES.COM,RSX$LP_FILES.DAT) SYS$COMMON:[SYSMGR]
```

In the preceding command line, ddcu is the drive in which you have placed the VAX-11 RSX distribution kit.

Print the RSX$MOVE_LP_FILES.COM procedure and follow the instructions therein for the product you are installing.

Table 1–4 Layered Products Requiring Special Installation Procedures

Layered Product	Version
FORTRAN IV/VAX to RSX	Version 2.7
MicroPower/Pascal-VMS	Version 2.4
PDP-11 DATATRIEVE/VAX	Version 3.2
PDP-11 FORTRAN-77/VAX to RSX	Version 5.2
PDP-11 SYMBOLIC DEBUGGER/VAX to RSX	Version 2.0-0
VAX-11 RTEM	Version 2.3

1.9 VAX–11 RSX Installation Example

This section shows the text of a typical VAX–11 RSX Version 2.5 installation on a Version 5.0 or later VMS system.

If VAX–11 RSX Version 2.4 already exists on your system, the installation procedure allows you to upgrade to Version 2.5. However, if VAX–11 RSX Version 2.3 or earlier is installed on your system you must delete your existing VAX–11 RSX software and then perform a new installation of Version 2.5. Section 1.7.1 describes a procedure to remove and delete your old VAX–11 RSX system.

```
Username:  SYSTEM
Password:  RET
   Welcome to VAX/VMS V5.4


   Last interactive login on Tuesday,  6-APR-1990 08:35
   Last non-interactive login on Tuesday,  6-APR-1990 09:19

$  @SYS$UPDATE:VMSINSTAL RSX025 MUA0:

        VAX/VMS Software Product Installation Procedure V5.4

It is 6-APR-1990 at 09:28.
Enter a question mark (?) at any time for help.

* Are you satisfied with the backup of your system disk [YES]?  RET

Please mount the first volume of the set on  MUA0:
* Are you ready?  Y RET
%MOUNT-I-MOUNTED, mounted on MUA0:

The following products will be processed:

  RSX V2.5

        Beginning installation of RSX V2.5 at 09:29

%VMSINSTAL-I-RESTORE, Restoring product saveset  A...
%VMSINSTAL-I-RELMOVED , The product's release notes have been successfully
  moved to SYS$HELP.

Please select one of the following actions:

     1)  Perform the installation
     2)  Create a file with a description of the installation
     3)  Both of the above

* What would you like to do [3]:  RET
```

1.9 VAX–11 RSX Installation Example

```
An optional help library is provided for use with the MCR  CLI.   This
library  supplies  help  for commands supported by MCR and for the RSX
utilities  supplied  with  VAX--11  RSX.   This  library  will  require
approximately  2000  additional  disk  blocks.  If you wish to install
this library on your system, answer yes to the following question.

* Do you want to install the MCR help library [YES]? RET
* Do you want to purge files replaced by this installation [YES]?  RET

        Product:      RSX
        Producer:     DEC
        Version:      2.5
        Release Date: 1-JUN-1990

* Does this product have an authorization key registered and loaded?  Y RET
* Do you want to run the IVP after the installation [YES]?  RET

RSX$STARTUP.COM, the startup command procedure, is used to set up  the
environment  for  VAX--11  RSX.   During  the  installation, it will be
placed in the SYS$STARTUP directory.  Your  system  startup  procedure
should  be  modified  to  invoke this procedure when the system boots.
However, it will not be necessary  to  reboot  the  system  after  the
installation,   since  this  procedure  is  invoked  as  part  of  the
installation.  See the VAX--11 RSX  Release  Notes,  or  the  procedure
itself for additional information.

The installation procedure has no further questions to ask.  The  rest
of  the installation will take approximately 20 minutes.  If you chose
to run the IVP it will take an additional 5 minutes.

%VMSINSTAL-I-SYSDIR, This product creates system disk directory
 VMI$ROOT:[SYSTEST.RSX].
%VMSINSTAL-I-SYSDIR, This product creates system disk directory
 VMI$ROOT:[001001].
%VMSINSTAL-I-SYSDIR, This product creates system disk directory
 VMI$ROOT:[001002].
%VMSINSTAL-I-SYSDIR, This product creates system disk directory
 VMI$ROOT:[001054].
%VMSINSTAL-I-SYSDIR, This product creates system disk directory
 VMI$SPECIFIC:[001001].
%VMSINSTAL-I-SYSDIR, This product creates system disk directory
 VMI$SPECIFIC:[001002].
%VMSINSTAL-I-SYSDIR, This product creates system disk directory
 VMI$SPECIFIC:[001054].
%VMSINSTAL-I-RESTORE, Restoring product saveset  B...

              ** VAXcluster common system disk setup **

If you intend to execute VAX--11 RSX on other nodes in your  VAXcluster
that share  this  common  system  root, and you have the appropriate
software license, you must ensure that SYS$STARTUP:RSX$STARTUP.COM  is
executed  on  each node to define the appropriate logical names and to
create the system directories required by RSX--11 software.

                    ** System $$n logical names **

RSX--11 tasks running  under  VAX--11  RSX  can  access  VAX/VMS  rooted
directories  only  through  the  use  of logical names having a format
that resembles that of an RSX--11 physical  device  name.   Similarly,
many  VAX/VMS  physical devices (including any device on a VAXcluster)
can be accessed by RSX--11 tasks only through the use of these  logical
names.
```

VAX--11 RSX uses several standard system-wide logical names, as well as internal "logical names" (such as SY and TI) for this purpose. See the VAX--11 RSX Compatibility Mode Reference Manual (Section 2.7) for more detailed information on VAX--11 RSX device name mapping.

The system directory root (SYS$SYSROOT) is always mapped by the VAX--11 RSX logical name LB. However, other devices and root directories should be mapped by "$$n" logical names, with the string "$$" followed by an octal unit number from 0 to 377. Note that even though $$n logical names are not strictly necessary (except for rooted directories) on non-VAXcluster systems, RSX--11 task performance (particularly the ALUN$ and ELP$ directives) will be improved through their use. This installation will create $$n names for all the disk and tape devices currently present on your system.

%RSX-I-$$N, the following $$n logical names will be defined:

$$n	VAX/VMS device
$$0	1DLA0:
$$1	1DLA1:
$$2	BLKHOL$DJA3:
$$3	BLKHOL$DUA0:
$$4	BLKHOL$DUA1:
$$5	BLKHOL$DUA4:
$$6	BLKHOL$DUA5:
$$7	BLKHOL$DUA6:
$$10	BLKHOL$DUA7:
$$11	BLKHOL$MUA0:
$$12	BLKHOL$MUA1:

%RSX-I-CONSOLE, console device CSA1: not configured; defining $$n anyway

The file SYS$STARTUP:RSX$DEVICE_NAMES.COM contains DEFINE commands to create each of the $$n logical names listed above. The RSX$STARTUP.COM startup procedure will execute RSX$DEVICE_NAMES.COM automatically when it runs following this installation, and during future system startups.

If there are devices that were not displayed above, that will be commonly accessed by compatibility mode tasks on your system, you may add $$n names to RSX$DEVICE_NAMES.COM. You may also delete $$n names that you do not want to be defined system-wide from this file. Additionally, you should define $$n names for any rooted directories that will be commonly accessed (look at the current definition of LB with the command "SHOW LOGICAL/FULL LB", for an example of this).

Note: The logical name RSX$DEVICE_CACHE is defined in RSX$DEVICE_NAMES.COM so that $$n logical names between

$$16 and $$347 inclusive

cannot be used. This is done to improve performance. You can alter this behavior by changing the definition of RSX$DEVICE_CACHE. See the VAX--11 RSX Installation Guide and Release Notes, and the VAX--11 RSX Compatibility Mode Reference Manual for more information.

Additionally, to improve performance, you may wish to reorganize the $$n names that have been defined, so that those that are most commonly used have the lowest unit numbers. Currently, they have been defined in the order in which SHOW DEVICE displays them.

RSX$DEVICE_NAMES.COM will not be modified on subsequent installations of VAX--11 RSX. If you wish to perform a future installation that requires changes to the RSX$DEVICE_NAMES.COM file, the old file must be deleted first. The installation will then create a new copy of RSX$DEVICE_NAMES.COM.

1.9 VAX–11 RSX Installation Example

```
%RSX-I-DESCRIBE, the installation is described in SYS$UPDATE:RSX025.TXT

 1) BACKTRANS (new image)
 2) BCK (new image)
 3) BRU (new image)
 4) CNV (new image)
 5) CRF (new image)
 6) DEF (new image)
 7) DES (new image)
 8) DMP (new image)
 9) DSP (new image)
10) DTE (new image)
11) EDI (new image)
12) FCSRES (new file)
13) FCSRES (new image)
14) FLX (new image)
15) ICM (new image)
16) IFL (new image)
17) INDSYS (new file)
18) LBR (new image)
19) MAC (new image)
20) MCR (new image)
21) MCRHELPLIB (new file)
22) MFT (new image)
23) ODT (new file)
24) PAT (new image)
25) PIP (new image)
26) QIOSYM (new file)
27) RMS11ODL (new file)
28) RMSDES (new file)
29) RMSLIB (new file)
30) RMSMAC (new file)
31) RMSRES (new file)
32) RMSRES (new image)
33) RST (new image)
34) RSX$DEVICE_NAMES (new file)
35) RSX$IVP (miscellaneous)
36) RSX$MCR_COMMANDS (new file)
37) RSX$REMOVE_AND_DELETE (new file)
38) RSX$STARTUP (new file)
39) RSX (new image)
40) RSXMAC (new file)
41) RSXSHR (new image)
42) RSXUSR (new image)
43) SLP (new image)
44) SYSLIB (new file)
45) TKB (new image)
46) TKB11MP43 (new image)
47) VMLIB (new file)
48) ZAP (new image)

%VMSINSTAL-I-MOVEFILES, Files will now be moved to their target directories...
%RSX-I-REDEFLOGICAL, system logical name SYS$SYSTEM will be redefined as
a search list to include SYS$SYSROOT:[001054]
%RSX-I-IVPSTART, VAX--11 RSX Version 2.5 IVP starting

                ** Installation Verification Procedure **

The installation verification procedure (IVP) tests  that  VAX--11  RSX
installed  correctly  by testing various compatibility mode utilities.
This serves as a test of the utilities themselves and of the RSX  AME.
Also,  when  DCL commands exist for compatibility mode utilities (i.e.
LIBRARY/RSX for LBR), these will be used to test  the  translation  of
DCL commands to MCR syntax for these utilities.
```

```
Several of the compatibility mode utilities will log their normal
actions to SYS$OUTPUT.  These informational messages as well as any
error messages that may be produced by these utilities, will not be in
the standard VAX/VMS message format.  If an error occurs, the IVP
will detect it and produce an error message in the standard VAX/VMS
format.  The IVP will also produce informational and success messages
in the standard format.

%RSX-I-TESTSTART, DMP testing started at 09:52:51
DMP -- DMP Version M08.01 (ANSI)

%RSX-S-TESTSUCCESS, DMP testing completed successfully at 09:53:21
%RSX-I-TESTSTART, LBR testing started at 09:53:21
Modules deleted:
OBJ3

Module "OBJ1  " replaced

Module "OBJ2  " replaced

Entry points deleted:
ENT$A1
ENT$B2
ENT$A3

Modules deleted:
MCLBR3

Module "MCLBR1" replaced

Module "MCLBR2" replaced

%RSX-S-TESTSUCCESS, LBR testing completed successfully at 09:54:55
%RSX-I-TESTSTART, PAT testing started at 09:54:55
%RSX-S-TESTSUCCESS, PAT testing completed successfully at 09:55:03
%RSX-I-TESTSTART, PIP testing started at 09:55:03
PIP -- PIP Version M1901 (ANSI)

%RSX-S-TESTSUCCESS, PIP testing completed successfully at 09:56:45
%RSX-I-TESTSTART, FLX testing started at 09:56:45
FLX -- FLX Version 17.01

%RSX-S-TESTSUCCESS, FLX testing completed successfully at 09:56:50
%RSX-I-TESTSTART, SLP testing started at 09:56:51
%RSX-S-TESTSUCCESS, SLP testing completed successfully at 09:57:01
%RSX-S-IVPSUCCESS, the installation verification of VAX--11 RSX
 Version 2.5 succeeded

        Installation of RSX V2.5 completed at 09:57

        VMSINSTAL procedure done at 09:58
```

1.10 VAX–11 RSX Messages

This section describes the messages that are displayed by VAX–11 RSX and suggests possible user action, if appropriate. Messages are displayed or printed in both uppercase and lowercase characters (when the output device is capable of displaying or printing lowercase characters).

In this section, the fixed and variable portions of the text of a message are distinguished by enclosing the variable portions in apostrophes (single quotation marks).

Message Format

Messages displayed by VAX–11 RSX have the following format:

```
%FACILITY-L-IDENT, text
[-FACILITY-L-IDENT, text]
```

FACILITY
A VAX–11 RSX facility or component name. A percent sign (%) prefixes the first message issued, and a hyphen (-) prefixes each subsequent message. For the VAX–11 RSX kit installation procedure, the facility name is RSX.

L
A severity level indicator with one of the following values:

Severity Level	Meaning
S	Success
I	Information
W	Warning
E	Error
F	Fatal or severe error

IDENT
An abbreviation of the message text; the message descriptions are alphabetized by this abbreviation.

TEXT
The explanation of the message.

1.10.1 Installation and Startup Messages

The following messages are displayed during the installation and startup procedures:

CONSOLE, console device CSA1: not configured; defining $$n anyway

Explanation: The installation or startup procedure has assigned the console mass-storage device a $$n name even though the console has not been connected to the system (using the system generation CONNECT CONSOLE command).

User Action: None.

DESCRIBE, the installation is described in SYS$UPDATE:RSX025.TXT

Explanation: Indicates where the installation description can be found after the installation is complete.

User Action: None.

DIRECTNF, directory 'RSX directory' not found when trying to redefine 'logical name'

Explanation: One of the RSX directories has been deleted.

User Action: Install VAX–11 RSX from the distribution kit again.

ERRORS, of the 'nn' actions listed above, the following 'nn' should be reviewed:

Explanation: If errors occur during the preceding nn number of actions in the software installation, this message indicates how many errors occurred and should be reviewed. A list follows that summarizes the points in the installation at which the errors occurred.

User Action: Review the specified actions.

GBLPAGES, not enough global pages available

Explanation: There are not enough unused global pages available to install the VAX–11 RSX images.

User Action: Increase the number of global pages by following the instructions in Section 1.4, step 3. Then perform the following steps:

- If you received this message during the VAX–11 RSX installation procedure, restart the installation.
- If you received this message during system startup, log in to the system manager's account and run the VAX–11 RSX startup procedure as follows:

```
$ @SYS$STARTUP:RSX$STARTUP
```

GBLSECTS, not enough global sections available

Explanation: There are not enough unused global sections available to install the VAX–11 RSX images.

User Action: Increase the number of global sections by following the instructions in Section 1.4, step 3. Then perform the following steps:

- If you received this message during the VAX–11 RSX installation procedure, restart the installation.
- If you received this message during system startup, log in to the system manager's account and run the VAX–11 RSX startup procedure as follows:

```
$ @SYS$STARTUP:RSX$STARTUP
```

KEEPOLD, retaining previous SYS$STARTUP:RSX$DEVICE_NAMES.COM

Explanation: A $$n name definition file already exists for your system. It will not be superseded.

User Action: None.

NEWDEVNAM, SYS$STARTUP:RSX$DEVICE_NAMES.COM file not found. New file being created.

Explanation: The startup procedure did not find a $$n name definition file., so it will create one.

User Action: See the description of the RSX$DEVICE_NAMES.COM file in Section 1.6 for information on tailoring this file for your system.

NOBLOCKS, RSX Version 2.5 requires 'n' additional blocks after installation

Explanation: After installation, VAX–11 RSX will use the specified number of blocks on the system disk. If the specified number of blocks is not available, the installation will fail. The number n in the message represents the difference between the required number of blocks and the number of free blocks on the system disk.

User Action: Purge or delete unnecessary files from the system disk.

NOIVP, the IVP will not be run because the license is not installed

Explanation: The license for VAX–11 RSX is not installed. Therefore, the IVP cannot be run. However, the installation will be completed. Although the IVP cannot be run automatically as part of the installation, you should run it after the license is registered to verify that the installation was successful.

User Action: Obtain and register a valid license for VAX–11 RSX. Log into the system manager account and run the IVP after the license has been registered, by entering the following command line:

```
$ @SYS$TEST:RSX$IVP
```

NO$$N, unable to read device information to create $$n names

Explanation: An error occurred while attempting to access the SHOW DEVICE command output.

User Action: You must manually add $$n name definitions to the RSX$DEVICE_NAMES.COM command file.

PREV24, old version of VAX–11 RSX found. Run RSX$REMOVE_AND_DELETE before installing this version.

Explanation: The installation or startup procedure found files or directories from VAX–11 RSX Version 2.3 or an earlier version.

User Action: Run the RSX$REMOVE_AND_DELETE.COM procedure to delete the files from a previous version of VAX–11 RSX before continuing with this installation. See Section 1.7 for information on running the RSX$REMOVE_AND_DELETE.COM procedure.

REDEFLOGICAL, system logical name 'logical name' will be redefined as a search list to include 'RSX directory'

Explanation: The startup procedure will redefine the system logical name to include the specified VAX–11 RSX directory. With the elimination of synonym directories, it becomes necessary to include the VAX–11 RSX directory [1,54] in the SYS$SYSTEM logical name. If necessary, you can also include VAX–11 RSX directories in SYS$LIBRARY and SYS$MESSAGE by modifying the VAX–11 RSX startup file.

User Action: None.

SOME, unable to collect all device information

Explanation: An error occurred during the processing of device information while creating $$n names.

User Action: You must manually complete the list of $$n definitions in the RSX$DEVICE_NAMES.COM command file.

SYSDIR, creating system disk directory 'RSX directory'

Explanation: The startup procedure will create the VAX–11 RSX directories [1,1], [1,2], and [1,54] if they do not already exist on your system disk.

User Action: None.

VERSION, RSX Version 2.5 must be installed on an existing VMS Version 5.0 system

Explanation: This version of VAX–11 RSX can be installed only on the specified version of VMS.

User Action: You have the wrong version of either the VAX–11 RSX or VMS operating system. You must obtain the correct version.

$$N, the following $$n logical names will be defined:

Explanation: The installation or startup procedure will list the $$n names created for disks and magnetic tapes on your system.

User Action: None.

1.10.2 IVP Messages

The following messages are displayed by the Installation Verification Procedure (IVP):

IVPFAILURE, the installation verification of VAX–11 RSX Version 2.5 failed

Explanation: The installation did not pass the IVP. Previous error messages should have indicated the specific reason or reasons for the failure.

User Action: Submit a Software Performance Report (SPR) describing the problem and include the console listing of the software installation.

IVPSTART, VAX–11 RSX Version 2.5 IVP starting

Explanation: The IVP is beginning.

User Action: None.

IVPSUCCESS, the installation verification of VAX–11 RSX Version 2.5 succeeded

Explanation: The IVP did not detect any errors in the software installation.

User Action: None.

TESTERROR, 'utility' failed test of 'test description'

Explanation: An error occurred while performing the specified utility test during the IVP.

User Action: Submit an SPR describing the problem and include the console listing of the software installation.

TESTSTART, 'utility' testing started

Explanation: The IVP has begun testing the specified utility.

User Action: None.

TESTSUCCESS, 'utility' testing completed successfully

Explanation: The IVP detected no errors.

User Action: None.

1.10.3 RSX$REMOVE_AND_DELETE Messages

The following messages are displayed by the new procedure to remove and delete the VAX–11 RSX files from your system:

DEASSIGNLOGS, the VAX–11 RSX logical names will be deassigned

Explanation: The VAX–11 RSX defined logical names will be deassigned. This includes logical names such as SY, LB, WK, and the $$n logical names.

User Action: None.

DELAFTEREXIT, delete this command procedure after it exits

Explanation: After you have run the remove and delete procedure on each node of your cluster, it is no longer needed and can be deleted.

User Action: After you have run it on all nodes in your cluster, delete the RSX$REMOVE_AND_DELETE command file. For example:

```
$ DELETE SYS$MANAGER:RSX$REMOVE_AND_DELETE.COM;*
```

DELETEDIR, directory 'RSX directory' will be deleted

Explanation: The VAX–11 RSX directory will be deleted from your system disk.

User Action: None.

DELETEFILES, the VAX–11 RSX files will be deleted

Explanation: All the files provided with VAX–11 RSX will be deleted from your system disk, except for the RSX$DEVICE_NAMES.COM file and the RSX$REMOVE_AND_DELETE.COM file. You will be given the option of keeping the RSX$DEVICE_NAMES.COM file or deleting it.

User Action: None.

EACHNODE, run this procedure on each node of the cluster

Explanation: Some of the actions of the remove and delete procedure, such as deleting the VAX–11 RSX directories from SYS$SPECIFIC and deassigning the VAX–11 RSX logical names, must be done on each node of a cluster.

User Action: Log in to each node of your cluster and run the remove and delete procedure on each node.

NOPRIV, you must have <CMKRNL,SYSNAM,SYSPRV> to run this procedure

Explanation: CMKRNL, SYSNAM, and SYSPRV privileges are needed to run the remove and delete procedure.

User Action: Log in to the system manager's account or another account that has the CMKRNL, SYSNAM, and SYSPRV privileges in order to run the remove and delete procedure.

REDEFLOGICAL, system logical name 'logical name' will be redefined to remove the reference to 'RSX directory'

Explanation: The system logical name will be redefined to remove the specified VAX–11 RSX directory. The directory was added to the system logical name by the VAX–11 RSX startup procedure and is not needed after VAX–11 RSX has been removed.

User Action: None.

REMDEL, WARNING: This procedure will remove VAX–11 RSX and delete all of its files. In order to use VAX–11 RSX, you will have to reinstall it.

Explanation: Before deleting the VAX–11 RSX files, the remove and delete procedure issues this warning and asks if you want to continue. If you continue, you will not be able to use VAX–11 RSX unless you install it again from its distribution kit.

User Action: Consider carefully before continuing and deleting all of the VAX–11 RSX files. Answer "No" if you decide that you want VAX–11 RSX to remain on your system, and to prevent the remove and delete procedure from deleting the VAX–11 RSX files. Answer "Yes" if you want to remove VAX–11 RSX from your system.

REMOVEIMAGES, the VAX–11 RSX installed images will be removed

Explanation: The VAX–11 RSX images that have been installed with the VMS Install Utility (INSTALL) will be removed.

User Action: None.

SYNREMOVED, old synonym directory 'RSX directory' will be removed

Explanation: The remove and delete procedure has found an RSX synonym directory entry from a previous version of VAX–11 RSX. The synonym directory will be removed.

User Action: None.

2 New Features for VAX–11 RSX

This chapter describes the major new software features introduced to VAX–11 RSX from Version 2.0 to Version 2.5.

- New system directives
- Common File Control Services (FCS)
- MCR support for most DCL commands
- R.MLB macro library
- Data Terminal Emulator (DTE) support
- Professional host communications functionality
- Help on MCR
- Changes to RMS–11 on VAX–11 RSX

See the Preface for a list of the new features introduced with VAX–11 RSX Version 2.5.

2.1 New System Directives—FSS$, PFCS$, and PRMS$

The Applications Migration Executive (AME) in the VAX–11 RSX product now supports the following RSX–11M–PLUS Executive directives: File Specification Scanner (FSS$), Parse FCS (PFCS$), and Parse RMS (PRMS$).

These directives are documented in the *RSX–11M–PLUS and Micro/RSX Executive Reference Manual*.

2.2 Common File Control Services

VAX–11 RSX Version 2.5 includes a common File Control Services (FCS) with RSX–11M–PLUS and VAX CoProcessor/RSX.

This means that a task built on an RSX–11M–PLUS, VAX CoProcessor/RSX or VAX–11 RSX system will run under any of the other systems. Thus, for example, a task originally built on an RSX–11M–PLUS system does not have to be rebuilt in order to run on a VAX–11 RSX system.

2.3 MCR Support for Most DCL Commands

The MCR CLI now uses the same command table used by DCL (DCLTABLES). Previously, MCR used a separate command table (MCRTABLES) and supported a small subset of DCL commands that were updated only with new releases of VAX–11 RSX. Because MCR now shares DCLTABLES with DCL, all DCL external commands are available

in MCR. Additionally, changes to DCLTABLES by users, layered product installations, and VMS upgrades are available immediately in MCR.

2.4 R.MLB Macro Library

Beginning with Version 2.3, the R.MLB macro library has been included in the VAX–11 RSX distribution kit. This library is useful if you are adding custom directives to VAX–11 RSX. Note, however, that the installation process does not copy it to your system disk. To copy the R.MLB library to your default directory, use the following VMS command:

```
$ BACKUP ddcu:RSX025.B/SAVE_SET/SELECT=R.MLB *
```

In the preceding command line, ddcu is the drive in which you have placed the VAX–11 RSX distribution kit.

2.5 Data Terminal Emulator

Beginning withe Version 2.1, VAX–11 RSX supports RSX–11 data terminal emulation (DTE).

VAX–11 RSX DTE allows communication from your VMS system to a RSX–11M–PLUS or Micro/RSX system. You can log in to the host system by using a terminal line connected from a system that runs DTE (the server system) to another system (the host system). If you have the SYSPRV privilege, you can invoke DTE as follows:

```
> DTE device[/DIAL[="number"]]
```

In the preceding command line, the device parameter specifies the terminal line on the server system that is connected to a host system or to a modem. If you need to define a $$n logical name for the device, see the *VAX–11 RSX Compatibility Mode Reference Manual* for more information on $$n logical names.

The optional /DIAL qualifier allows the specification of a dial command string when the line is connected to a DF03, DF112, or DF224 modem. If you specify /DIAL without a number, DTE prompts for the number to be dialed. The string may consist of any of the digits 0 through 9 and the characters P, T, *, #, –, and a space. (Refer to the user guide for your modem for an explanation of these characters.) DTE specifies the terminating character.

DTE sets most of the necessary terminal attributes. However, before running DTE you can set attributes by entering the following command:

```
> SET TERMINAL/PERMANENT
```

The LOG_IO privilege is required to change attributes.

To terminate emulation, type the LOGOUT command from the host system and then press CTRL/P. The following message is displayed:

```
%DTE-S-EMUEXIT, Emulation exiting... Please wait
```

You are then returned to command level execution on the server system. See the *RSX–11M–PLUS and Micro/RSX System Management Guide* for more information on using DTE and MFT.

Note: When you use DTE on VAX–11 RSX, there are several differences not noted in the *RSX–11M–PLUS and Micro/RSX System Management Guide*. These include the following:

- **DCL command syntax (as described in the *RSX–11M–PLUS and Micro/RSX System Management Guide*) is not applicable when using DTE on VAX–11 RSX.**
- **There is no recommended baud rate for running DTE on VAX–11 RSX.**
- **The DTE qualifier /SLAVE is not supported.**

2.6 Professional Host Communications

With VAX–11 RSX Version 2.4, the Micro/RSX File Transfer (MFT) Utility superseded the Professional File Transfer (PFT) Program. MFT is available on VAX–11 RSX with both the forms interface and the PFT Command Line Format (CFT).

To use MFT in place of the PFT Command Line Format, substitute the command MFT for the command PFT on all command lines shown in the *PRO/Communications User's Guide*.

Previously, for example, when you transferred a file from a Professional 300 series system to a host, the CFT command line was as follows:

```
> CFT output_filespec/PRO=input_filespec
```

Now, to transfer a file from a Professional system to a host, using MFT, substitute MFT in the command line, as follows:

```
> MFT output_filespec/PRO=input_filespec
```

In the preceding command lines, the /PRO switch indicates that the Professional system is sending the file.

2.7 Help on MCR

Beginning with Version 2.3, VAX–11 RSX provides a brief help file for the Monitor Console Routine (MCR) to the VMS help files. This allows you to get help on the MCR command.

2.8 RMS–11 on VAX–11 RSX

This section describes new features and supplementary information for Record Management Services (RMS–11) on VAX–11 RSX.

Because RMS–11 Version 2.0 has not changed version numbers since the last release of VAX–11 RSX, new information in this chapter has not yet been incorporated into the RMS–11 manuals.

2.8.1 Files and Placement on the Distribution Kit

The following table describes the contents of the RMS–11 Version 2.0 distribution kit:

File Name	Destination	Comments
RMSMAC.MLB	LB:[1,1]	Can be deleted if you are not using MACRO–11 RMS–11 programs
RMSLIB.OLB	LB:[1,1]	Object library for RMS–11 local access
BCK.EXE	LB:[1,54]	RMSBCK utility; does not use RMSRES
RST.EXE	LB:[1,54]	RMSRST utility; does not use RMSRES
DEF.EXE	LB:[1,54]	RMSDEF utility; does not use RMSRES
DSP.EXE	LB:[1,54]	RMSDSP utility; does not use RMSRES
CNV.EXE	LB:[1,54]	RMSCNV utility; does not use RMSRES
DES.EXE	LB:[1,54]	RMSDES utility; does not use RMSRES
IFL.EXE	LB:[1,54]	RMSIFL utility; does not use RMSRES
RMSDES.IDX	LB:[1,2]	Indexed help file used by RMSDES
RMS11.ODL	LB:[1,1]	Prototype ODL file
RMS11S.ODL	LB:[1,1]	ODL file for sequential access
RMS12S.ODL	LB:[1,1]	ODL file for sequential access
RMS11X.ODL	LB:[1,1]	Standard indexed file ODL
RMS12X.ODL	LB:[1,1]	Indexed file ODL
DAP11X.ODL	LB:[1,1]	ODL for use with overlaid RMSDAP
RMSRES.EXE	LB:[1,1]	TSK image for linking against RMSRES
RMSRES.STB	LB:[1,1]	STB file for RMSRES

2.8.2 Building RMS–11 Applications

The RMSLIB object library provided on VAX–11 RSX contains four versions of the R0EXEC programming module. The default version (R0EXEC) of this module is sufficient for most situations; however, you may wish to use another version, depending on your requirements. A list of the modules and their functions follows.

Module	Function
R0EXEC	Asynchronous/synchronous Executive with RMS–11 software that allows execution of a task when the task is transported to an RSX–11 system.
R0EXSY	Synchronous Executive with RMS–11 software that allows execution of a task when the task is transported to an RSX–11 system.

Module	Function
R0EXEV	VAX–11 RSX version only; asynchronous/synchronous Executive optimized for size (no RMS–11 software is included). If the task is transported to an RSX–11 system, R0EXEV reports ER$ENV (feature not in selected RMS–11 environment) error messages on all RMS–11 operations except $WAIT.
R0EXSV	VAX–11 RSX version only; synchronous Executive optimized for size (no RMS–11 software included). If the task is transported to an RSX–11 system, R0EXSV reports ER$ENV (feature not in selected RMS–11 environment) error messages on all RMS–11 operations except $WAIT.

Building Applications on VAX–11 RSX Only

If you are building RMS–11 applications to run only on VAX–11 RSX, you can replace the RMS–11 overlay specification in the ODL files of your tasks with one of the following modules:

- R0EXEV—for asynchronous/synchronous RMS–11 operations
- R0EXSV—for synchronous RMS–11 operations

Usually, R0EXEV or R0EXSV can replace the entire RMS–11 overlay structure because most RMS–11 functions are performed in the VAX–11 RSX RMS–11 to RMS-32 translator. The following example illustrates how your RMS–11 overlay can be defined:

```
;
;       VAXRSXRMS.ODL VAX--11 RSX Version 2.4
;
;       ODL file to build an RMS--11 task to run only on VAX--11 RSX
;
        .NAME   RMS11
;
; For asynchronous/synchronous support
;
RMSROT: .FCTR  LB:[1,1]RMSLIB/LB:RMSSYM:R0EXEV
;
; For synchronous support
;
;RMSROT:.FCTR  LB:[1,1]RMSLIB/LB:RMSSYM:R0EXSV
;
RMSALL: .FCTR  RMS11
        .END
```

The following example shows an ODL file using an RMS–11 overlay defined by LB:[1,1]RMS12S.ODL:

```
;
; VAX--11 RSX Version 2.4
; ODL to build an RMS--11 task, using the RMS--11 overlay defined by
; LB:[1,1]RMS12S.ODL
;
        .NAME   NUME
        .ROOT   RAD

RAD:    .FCTR   NUME-RMSROT-F1-F2-F3,RMSALL
F1:     .FCTR   RMSQIOCL
F2:     .FCTR   GSA
F3:     .FCTR   LB:[1,1]VMLIB/LB:INIDM:EXTSK
;
;       Define the RMS--11 overlay
;
@LB:[1,1]RMS12S.ODL

        .END
```

The following example shows how to replace the RMS–11 overlay defined by LB:[1,1]RMS12S.ODL with the R0EXEV module:

```
;
; VAX--11 RSX Version 2.4
; ODL to build an RMS--11 task, that runs only on VAX--11 RSX.
; The RMS--11 overlay defined by LB:[1,1]RMS12S.ODL is replaced
; with the R0EXEV module.
;
        .NAME   NUME
        .ROOT   RAD

RAD:    .FCTR   NUME-RMSROT-F1-F2-F3,RMSALL
F1:     .FCTR   RMSQIOCL
F2:     .FCTR   GSA
F3:     .FCTR   LB:[1,1]VMLIB/LB:INIDM:EXTSK

;
;       Define the RMS--11 overlay
;
        .NAME   RMS11
RMSROT: .FCTR   LB:[1,1]RMSLIB/LB:RMSSYM:R0EXEV
RMSALL: .FCTR   RMS11

        .END
```

Note: If you use the R0EXEV or R0EXSV module to replace the RMS–11 overlay in your task, you will not be able to transport your task to an RSX–11 system. However, using these modules reduces the amount of memory the task requires by 6 to 12 Kb, depending on the size of the RMS–11 overlay.

2.8.3 RMS–11 Memory-Resident Libraries

VAX–11 RSX supports RMS–11 memory-resident libraries. The RMS–11 memory-resident library configuration on VAX–11 RSX consists of the following two files:

- LB:[1,1]RMSRES.EXE
- LB:[1,1]RMSRES.STB

These two files are distributed with VAX–11 RSX; they are the same files distributed with RSX–11M–PLUS.

On VAX–11 RSX, the RMS–11 memory-resident library files function only at task initialization time to attach the required regions. They have no further role because all the RMS–11 functions during the task's execution are performed by the VAX–11 RSX RMS–11 to RMS-32 translator.

Tasks built on RSX–11 do not correctly transport to VAX–11 RSX unless they are built with the R0EXSY or R0EXEC modules. VAX–11 RSX supplies these modules in RMSLIB.OLB. They check whether the task is executing in the VAX–11 RSX environment and can interface with the system.

Follow these steps to replace the R0EXSY or R0EXEC modules in your RSX–11 RMSLIB.OLB:

1 Extract the R0EXEC and R0EXSY modules on VAX–11 RSX as follows:

```
> LBR R0EXECAME.OBJ=LB:[1,1]RMSLIB/EX:R0EXEC
> LBR R0EXSYAME.OBJ=LB:[1,1]RMSLIB/EX:R0EXSY
```

2 Replace the R0EXEC and R0EXSY modules on RSX–11 as follows:

```
> LBR LB:[1,1]RMSLIB/RP=[UIC]R0EXECAME
> LBR LB:[1,1]RMSLIB/RP/-EP=[UIC]R0EXSYAME
```

To ensure that the RMS–11 memory-resident library files on VAX–11 RSX and RSX–11M–PLUS are identical, RMSRES is not built differently for compatibility mode. Therefore, to allow VAX–11 RSX to map the library, the RMSRES.EXE image must be installed as a shared, read-only image. The VAX–11 RSX startup command file, SYS$STARTUP:RSX$STARTUP.COM, automatically installs the RMSRES.EXE image.

3 Modifications to Existing VAX–11 Features

This chapter describes enhancements and corrections to the existing features of VAX–11 RSX and to the utilities that VAX–11 RSX uses. It includes cumulative changes from VAX–11 RSX Version 2.0 to Version 2.5.

The modifications to VAX–11 RSX include changes to the following features and utilities:

- Installation and startup
- Synonym directories
- LMF support
- Applications Migration Executive (AME)
- Backup and Restore Utility (BRU)
- Multivolume tapes
- Device and file handling
- Peripheral Interchange Utility (PIP)
- Executive directives
- Indirect
- Login procedure
- MACRO–11
- Monitor Console Routine (MCR)
- PDP–11 Emulator
- RMS–11
- VAX–11 RSX System Library (SYSLIB)
- Task Builder (TKB)
- Terminal Driver
- VAX–11 RSX Error Messages

See the preface of this manual for a list of the corrections and modifications specific to VAX–11 RSX Version 2.5.

3.1 Installation and Startup

This section describes changes to the installation and startup procedures since VAX–11 RSX Version 2.0, and is of particular interest to system managers.

Modifications to Existing VAX–11 Features

3.1 Installation and Startup

3.1.1 New and Renamed Files

The following table summarizes the new and renamed installation and startup files in VAX–11 RSX:

New File	Renamed File
SYS$MANAGER:RSX$MCR_COMMANDS.CLD	(No file replaced)
SYS$MANAGER:RSX$REMOVE_AND_DELETE.COM	(No file replaced)
SYS$STARTUP:RSX$STARTUP.COM	SYS$MANAGER:VAX11RSX.COM
SYS$STARTUP:RSX$DEVICE_NAMES.COM	SYS$MANAGER:VAX11RSX$DEVICE_NAMES.COM
SYS$SYSROOT:[SYSTEST.RSX]*.* (all files in this subdirectory)	(No file replaced)
SYS$TEST:RSX$IVP.COM	SYS$SYSROOT:[SYSTEST.RSX]RSX$IVP.COM

3.1.1.1 RSX$DEVICE_NAMES.COM File

The file SYS$STARTUP:RSX$DEVICE_NAMES.COM (previously named SYS$MANAGER:VAX11RSX$DEVICE_NAMES.COM) defines the $$n logical names used by VAX–11 RSX to refer to VMS devices.

Prior to Version 2.5, the RSX$DEVICE_NAMES.COM file was created during the installation of VAX–11 RSX if it did not already exist. With Version 2.5, the RSX$DEVICE_NAMES.COM file is also created during startup if it does not exist.

Thus, if your device configuration changes, you can have the VAX–11 RSX startup procedure create a new device names file for you.

For example, if you delete the existing RSX$DEVICE_NAMES.COM file and then invoke the SYS$STARTUP:RSX$STARTUP.COM procedure, a new RSX$DEVICE_NAMES.COM file will be created that includes the devices currently configured in your system.

3.1.1.2 Renamed Startup Files

To maintain consistency among all VMS Version 5.0 layered products, a new convention was established with VAX–11 RSX Version 2.4 for layered product startup files. The names and locations of VAX–11 RSX startup files were changed as follows:

Old File	New File
SYS$MANAGER:VAX11RSX.COM	SYS$STARTUP:RSX$STARTUP.COM
SYS$MANAGER:VAX11RSX$DEVICE_NAMES.COM	SYS$STARTUP:RSX$DEVICE_NAMES.COM

3.1.2 Defining Device Logical Names During Installation

Prior to Version 2.4, the following message was displayed if the $$n logical names created during your VAX–11 RSX installation had been previously defined:

```
%DCL-I-SUPERSEDE, previous value of logical_name has been superseded
```

This message is no longer displayed.

3.1.3 RSX$DEVICE_CACHE Logical Name

The logical name RSX$DEVICE_CACHE is defined in RSX$DEVICE_NAMES.COM. This logical name controls whether the $$n logical name translations are cached and restricts the range of $$n names you can use on your system. It can greatly improve performance because it reduces the number of $$n logical name translations that VAX–11 RSX must perform to map physical device names to $$n names.

For example, the following definition enables caching of $$n logical name translations and also causes VAX–11 RSX to ignore all $$n logical names from $$31 to $$347 (inclusive):

```
$ DEFINE RSX$DEVICE_CACHE CACHE,30,350 /SYSTEM/EXECUTIVE_MODE)
```

You can modify the definition of RSX$DEVICE_CACHE to disable caching and to change the range of valid $$n logical names. See the *VAX–11 RSX Compatibility Mode Reference Manual* for more information.

Previously, the $$347 logical name could always be used even when the upper limit of the RSX$DEVICE_CACHE logical was set to 350. Beginning with VAX–11 RSX Version 2.5, setting the upper limit of the RSX$DEVICE_CACHE logical to 350 works as expected and prevents the use of $$347.

3.1.4 Contiguous Space Requirements

Since Version 2.3, VAX–11 RSX installations no longer require contiguous space on system disks. If contiguous space is available, some of the VAX–11 RSX files will be contiguous; otherwise, the installation procedure proceeds as normal and files are not contiguous.

3.1.5 Running the IVP After Installation

You can run the Installation Verification Program (IVP) for VAX–11 RSX during the installation or at any time after the installation.

Prior to Version 2.4, the IVP could only be run during the installation and was deleted before the installation completed. Now the subdirectory RSX is created in directory SYS$TEST and the files that comprise the IVP are copied there. You can run the IVP by logging into the system manager account and entering the following command:

```
$  @SYS$TEST:RSX$IVP RET
```

Modifications to Existing VAX–11 Features

3.1 Installation and Startup

3.1.6 Remove and Delete Procedure

A cleanup procedure, SYS$MANAGER:RSX$REMOVE_AND_ DELETE.COM, was supplied with VAX–11 RSX Version 2.4. This procedure removes any installed images and logical names assigned by the VAX–11 RSX startup procedure and deletes all files created by the installation of VAX–11 RSX. You can use the remove and delete procedure if you no longer need VAX–11 RSX on your system. In order to user VAX–11 RSX at a later date, you have to install it again from its distribution kit.

Run the remove and delete procedure with the following command line:

```
$  @SYS$MANAGER:RSX$REMOVE_AND_DELETE
```

3.2 Synonym Directories

Prior to VAX–11 RSX Version 2.4, the RSX directories [1,1], [1,2], and [1,54] were created as synonym directories for the VMS directories [SYSLIB], [SYSMSG], and [SYSEXE]. This made file management on a VMS system disk difficult.

After Version 2.4, VAX–11 RSX no longer created the RSX directories as synonyms. Now they are created as actual directories on the system disk. They remain distinct from the [SYSLIB], [SYSMSG], and [SYSEXE] directories and the SYS$LIBRARY, SYS$MESSAGE, and SYS$SYSTEM logical names that refer to them.

The VMS MCR command expects to find VAX–11 RSX images in SYS$SYSTEM, so the VAX–11 RSX startup procedure defines SYS$SYSTEM to be a search list that includes the new [1,54] directory. Since neither the VMS nor VAX–11 RSX operating system looks for any VAX–11 RSX files in SYS$LIBRARY or SYS$MESSAGE, you do not need to redefine those logicals as search lists.

If you have code that refer to VAX–11 RSX files by using the VMS logicals SYS$LIBRARY and SYS$MESSAGE (instead of the corresponding VAX–11 RSX device and directories LB:[1,1] and LB:[1,2]), you can define SYS$LIBRARY and SYS$MESSAGE as search lists. You can do this by removing the comment character from the appropriate lines in the VAX–11 RSX startup file SYS$STARTUP:RSX$STARTUP.COM.

However, DIGITAL strongly recommends that your code and command procedures use LB:[1,1] and LB:[1,2] to reference these files.

If you are upgrading from a version of VAX–11 RSX prior to Version 2.4, you should make sure that your compatibility mode libraries, files, and tasks are in the correct directories, as shown in the following table:

From	To
SYS$LIBRARY	LB:[1,1]
SYS$MESSAGE	LB:[1,2]

From	To
SYS$SYSTEM	LB:[1,54]

Because VAX–11 RSX redefines SYS$SYSTEM to be a search list that includes the VAX–11 RSX directory [1,54], you may notice the presence of the [1,54] directory when you use the SYS$SYSTEM logical.

For example, if you issue a directory command on SYS$SYSTEM, you also get a directory of the files in [1,54]. If you get a "file not found" error when trying to access a file in SYS$SYSTEM, the [1,54] directory is given in the error message text since it is the last leaf of the SYS$SYSTEM search list.

The following interactive example illustrates a command that generates such an error message:

```
$  RUN SYS$SYSTEM:XXX.EXE
%DCL-W-ACTIMAGE, error activating image SYS$SYSTEM:XXX.EXE
-CLI-E-IMAGEFNF, image file not found NOD$DJA3:[SYS0.SYSCOMMON.]
[001054]XXX.EXE;
```

3.3 LMF Support

Beginning with Version 2.4, VAX–11 RSX supports the License Management Facility (LMF) features provided by the VMS operating system. To manage the advanced features of this facility, please refer to the *VMS License Management Utility Manual*.

3.4 Accessing Multivolume Tapes

When you access a multivolume tape under VAX–11 RSX (for instance, to save and restore files using BRU), you must mount the tape with the /MULTI_VOLUME qualifier. For example:

```
$  MOUNT /FOREIGN /MULTI_VOLUME ddcu:
```

The use of the preceding command requires VOLPRO privilege.

3.5 Applications Migration Executive

This section describes changes to the Applications Migration Executive (AME) of VAX–11 RSX.

3.5.1 Task Overlay Loading Correction

Prior to VAX–11 RSX Version 2.5, tasks which contained both memory-resident and disk-resident overlays did not load correctly at runtime.

This problem was caused by an error in the RSX AME's emulation of memory-resident overlays when a task also contained disk-resident overlays. It did not occur if the task was built with all memory-resident or all disk-resident overlays.

The problem was corrected for Version 2.5

3.5.2 Asynchronous System Traps

Execution Failures

Prior to VAX–11 RSX Version 2.2, an asynchronous system trap (AST) might not have been executed when all the following conditions were present:

- The Suspend (SPND$S) directive, Stop (STOP$S) directive, or a similar directive was used to wait for the AST to occur.
- The program handled only one type of AST (for example, only timer ASTs).
- The AST occurred during the execution of an AST Service Exit (ASTX$S) directive.

  The following example shows how the last condition may occur:

  1 A task issues a Mark Time (MRKT$) directive that specifies an AST routine.

  2 The time interval unit (tnt) and the time interval magnitude (tmg) specified with the MRKT$ directive cause the ASTs to occur at a time interval equal to the amount of time it takes to execute the AST routine.

  Under these circumstances, the AST routine may not have been executed.

This problem has been corrected.

AST Problem After Executing RSX Directive

Prior to VAX–11 RSX Version 2.5, there was a small time interval at the end of processing an RSX directive during which ASTs may not have been handled correctly. If an AST occurred during this window, VAX–11 RSX queued the AST for the compatilbility-mode task, but did not deliver it. The AST would not be delivered until the task issued another RSX directive. Normally, this delay in delivering the AST would not be noticed, however if the task never issued another RSX directive, the AST would never be delivered.

This problem has been corrected.

3.5.3 Task Names and Process Names

If a task name is specified using the Task Builder TASK option when an RSX–11 task is built, the system uses that name as the task name when the task is run under VAX–11 RSX.

Under RSX–11, the task name specified at the time of task building can be overridden when the task is installed. VAX–11 RSX does not incorporate the concept of an installed task. Thus, if a task name is specified at the time of task building, it cannot be overridden.

If you do not specify a task name at the time of task building, or if the task name is of the form ...xxx, VAX–11 RSX uses the process name, modified to make it a unique, valid task name as described in Section 2.3.2, of the VAX–11 RSX Compatiblity Mode Reference Manual, as the task name.

When a task starts running under VAX–11 RSX, the system changes the process name to match the task name. The system restores the original process name when the task completes processing.

Spawning a task using the Spawn directive (SPWN$) creates a subprocess to execute a task or a command line. The subprocess will have an initial process name that is the same as the tname parameter of the SPWN$ directive. 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 by 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.

Once the subprocess starts running an RSX–11 task, the system changes the subprocess name to the RSX–11 task name as described above.

The preceeding is not a change in VAX–11 RSX behavior, but merely a clarification.

3.5.4 Batch Process Names

Beginning with Version 2.3, VAX–11 RSX now scans a process name created by VAX–11 RSX for a batch job to determine whether a character in the batch process name is not part of the Radix-50 character set. If the character is not a Radix-50 character, VAX–11 RSX first replaces that character with an “X” in the batch process name and then continues processing the batch job.

3.5.5 Compatibility Mode

Previously, a compatibility mode task that exceeded 64K bytes and included memory-resident overlays aborted, returning the following error message:

```
%RSX-E-TOOBIG, task exceeds compatibility mode address limits
```

This problem has been corrected.

3.5.6 GRPNAM Privilege Required for Group-Wide Logical Names

The discussion of logical names in Section 2.6.3 of the *VAX-11 RSX Compatibility Mode Reference Manual* states that the GRPNAM privilege is required to create or delete a logical name from the group logical name table.

Previously, this requirement was not enforced because the VAX-11 RSX AME was installed with sufficient privileges to allow creating and deleting group logical names.

Beginning with VAX-11 RSX Version 2.5, your process must have the GRPNAM privilege to create or delete logical names from the group logical name table.

3.5.7 SYS$ERROR Definition

The AME no longer aborts if SYS$ERROR contains a full directory specification. This modification was made because an error occurred when the AME translated VMS logical names that were used as RSX pseudodevices; the error subsequently caused the AME to abort.

3.5.8 Privileged Images

Previously, pressing CTRL/Y terminated the execution of the compatibility mode task. A privileged image (for example, the AME) running in a process that has MCR as the command interpreter now continues, if you press CTRL/Y and then issue the CONTINUE or DEBUG command.

3.6 Backup and Restore Utility

This section describes changes to the Backup and Restore Utility (BRU).

3.6.1 Multi-Volume Restore Correction

Prior to Version 2.5, BRU sometimes failed to set the volume valid bit on a volume that was replaced by the user. This failure could prevent BRU from restoring multi-volume disk savesets.

The problem was corrected for VAX–11 RSX Version 2.5.

3.6.2 BRU /APPEND Qualifier

To ensure the integrity of backup sets, BRU now restricts the use of the /APPEND qualifier to the first volume. Consequently, backup sets cannot be appended to continuation volumes. Prior to Version 2.4, if you used the /APPEND qualifier for a volume other than the first, the backup set could not be restored.

3.6.3 BRU /IDENTIFICATION Qualifier

For VAX–11 RSX Version 2.4, BRU included the following new qualifier:

/IDENTIFICATION
The /IDENTIFICATION qualifier directs BRU to identify itself by displaying its version number. This qualifier may be specified on a command line alone or in combination with other qualifiers.

3.6.4 BRU Messages

For VAX–11 RSX Version 2.4, BRU included the following new messages:

- BRU--*FATAL*-- Cannot append on continuation volume
- BRU-- BRU version xx.xx

3.6.5 BRU Multiple Disk Operations

Prior to Version 2.4, multiple disk backup and restore operations failed with a privilege violation error when performed by using BRU. This occurred because the program code that attaches the image disk did not function properly. Consequently, BRU failed whenever it tried to attach to image disks that were mounted foreign. This problem has been corrected.

3.6.6 BRU Performance Enhancement

The overlay description used to build BRU was modified with VAX–11 RSX Version 2.4. As a result, files are restored to mounted volumes faster. Overlay loads are no longer needed to create a file on a mounted output volume.

3.6.7 Creating User File Directories

Structure Level 2 Volumes

For VAX–11 RSX Version 2.5, BRU can now create User File Directories (UFDs) on Files–11 Structure Level 2 disk volumes when the /UFD qualifier is used.

Structure Level 1 Volumes

Beginning with VAX–11 Version 2.3, BRU can be used to create User File Directories (UFDs) on a mounted Files–11 Structure Level 1 output volume when the /UFD qualifier is specified. This feature eliminates the need to manually create the necessary UFDs when copying files to a mounted output volume. For more information on using the /UFD qualifier, refer to the section of the *RSX–11M/M–PLUS Utilities Manual* that describes BRU.

3.7 Disk Save and Compress Utility

The Disk Save and Compress Utility (DSC1) is no longer included with VAX–11 RSX. The Backup and Restore Utility (BRU), which is included in the VAX–11 RSX system, performs all the operations possible with DSC1, and it performs them with greater efficiency and accuracy.

3.8 Device and File Handling

This section describes changes in device name and file handling for VAX–11 RSX.

3.8.1 Device Mnemonics and Physical Devices

Table 3–1 lists the changes made to RSX–11 device mnemonics and physical device correspondence since VAX–11 RSX Version 2.1. These mnemonics are returned by the Get Device Information (GDVI$) directive for each device.

Table 3–1 RSX–11 Device Mnemonics

Media	RSX–11 Mnemonic	Physical Device
Tape	MU	TK50[1]
	MS	TS11, TU80[2]

[1]New device

[2]Changed device

3.8.2 Device Name Logicals

Previously under VAX-11 RSX, leading zeros in RSX logical device names were significant. Logical device names of the form dd, dd0, dd00, and dd000 could each refer to a different device. For example, previously, you could define the logical names AB0 and AB00 as follows:

```
$  DEFINE AB0 DUA3:
$  DEFINE AB00 DUA5:
```

With the preceding definitions, AB0: would refer to DUA3: and AB00: would refer to DUA5:. This did not match the behavior under RSX.

Beginning with VAX–11 RSX Version 2.5, references to RSX device names of the form dd0, dd00, and dd000 are converted to the form dd. For example, AB00 refers to the logical name AB.

Any logical names of these forms should be replaced with a logical name of the form dd. (For compatibility with previous releases of VAX–11 RSX, if a logical name of the form dd is not found, VAX–11 RSX also looks for a logical name of the form dd0.)

Similarly, logical device names of the form dd0n and dd00n are converted to the form ddn. For example, AB03 refers to the logical name AB3.

3.8.3 IE.NSF Error Code

Previously, VAX–11 RSX returned the error code IE.BDI when a directory did not exist. This differed from the error code, IE.NSF, that RSX–11 returns for a nonexistent directory.

This problem was corrected by changes to the AME FCS–11 interface. VAX–11 RSX now returns the error code IE.NSF for a nonexistent directory.

3.8.4 FCS Create and Open Operations

Previously, a problem occurred when tasks were built under VAX–11 RSX that specified file names for File Control Services (FCS) create and open file operations without one of the following specifications:

- A directory name string in the FCS data set descriptor block
- A valid directory identification in the FCS default file name block

When the task was copied to an RSX–11 system and then run, the files were created or opened in the directory [0,0] instead of in the user's default directory. This problem has been corrected.

3.8.5 FCS Created Files

Previously, a compatibility mode task that used FCS to create a new file received a version number of zero (0) in the file header. This problem has been corrected; files created by FCS now specify the correct version number in the file header.

3.8.6 FCS .PARSE Routine

Previously, file name parse operations performed by the File Control Services (FCS) .PARSE routine may have failed and returned the IE.BDI error code if both of the following conditions existed:

- The device was a branch of a search list.
- The directory string was not specified in the FCS data set description.

This error, which occurred even if a valid directory identification was specified in the FCS default file name block, has been corrected.

3.9 Peripheral Interchange Program (PIP)

This section describes modifications to the Peripheral Interchange Program (PIP) for VAX–11 RSX.

3.9.1 Used and Free File Header Count

Previously, the PIP /FR command incorrectly reported free and used disk file headers on Files–11 On-Disk Structure Level 2 disks that had been initialized with a cluster size greater than one. Although cluster size is used by the PIP /FR command when counting used data file blocks, there is no clustering in the index file bit map.

3.9.2 PIP /PU Command

Previously, the PIP /PU command failed to fully purge files that were positioned in certain sequences on a Files–11 On-Disk Structure Level 1 disk. For example, file BETA.LST would not have been purged in the following sequence:

```
ALPHA.LST;2
BETA.LST;3
BETA.LST;1
BETA.LST;2
ALPHA.LST;5
ALPHA.LST;4
```

In the preceding example, the purge failed for the following reasons:

- All versions of BETA.LST are listed after the first occurrence of ALPHA.LST.
- All versions of BETA.LST are listed before the highest version number of ALPHA.LST.

This problem has been corrected. The PIP /PU command no longer depends on file sequence to purge correctly.

3.10 Executive Directives

This section describes modifications that have been made to Executive directives.

3.10.1 ALUN$ Directive

The following problems with the Assign Logical Unit Number (ALUN$) directive were corrected in VAX–11 RSX Version 2.4:

Reassigning a LUN

Previously, changes made to an RSX–11 device name while a logical unit number (LUN) was assigned to that device may not have been recognized by the AME, even after the LUN was reassigned. In particular, when processing an indirect command file that modified the name of an RSX–11 device that had a LUN assigned, Indirect ignored the new name definitions and retained the oldest device name, despite the LUN's reassignment.

Reassigning a LUN from TI

Previously, if a task had a LUN assigned to the device TI and it tried to reassign the LUN to a different device, Tx0 (where the first character of the device name was T and the unit number was 0), VAX–11 RSX did not reassign the LUN. This problem has been corrected.

For example, assume that LUN 5 is assigned to TI by default. If a task reassigned LUN 5 to TT0, the reassign operation appeared to succeed, but in previous versions of VAX–11 RSX, LUN 5 was still assigned to TI. Now the task actually reassigns LUN 5 to TT0.

Targeted to a Template Device

Previously, if the target of an ALUN$ directive was a template device, the ALUN$ directive failed.

This problem has been corrected.

3.10.2 ALUN$ and ACHN$ Directive

The following is a list of ALUN$ or Assign Channel to Logical Name (ACHN$) directive problems in previous versions of VAX–11 RSX that have been corrected:

- An ALUN$ directive or ACHN$ directive failed and returned an error code (IE.LNL) when you assigned a device that had been used previously for Record Management Services (RMS–11) operations.
- In a compatibility mode task, if two or more LUNs were assigned to the same device (except SY and LB), reassigning the first LUN to another device caused the I/O operation directed to the other LUNs to fail.
- A LUN could not be reused for RMS–11 file operations after an RMS–11 $CLOSE command because VAX–11 RSX returned an error code, even though the file had been successfully closed.

3.10.3 ATRG$ Directive

Prior to VAX–11 RSX Version 2.5, the Attach Region directive (ATRG$) could prevent multiple tasks from accessing shared regions with memory-resident overlays.

The ATRG$ directive has an option that returns the region ID of a previously attached region without attaching the region a second time.

In returning the region ID, VAX–11 RSX left the region lock in protected write mode. The prevented a second process from accessing a shared region until the first process released it. This problem has been corrected—ATRG$ now lowers the region lock of the attached region to null mode.

The Attach Region (ATRG$) directive has an option that returns the region ID of a previously attached region without attaching the region a second time.

In Version 2.3, VAX–11 RSX reattached to the region and incorrectly returned a unique region identifier for each subsequent attachment.

While this problem only reduced efficiency, it was corrected in Version 2.4.

3.10.4 BOM$ Directive

For compatibility with RSX–11M–PLUS and Micro/RSX, VAX–11 RSX supports the overlay control error (ALERR) and high-level language interface error (XERR) routines in SYSLIB.OLB. These routines use the Breakpoint Or Message (BOM$) directive, which can simulate a breakpoint (BPT) instruction or cause the printing of an error message before task termination. One of the following error messages is returned by the BOM$ directive:

```
%RSX$S_ERRLANGIN, Error in a high level language interface
%RSX$S_ERRLODOVR, Load overlay failure
%RSX$S_INVBOMCOD, Invalid BOM$ directive parameter
```

3.10.5 Task Size

The Extend Task (EXTK$) directive increases and decreases a task's size in increments of 32-word blocks. However, on VMS, the size of a process can only be increased or decreased in 512-byte increments. VAX–11 RSX hides this discrepancy between the RSX and VMS behavior from compatibility mode tasks. The task size returned by the Get Partition Parameters (GPRT$) directive, the Get Region Parameters (GREG$) directive, and the Get Task Parameters (GTSK$) directive is in terms of 32-word blocks.

If you must determine the size of a task in 512-byte increments, round the RSX task size up to the next 512-byte boundary.

3.10.6 GPRT$ Directive

The Get Partition Parameters (GPRT$) directive allows a task to acquire information about any region known to VAX–11 RSX. Known regions include the following:

- The GEN partition, which is the task region of the current task
- A group-global section created by VAX–11 RSX
- A group-global section created by a VMS native mode image
- A system-global section created by the VMS Install Utility
- A valid image file of a resident library or common located in SYS$LIBRARY
- A valid image file of a resident library or common to which the region's name translates (if the latter is defined as a logical name)

Correction

Previously, the Get Partition Parameters (GPRT$) directive sometimes returned an incorrect value in the Directive Status Word (1 instead of 0) for regions other than the task region.

Previously, the Get Partition Parameters (GPRT$) directive would return an incorrect size for the task region.

These problems have been corrected.

3.10.7 GREG$ Directive

The Get Region Parameters (GREG$) directive returns the region address, size, and flags word in a 3-word buffer.

Previously, the Get Region Parameters (GREG$) directive would return an incorrect size for the task region.

Beginning with VAX–11 RSX Version 2.5, this directive returns the correct size for the task region.

3.10.8 QIO$ Directive

This section describes cumulative changes to the QIO$ directive and its subfunctions.

3.10.8.1 Logical Block Number

Under the RSX–11M and RSX–11M–PLUS operating systems, the logical block number (LBN) specified in P4 and P5 of an IO.RLB or IO.WLB QIO to a disk is a 24-bit quantity. The high byte of P4 is ignored. Previously, the high byte of P4 was used as part of the LBN. This caused tasks that worked correctly under RSX–11M and RSX–11M–PLUS to fail under VAX–11 RSX.

This problem corrected for VAX–11 RSX Version 2.4.

3.10.9 SPWN$ Directive

Previously, if a running task that specified MCR and a command line (for example “RUN TSK2”) spawned another task by using the Spawn (SPWN$) directive, and then the spawned task tried to read from the terminal, the subprocess suspended terminal activity. This problem has been corrected.

3.11 Indirect

This section describes the changes and additions to the Indirect Command Processor (Indirect) for VAX–11 RSX.

3.11.1 Indirect Directives

The following enhancements have been made to Indirect directives and command processing:

.ERASE SYMBOL Directive

It is now possible to delete individual local symbols with the .ERASE SYMBOL directive.

.TRANSLATE Directive

If the optional [num] parameter exceeds the number of translation iterations performed by the .TRANSLATE directive, the special string symbol <EXSTRI> contains the results of the final translation.

ATTACH Command

Previously, the ATTACH command occasionally suspended terminal activity when executed from Indirect or an MCR mode subprocess,

If you attempt to attach to the parent process of an Indirect or MCR mode subprocess, the terminal now displays the following error message:

```
%MCR-E-REFUSED, you cannot attach to this process
```

.PAUSE Directive

Previously, the Indirect .PAUSE directive could only be used from the Indirect Command Processor (Indirect) while running in the main process. This meant that the following sequence failed:

```
$  MCR
>  @TI:
IND> .PAUSE
```

This problem has been corrected.

SPAWN Command

Previously, the SPAWN command occasionally suspended terminal activity when executed from Indirect or an MCR mode subprocess. This problem has been corrected.

If you do not specify the /INPUT qualifier when executing the MCR SPAWN command from Indirect or MCR mode, the terminal now displays the following error message:

```
%MCR-E-SPWNINV, SPAWN not allowed from ICM process or
 special MCR mode
```

.TESTPARTITION Directive

The .TESTPARTITION directive allows a command file to acquire information about any region known to VAX–11 RSX. Known regions include the following:

- The GEN partition, which is the task region of the current task
- A group-global section created by VAX–11 RSX

- A group-global section created by a VMS native mode image
- A system-global section created by the VMS Install Utility
- A valid image file of a resident library or common located in SYS$LIBRARY
- A valid image file of a resident library or common to which the region's name translates (if the latter is defined as a logical name)

3.11.2 Indirect Error Messages

The following Indirect error messages are new or have not been documented before:

- .EXIT without .END
- File attributes not available

3.11.3 Multiple Indirect Command Files Using MCR Commands

Previously, executing more than two indirect command files that contained MCR commands caused the following error when the third command file issued an MCR command:

```
IND -- Spawn failure DSW = -4
```

When executing an MCR command within an indirect command file, VAX–11 RSX spawns a new process. When three or more command files were running and the third issued an MCR command, the AME attempted to create a new process by using the same process name that was assigned to the second indirect command file.

VAX–11 RSX now allows you to execute more than two indirect command files that contain MCR commands.

3.11.4 Ownership of Files and Indirect

If a rightslist identifier specifies a directory's ownership, a file created in that directory should have the identifier as its owner under certain conditions. Indirect has provided for this when a file is closed in a normal manner (for example, by using the .CLOSE directive).

Previously, closing the file during exit processing after an error (such as an Indirect substitution error) erroneously forced the file owner to be the User Identification Code (UIC) of the creator. This problem has been corrected: the rightslist identifier is the file owner, when appropriate, under either normal file closure or error exit.

3.11.5 Storing String Symbol Values

Previously, string symbol values were stored with the symbol descriptor in the symbol table area. If the length of a string symbol value changed, the symbol table area was shuffled to accommodate the new length.

Now, Indirect stores string symbol values in a special internal region that is not shuffled every time a variable is deleted or its value is changed. This new feature is designed to improve performance. To help manage the string symbol storage region, Indirect supports the following special numeric symbols:

<REGSEG> Assigns the number, in octal, of groups of free bytes in the internal string symbol storage region. This symbol can be used to detect severe fragmentation in the region, which occurs because the region is not shuffled after each addition or deletion of a variable.

<REGSIZ> Assigns the number, in octal, of free bytes in the internal string storage region.

3.11.6 Error Deleting Global Section When Indirect Exits

Beginning with VAX–11 RSX Version 2.4, Indirect creates a temporary dynamic region to hold string symbol values. Under VAX–11 RSX, regions are specific to a particular UIC group. If an Indirect command file changes the UIC to a different group, Indirect causes BUGCHECK (internal consistency failure) and DGBLERR (error deleting global section) errors when it tries to delete the dynamic region on exit.

This problem has been corrected for Indirect in VAX–11 RSX Version 2.5. However, note that, in general, regions created under VAX–11 RSX can only be accessed from the same UIC group under which they were created.

3.12 LOGIN Procedure

Previously, VAX–11 RSX running under VMS Versions 4.4 and 4.5 did not allow you to log in if a previous user had specified the Monitor Console Routine (MCR) by typing /CLI=MCR at login. The login procedure occasionally left a lock outstanding on the rights database. As a result, users could not log in or execute commands that accessed the rights database until the terminal that was using MCR had been logged out. This problem has been corrected.

3.13 MACRO–11

This section explains the changes that have been made to MACRO–11 for each version release since MACRO–11 Version 5.0. There have been no changes specific to VAX–11 RSX.

3.13.1 MACRO–11 Version 5.5 Update

The following changes were made for the Version 5.5 update of MACRO–11:

- Hexadecimal support was added. This includes the following:

  ^X, for temporary hexadecimal radix
  .RADIX 16, for permanent radix change
  .LIST HEX, to produce a hexadecimal listing

- Two new conditionals were added: .IF P1 and .IF P2.
- MACRO–11 now accepts more than 254_{10} psects, although only the first 254 appear in the symbol table. Previously, more than 254 psects caused an assembly error.
- If a .PAGE directive is issued with the listing already at top-of-page, the .PAGE directive is ignored. In prior versions of MACRO–11, the page number was incremented, even though a new physical page was not printed.
- A .PAGE directive in an unexpanded macro is now ignored.
- The XOR instruction is now flagged with the Z error for certain addressing modes (execution may be different on different processors).
- Space between a macro name and the opening left angle bracket (<) of an argument list is now optional. In prior versions, a space or tab was required. For example, if BUILD is a macro name, BUILD<A,B,C> is now valid; previously, it had to be written as follows:

  ```
  BUILD  <A,B,C>.
  ```

- In prior versions of MACRO–11, it was possible to change the value of a symbol that was assigned a value by using the =: operator, even though values assigned with =: are supposed to be permanent. MACRO–11 now retains a symbol's permanent attribute in all cases.
- .SBTTL lines in listings can now include up to 80 characters.
- The .IRPC directive now accepts arguments of up to 124 characters; the previous limit was 96 characters.
- The error message "?MACRO–F–INTERNAL ERROR" (on RT–11/RSTS) or "MAC--Internal error" (on IAS/RSX) was added. If you receive this error, please submit a Software Performance Report (SPR) to DIGITAL along with a method of duplicating the problem.

3.13.2 MACRO–11 Version 5.4 Update

In prior versions of MACRO–11, the assembler parsed the arguments of .IF conditional statements even if the .IF statements were within unsatisfied conditional code blocks. This caused assembly errors. MACRO–11 no longer parses the arguments of conditional statements within blocks of code that do not get assembled.

3.13.3 MACRO–11 Version 5.3 Update

The following changes were made for the Version 5.3 update of MACRO–11:

- Prior versions of MACRO–11 hung in an infinite loop if they encountered a record with an invalid record size. That problem has been corrected.
- MACRO–11 now fully supports VAX–11 logical names by calling the .CSI4 SYSLIB parsing routine.
- MACRO–11 did not mark symbolic expressions as complex when they contained a symbol from a relocatable psect and a symbol from an absolute psect. That resulted in incorrect linker output if the base of the absolute psect was not zero. Those symbolic expressions are now correctly marked as complex.
- MACRO–11 did not mark symbolic expressions as complex when they contained symbols from different absolute psects. That resulted in incorrect linker output if the base of either psect was not zero. Those symbolic expressions are now correctly marked as complex.
- When MACRO–11 directly assigned the current location counter symbol (.) to a global symbol in an absolute psect, MACRO–11 incorrectly bound the global symbol to the . ABS. psect. MACRO–11 now correctly binds the global symbol to the absolute psect in which the assignment occurred.
- If MACRO–11 encountered a label containing invalid characters, MACRO–11 hung in an infinite loop if there existed a macro with the same name as the valid part of the label name. MACRO–11 now correctly returns an error.

3.13.4 MACRO–11 Version 5.2 Update

The following changes were made for the Version 5.2 update of MACRO–11:

- MACRO–11 does not allow the colon (:) character as a delimiter for .ASCII or .ASCIZ strings. This is now documented in the *PDP–11 MACRO–11 Language Reference Manual*.
- MACRO–11 now provides support for the 8-bit DEC Multinational Character Set.

  The following directives support the DEC Multinational Character Set:

.ASCII	.IF DIF	.IFF IDN	.PRINT
.ASCIZ	.IF IDN	.IRP	.REM
.ERROR	.IFF	.IRPC	.PRINT
.IF	.IFF DIF	.NCHR	.PRINT

  Further information on the 8-bit DEC Multinational character set is located in the *PDP–11 MACRO–11 Language Reference Manual*.

3.13.5 MACRO–11 Version 5.1 Update

The following changes were made for the Version 5.1 update of MACRO–11:

- Previous versions of MACRO–11 exited with SUCCESS exit status even though errors were reported. That problem has been corrected.
- If MACRO–11 detected an I/O error while reading a command file, MACRO–11 produced an odd-address trap. Now, MACRO–11 reports the error message "MAC--Command I/O error."
- MACRO–11 processed some index deferred arguments as floating-point numbers by default. MACRO–11 now processes all index deferred arguments as octal by default.
- MACRO–11 did not mark internal displaced relocatable statements as relocatable with an apostrophe (′) in the assembly listing. They are now marked correctly.
- MACRO–11 did set bit 3, an unused bit, in all .PSECT object records. MACRO–11 no longer sets bit 3. That change makes object files created with the new version of MACRO–11 different from object files created with previous versions of MACRO–11. As a result, they have different Object Module Patch Utility (PAT) checksums, and a binary comparison of the files will show differences. However, the resulting task image files are the same.

3.13.6 MACRO–11 Version 5.0 Update

The following changes were made to the Version 5.0 update of MACRO–11:

- The cross-reference options SEC and ERR have been added.
- The default for the command line option /[–]SP has been modified from /SP to /–SP. The new default may be modified by the system manager by using the TKB GBLPAT option described in the MACRO–11/RSX Task Build command file.
- The op code CALLR addr (Call-Return) has been added to the permanent symbol table (PST). This op code is equivalent to the JMP addr op code. The CALLR addr op code was added to complement the CALL addr op code, which is equivalent to the JSR PC addr op code.
- The previous version of MACRO–11 used a range of 64$ to 127$ for automatic local symbol generation. MACRO–11 now uses a range of 30000$ to 65535$ when generating local symbols.
- Most assembler-generated listing text is now in uppercase and lowercase. This change was made to increase the readability of MACRO–11 code. Lines of code that include the .SBTTL or the .TITLE directive are not converted to uppercase.

- Lines of code that include the .SBTTL directive are listed in the table of contents of an assembly listing, even if a .NLIST statement is in effect at the time the .SBTTL lines are encountered. You can specify the .NLIST directive with the TOC argument to prevent the table of contents from being printed.
- The symbol table is printed at the end of an assembly, even if the .NLIST directive is in effect. You can specify the .NLIST directive with the SYM argument to prevent the symbol table from being printed.
- All page headers include the day of the week.
- The assembler statistics information that appears at the end of the assembly listing file has been updated to include the following additional information:
  - — Total number of virtual work-file reads
  - — Total number of virtual work-file writes
  - — Maximum amount of virtual memory used (in words and pages)
  - — Size of physical memory free space (in words and pages)
  - — Operating system and environment that the assembler is running under
  - — Total elapsed assembly time
  - — MACRO–11 command line
- The program section (.PSECT) synopsis that is printed after the symbol table in the listing file includes the program section attributes.
- The maximum number of relocatable terms in a complex expression has been changed. The maximum size of an OBJ record that MACRO–11 can produce was increased from 42_{10} bytes to 128_{10} bytes.

  Do not compare OBJ files that have been created by different versions of MACRO–11 when verifying whether your code generation is correct. Changes that have been made for this version of MACRO–11 (mentioned above) invalidate a direct comparison of assembler OBJ output. Verify code generation by linking or task building the OBJ files involved and then comparing the SAV or the TSK image files.

Note: OBJ files produced by this version of MACRO–11 are different from those produced by older versions. If you use the Object Module Patch Utility (PAT), checksums must be recomputed on any object patches assembled with this new version of MACRO–11.

- The default for the LC argument has been changed from .DSABL LC to .ENABL LC.
- The following .ENABL/.DSABL options have been added to MACRO–11:

  .ENABL LCM/.DSABL LCM
  .ENABL MCL/.DSABL MCL

- The following directives have been added to MACRO–11 and documented in the *PDP–11 MACRO–11 Language Reference Manual*:

.CROSS	.NOCROSS
.INCLUDE	.REM
.LIBRARY	.WEAK
.MDELETE	

3.14 Monitor Console Routine

This section describes changes to the Monitor Console Routine (MCR) for VAX–11 RSX.

3.14.1 MCR Commands

The following MCR commands have been modified:

ASN Command Modification

Previously, the MCR ASN (ASSIGN) command aborted the AME when the device object of the ASN command was associated with a $$n logical name representing an odd number (such as $$1 or $$3). The following error message was returned:

```
%SYSTEM-F-ACCVIO, access violation, reason mask=01 .....
 Improperly handled condition, image exit forced.
```

A brief dump of the stack and registers followed the message.

This problem has been corrected.

DEBUG Command Modification

The MCR DEBUG command was modified in VAX–11 RSX Version 2.5 to work correctly on VAX systems without compatibility mode hardware. After interrupting a compatibility mode image, the MCR DEBUG command should cause the image to be resumed in ODT with a T-bit error. Prior to VAX–11 RSX Version 2.5, the image was simply continued on systems using the compatibility mode software emulator.

SET and SHOW TERMINAL Command Modification

Previously, the MCR commands SET TERMINAL and SHOW TERMINAL required a device specification if these commands were issued in an indirect command file or in MCR mode. This requirement has been removed.

The SET TERMINAL and SHOW TERMINAL commands can now be issued in an indirect command file or in MCR mode without specifying the terminal device. The device specification defaults to SYS$COMMAND, which is the user terminal. However, if you assign SYS$COMMAND to a non-terminal device, the terminal displays the following error message:

```
%SYSTEM-F-IVDEVNAM, invalid device name
```

3.14.2 MCR Mode Error Messages

The following new messages are generated in MCR mode:

- If you enter an MCR command without the TMPMBX privilege, MCR mode now returns the following message:

  ```
  %SYSTEM-F-NOPRIV, no privilege for attempted operation.
  ```

- Entering MCR mode or invoking Indirect creates a subprocess with MCR as the Command Line Interpreter (CLI). If the DEFCLI or CAPTIVE flag is set in the UAF, you cannot change your default CLI by entering MCR mode or invoking Indirect. Therefore, if MCR is not the default CLI, the subprocess cannot be created. The AME now checks for this condition and displays the following error message if MCR is not the default CLI:

  ```
  %RSX-E-NOMCR, unable to spawn MCR CLI.
  ```

- Previously, an error during an MCR mode command execution occasionally caused the AME to fail and return the following message:

  ```
  %RSX-E-???????, Improperly Handled Condition
  ```

  A brief stack and registers dump followed the message. This problem has been corrected.

- The following error message is generated under certain conditions when you use the MCR ATTACH command (see 3.11.1 for details):

  ```
  %MCR-E-REFUSED, you cannot attach to this process
  ```

- The following error message is generated under certain conditions when you use the MCR SPAWN command (see 3.11.1 for details):

  ```
  %MCR-E-SPWNINV, SPAWN not allowed from ICM process or special MCR mode
  ```

- The following MCR error message was introduced with VAX–11 RSX Version 2.3:

  ```
  %RSX-E-INSUFDYN, Unable to change UIC, insufficient dynamic resources
  ```

  Explanation: After you entered the MCR command SET /UIC, MCR was not able to create the process or termination mailbox it needed to change the process UIC.

  User Action: Reenter the command.

3.14.3 MCR Mode Informational Messages

When you enter MCR mode, VAX–11 RSX displays an informational message on your terminal. It contains the name of the subprocess (spawned from DCL) that will be executing MCR commands. For example:

```
$  MCR
%RSX-S-MCRMODENTER, executing MCR commands in process "process name"
>
```

The subprocess name can be used to abort the execution of any current MCR command and delete the subprocess in which the MCR commands are executed. For example:

```
> CTRL/Y  INTERRUPT
>  STOP "process name"
```

When you exit MCR mode, VAX–11 RSX displays another informational message on your terminal. This one contains the name of the main process now in control of your terminal. For example:

```
> CTRL/Z
%RSX-S-MCRMODEXIT, control returned to process "process name"
$
```

3.14.4 SUBMIT /LOG_FILE Keyword

The SUBMIT /LOG_FILE keyword no longer requires that you specify a file. (You may, however, specify one.)

3.15 PDP–11 Emulator

The following list describes problems that have occurred when running VAX–11 RSX on a VAX processor without compatibility mode hardware or microcode (these problems have been corrected):

- When running on VAX processors that do not have compatibility mode hardware, an invalid parameter specification in the Directive Parameter Block (DPB) of a directive caused an access violation. The AME aborted with the following message:

  ```
  %SYSTEM-F-ACCVIO, compatibility mode access violation,
   reason mask=01 .....
   Improperly handled condition, image exit forced.
  ```

  A brief dump of the stack and registers followed the message.

  This problem only affected RSX–11 tasks that caused access violations.

- RSX–11 tasks that frequently used ASTs, such as the Micro/RSX File Transfer Utility (MFT), terminated with the following message:

  ```
  %SYSTEM-F-ACCVIO, compatibility mode access violation,
   reason mask=01 .....
   Improperly handled condition, image exit forced.
  ```

  A brief dump of the stack and registers followed the message.

- RSX–11 applications that frequently used trap or EMT instructions and ASTs suspended terminal activity. One such application is the RT–11 emulator on VMS (RTEM).

3.16 RMS–11

This section describes changes and corrections to the software of Record Management Services (RMS–11) on VAX–11 RSX Version 2.5.

Information in this section has not yet been incorporated into the RMS–11 manuals because RMS–11 has not changed version numbers since the last release of VAX–11 RSX.

3.16.1 Access Methods

The following list describes corrections to RMS–11 access methods:

- Previously, a record operation occasionally failed with the ER$RAB error code. This happened where an asynchronous operation was completing on one stream while another RMS operation was just beginning to execute on another stream. The correction has been made in the R0EXEC and R0EXEV modules in RMSLIB.OLB. Therefore, if you perform asynchronous record operations with RMS–11, you need to rebuild your application to incorporate this correction.
- During an RMS–11 $OPEN operation (or any similar operation), the device name was not returned in the O$DVI field of the RMS–11 NAM block when the LUN specified in the RMS–11 FAB block was used for a previous RMS–11 $OPEN operation.
- Previously, if an RMS $CREATE or $OPEN directive was issued and the IFI field in the FAB contained a non-zero value, the requested operation would fail. Similarly, an RMS $CONNECT directive would fail if the ISI field in the RAB contained a non-zero value. These problems are corrected in VAX–11 RSX Version 2.5.

3.16.2 RMSBCK Utility

This section describes changes and corrections to the RMS–11 File Backup Utility (RMSBCK).

3.16.2.1 RMSBCK Corrections

The following corrections have been made to the RMSBCK utility:

- Previously, RMSBCK failed to correctly back up more than one file to a tape container when the date qualifiers (/CD or /RD) were used. The task exit status reported failure; however, the summary listing for the file container reported no errors. After the first file, each subsequent file was not actually backed up (the header for the file was backed up, but no data were present). A summary listing would claim that all blocks were backed up, but a subsequent restore operation produced empty files. This problem has been corrected so that the date switch no longer interferes with the backup operation.
- RMSBCK failed to query the user about continuation in some cases where read/write errors were discovered. The query is now performed.

- RMSBCK skipped files whose attributes could not be backed up because of I/O errors; however, such files were reported as fully backed up. These output files are now reported correctly as empty files.
- The following problems with RMSBCK switches have been corrected:
  - — The /RC and /RA switches reported an incorrect total for the number of errors for each file.
  - — The /RC and /RA switches reported an incorrect block number for detected errors.
  - — Previously, the /QU and /NOQU switches did not function as documented. If you specified the /QU switch, RMSBCK did not enable query mode (which allows you to choose to continue or terminate the processing). Also, if you specified the /NOQU switch, RMSBCK did not terminate processing of the file.

    This problem has been corrected. The /QU switch enables query mode and the /NOQU switch terminates processing when an error is detected.

3.16.2.2 New RMSBCK Switches

The following switches were added to the RMSBCK Utility for VAX–11 RSX Version 2.0:

RMSBCK Switch	Function
/NOQU	Disables query mode
/NV	Creates a new version of the output file
/CD:date:A	Interprets the creation date as after (:A) the specified date, in normal creation/revision date format
/CD:date:B	Interprets the creation date as before (:B) the specified date, in normal creation/revision date format
/RD:date:A	Interprets the revision date as after (:A) the specified date, in normal creation/revision date format
/RD:date:B	Interprets the revision date as before (:B) the specified date, in normal creation/revision date format

3.16.3 RMSCNV Utility

The RMS–11 File Conversion Utility (RMSCNV) supports the RMS–11 /ER switch, which allows RMSCNV to continue processing a file after an exception record is encountered. This switch uses the following format:

```
/ER:filename
```

If the /ER switch is not specified, RMSCNV terminates immediately after encountering the first exception record. An error message indicating the type of exception record is returned.

If the /ER switch is specified, RMSCNV continues processing the file. If exception records are encountered, the exception records are placed in that file.

Corrections

The following problems with the RMSCNV utility have been corrected:

- RMSCNV failed when an indirect command file specified the /ER switch.
- The /EO switch worked incorrectly. RMSCNV now null fills all blocks from logical EOF to physical EOF when the /EO switch is specified.

3.16.4 RMSDEF Utility

Previously, the RMS–11 File Definition Utility (RMSDEF) always created a file with a null file name (.;1) even when a file name was specified. This occurred because the Applications Migration Executive (AME) incorrectly parsed the file name. This problem has been corrected.

3.16.5 Invoking RMSDEF From MCR

For VAX–11 RSX Version 2.5, the MCR command DEF no longer runs the RMSDEF Utility. Instead, it is the equivalent of the DCL DEFINE command. RMSDEF may be run from MCR with the following command:

```
> RUN SYS$SYSTEM:DEF
```

3.16.6 RMSDES Utility

This section lists changes that have been made to the RMS–11 File Design Utility (RMSDES).

3.16.6.1 RMSDES Enhancements

The following enhancements have been made to the RMSDES utility:

- There are two new RMSDES commands: SAVE_S[UPERSEDE] and EXIT_S[UPERSEDE]. These commands allow you to supersede a file design (see Section 3.16.12).
- RMSDES issues the warning message "%DES-W-CBK, Continuation buckets will be allocated for this key" when continuation buckets are required. (Continuation buckets may result in decreased performance; see the *RSX–11M/M–PLUS RMS–11 User's Guide*.)
- The informational message "?DES-F-VOR, value out of legal range" is issued if overflow occurs during calculations of the indexed file area allocations. Also, the "Number of duplicates:" prompt has been expanded to "Number of duplicates per record on this key:" for clarity.
- The following error message is issued if the help file LB:[1,2]RMSDES.IDX is not found: "%DES-F-NHF, help file is not available. Check release notes for the location of RMSDES.IDX on your installation media." This is a feature for small systems that have limited disk space.

3.16.6.2 RMSDES Corrections

The following corrections have been made to the RMSDES utility:

- RMSDES failed to request the key length for packed decimal keys.

  RMSDES now issues the correct prompt for packed decimal keys.

- Data area assignments for alternate keys to indexed files were sometimes incorrect when default areas were used. When there was more than one alternate key, the data area for all subsequent keys would still be the same as for the first alternate key.

  RMSDES has been modified to correctly assign data areas when alternate keys are used.

- Alternate index allocations were too large in many cases.

- Indexed file allocations were not correct when duplicates were allowed and the anticipated number of duplicates resulted in continuation buckets on alternate keys. As an added feature, RMSDES issues a warning message when continuation buckets are required. (Continuation buckets result in decreased performance; see the *RSX–11M/M–PLUS RMS–11 User's Guide*).

- Previously, RMSDES incorrectly shared channels between several commands. As a result, the GET and SAVE commands used improper default devices (where explicit devices did not exist in the design buffer or in the actual command). The default device would have been set to LB: if the HELP facility had been requested, or to either SY: or an explicit device when a CREATE command had been issued. The lack of an explicit device should have caused the GET or SAVE command to use the SY: device. This problem has been corrected.

3.16.7 RMSDSP Utility

The following switches were added to the RMS–11 File Display Utility (RMSDSP) for VAX–11 RSX Version 2.0:

RMSDSP Switch	Function
/BR	Briefly displays file attributes
/SU	Supersedes existing output file

3.16.8 RMSIFL Utility

The following problems with the RMS–11 Indexed File Load Utility (RMSIFL) have been corrected:

- The /DE switch failed to handle logicals correctly.

- If an exception record failure was reported in the first command and the /NOER switch was specified, the next command failed with the RMS error code, ER$ISI. This occurred because the sort routine did not terminate correctly.

- Previously, RMSIFL did not check the file access block (FAB) default extension quantity (DEQ) for an extension quantity if no explicit non-zero AREA XAB DEQ value had been provided. As a result, in order to set an explicit extension quantity, you had to first design areas.

  The modification to RMSIFL that corrects this problem does not change the default extension quantity; if you do not provide one, RMSIFL uses approximately 50 blocks (aligned with bucket size).

- A large number of alternate keys resulted in the RMS error code, ER$DME (dynamic memory exhausted).

- RMSIFL occasionally corrupted the alternate key root bucket. This happened when the output file allowed duplicates on an alternate key and the duplicate count field for a record straddled a block boundary. Therefore, once the file was loaded, any applications attempting alternate key access failed with ER$CHK (check byte error in bucket).

- Previously, RMSIFL failed to correctly report exit status when errors occurred while a file was loaded with more than one alternate key. Once the first alternate key had been loaded without errors, internal exit status was set to success. Any errors on subsequent keys were not reflected in the task's final exit status. This behavior (which has existed since RMSIFL was first released on VAX–11 RSX) occasionally caused command files or batch jobs to function incorrectly.

- Previously, RMSIFL would suddenly fail with RMS error -832 (ER$IFI) on an input file that had successfully loaded in the past but since then has grown in size. This problem occurred because more records existed than before and the records could not be sorted in memory. RMSIFL requires sort files for this situation. A problem with the AME, now corrected, prevented the use of sort files.

- Previously, RMSIFL, rejected a new record as out of sequence. This occurred because RMSIFL compared negative packed decimal key values as though the least significant digit represented a positive number. This problem has been corrected.

- Previously, RMSIFL did not sort duplicate user input records correctly before loading them into an output file. Now, the order of duplicate records found in an input file is maintained on insertion into the user output file. The corrected sort algorithms have a minor performance impact.

3.16.9 RMSRST Utility

This section describes changes to the RMS–11 File Restoration Utility (RMSRST).

3.16.9.1 RMSRST Switches Added

The following switches were added to the RMSRST Utility for VAX–11 RSX Version 2.0:

RMSRST Switch	Function
/NOCV	Disables version radix conversion
/NV	Creates new version of output file

3.16.9.2 RMSRST Corrections

The following problems with RMSRST have been corrected:

- RMSRST occasionally failed with a privilege violation when the /FR switch was used to restore a magnetic tape container file. If a user did not have the privilege to create the output file with the protection and ownership of the original account from which the file was backed up, RMSRST reported a failure. In reality, the file was restored, but without the correct protection codes.
- The /SE: switch incorrectly required a complete file specification (including version number). If the file was not specified, RMSRST reported that the file was not found.
- When the /SE: switch was specified, RMSRST did not terminate immediately when all files had been found.
- RMSRST did not correctly handle magnetic tape files with decimal version numbers if the files were being restored to a system that supported decimal version numbers. The version numbers were erroneously converted to octal in the newly created output files.
- In certain cases, RMSRST did not list the file version numbers as part of a summary listing.
- RMSRST failed to query the user about continuation where read/write errors were discovered.
- RMSRST did not count write errors in the summary listing. Exit status was reported as failure; however, the summary listing reported “no errors”.
- RMSRST ignored certain information when creating output files during backup operations. This caused the files to misrepresent the Longest Record Length (LRL) for the file. Some applications depend on this field to set up minimum-sized buffers. RMS–11 (or remote RMS implementations) reported “record too big for buffer” when attempting record-mode access on the file.

3.16.10 Additional Corrections to Utilities

Previously, the RMS–11 RMSDSP, RMSBCK, RMSRST, RMSCNV, and RMSIFL utilities did not accept certain characters in file specifications. The following characters caused syntax errors on one or more of these utilities:

— Logical names containing an underscore (_) or a dollar sign ($) if the logical name was not followed by a colon (:)

— The dollar sign or a number as the first character of a logical name

— Logical names beginning with a double dollar sign ($$)

— File name or type fields containing underscores, dollar signs, or hyphens (-)

These problems have been corrected.

3.16.11 Wildcard Characters in File Specifications

The RMSBCK, RMSDSP, and RMSRST utilities process file specifications that include two wildcard characters: the asterisk (*) and percent sign (%). These two wildcard characters can be used in any position within the directory, file name, file type, and file version number fields.

3.16.12 Documentation Corrections

This section corrects errors and omissions in the RMS–11 documentation set. The section titles in this section are the titles of manuals that require correction.

3.16.12.1 RSX–11M/M–PLUS RMS–11 User's Guide

Please add the following corrections to the *RSX–11M/M–PLUS RMS–11 User's Guide*:

- In Section 2.2.3.3, please add the following information to the discussion of deadlock:

  An application should use multistream rather than multichannel access to write to the same indexed file. When RMS–11 updates an RRV in a bucket that is currently locked, it must wait for that lock to be released. Control will not be returned to the program until this release occurs. Deadlock occurs when the lock is held on another channel within the same program; however, RMS–11 can update an RRV in a bucket that is locked on another stream within the same program. See your programming language documentation for details on the implementation of multistreaming.

- In Section 6.2.4, please add the following note to the discussion of writing a record:

  In the event that the record includes a partial alternate key but is not large enough to include space for the full alternate key field, RMS–11 behaves as follows:

RMS–11 treats the alternate key as if it were not present in the record, making no entry in the alternate key index structure.

- According to the *RSX–11M/M–PLUS Macro Programmer's Guide*, RMS–11 cannot perform an UPDATE operation on an alternate key with the key characteristics CHANGES and NODUPLICATES. This description is misleading. To clarify the description, please add the following information to Chapter 6, Section 6.2.5.2:

  Although RMS–11 does not support the CHANGES and NODUPLICATES combination, it does not prevent you from performing an UPDATE operation on an alternate key with these characteristics. When an update causes a duplicate of an alternate key, RMS–11 returns the completion code ER$DUP. However, it does not terminate the UPDATE operation. Instead, RMS–11 updates the primary data level for the record without updating the alternate index. As a result, the file contains duplicates of the alternate key.

  To prevent RMS–11 from creating duplicates when you make changes on alternate keys, modify your application as follows:

  1. Create the file with the key characteristics DUPLICATES and CHANGES.
  2. To disallow duplicates, perform a FIND operation on each alternate key. Then, perform an UPDATE operation on the modified record.

- In Section 6.3, please add the following information to the discussion of contiguous areas:

  You gain a small benefit by setting contiguous areas on a noncontiguous multiarea file; however, RMS–11 cannot determine if those areas remain contiguous. Consequently, RMSDSP and RMSDESAB display them as noncontiguous. As long as the areas are preallocated, they behave like contiguous areas; as soon as they need to be extended, they behave like noncontiguous areas.

- In Section 8.1.2.1, in the discussion of task building against the RMS–11 resident library, incorrect syntax is documented for the cluster option in the Task Builder command file. The correct syntax is as follows:

  ```
  CLSTR = RMSRES,DAPRES:RO
  ```

- In Appendix B, in the discussion of remote file and record access using the DECnet package, the documentation states that the RSTS/E file access listener (FAL) does not support remote record access to indexed files. This is no longer true.

3.16.12.2 RSX–11M/M–PLUS RMS–11 Macro Programmer's Guide

Please make the following corrections to the *RSX–11M/M–PLUS RMS–11 Macro Programmer's Guide*:

- In Section 2.3, the argument for P$BUF is incorrectly given as "bufcount." The correct argument is "iopoolsize," as discussed in Section 2.3.4.

- In Section 5.19, the last paragraph incorrectly describes the use of the FID field in the NAM block. It should read as follows:

"If this value is nonzero . . . "

- In Table 6–2, page 6–14, the value for the symbol XB$DAT is incorrectly given as 000003. The correct value is 000002.
- In Appendix A, page A–8, please add the following sentence:

  An attempt to insert a record that is too small to contain the whole primary key field may also cause the error ER$KEY.
- In Section A.1, add the following text to the description of the error ER$MRS:

  Or, the sum of the fixed-length record size and the record overhead exceeds the bucket size. Or, No Span Blocks has been selected with an invalid total record size.
- In Section A.1, add the following text to the description of the error ER$NOD:

  Or, the remote node rejected the operation. (STV contains the Network Services Protocol (NSP) code and can be found in Appendix C of the *DECnet-RSX Programmer's Reference Manual.*)
- In Section A.1, add the following text to the description of the error ER$FUL:

ER$FUL, Device or file allocation failure

```
Octal: 176360
Decimal: -784
```

Explanation: The specified device or directory does not have enough room for file creation or extension. In the case of a contiguous request, it is also possible that there is not enough contiguous space on the device.

3.16.12.3 RSX–11M/M–PLUS RMS–11 Utilities Manual

Please make the following corrections to the *RSX–11M/M–PLUS RMS–11 Utilities* manual:

- In Table 2–1, add the following commands:

Command	Format and Function
EXIT_SUPERSEDE	EXIT_S[UPERSEDE] filename[.typ] Names the description file in which the file design is stored.
SAVE_SUPERSEDE	SAVE_S[UPERSEDE] filename[.typ] Names the description file in which the file design is saved.

- In Section 2.2.5, add the new command EXIT_SUPERSEDE, as follows:

  The EXIT_SUPERSEDE command stores the file design in the description file specified in the command string, superseding any existing file by the same name. EXIT_SUPERSEDE then terminates RMSDESAB and returns the system prompt.

The format for the EXIT_SUPERSEDE command is as follows:

```
EXIT_S[UPERSEDE] filename[.typ]
```

EXIT_SUPERSEDE names the description file in which the file design is stored. The default file type is DE. If you do not want to supersede an existing description file, use the EXIT command.

- In Section 2.2.5, add the following statement to the description of the EXIT command:

  To supersede an existing description file, use the EXIT_SUPERSEDE command.

- In Section 2.2.9, add the new command SAVE_SUPERSEDE, as follows:

  The SAVE_SUPERSEDE command stores the file design in the description file specified in the command string, superseding any existing file by the same name. If you do not define areas when you issue the SAVE command, RMSDESAB prompts you for the areas.

  The format for the SAVE_SUPERSEDE command is as follows:

  ```
  SAVE_S[UPERSEDE] filename[.typ]
  ```

  SAVE_SUPERSEDE names the description file in which the file design is saved. The default file type is DE. If you do not want to supersede an existing description file, use the SAVE command.

  If you want to design another file, issue a CLEAR ALL command to restore the attribute values in the design buffer to their default values.

- In Section 2.2.9, add the following statement to the discussion of the SAVE command:

  To supersede an already existing description file, use the SAVE_SUPERSEDE command.

- In Section 2.6.2, add the following corrections:

  — Modify paragraph 3 of the ALLOCATION field discussion to read as follows:

    If you intend to create a single-area indexed file and do not require RSX positioning, RMS–11 uses the allocation from the file section if no area section exists in your design buffer. If you are allowing RMSDESAB to define areas for an indexed file by default, RMSDESAB automatically calculates an allocation value for each area it defines.

  — Modify paragraph 4 of the EXTENSION field discussion to read as follows:

    If you intend to create a single-area indexed file and do not require RSX positioning, RMS–11 uses the extension from the file section if no area section exists in your design buffer. If you are allowing RMSDESAB to define areas for an indexed file by default, RMSDESAB automatically calculates an extension value for each area it defines.

— Modify paragraph 5 of the BUCKETSIZE field discussion to read as follows:

If you intend to create a single-area indexed file and do not require RSX positioning, RMS–11 uses the bucket size from the file section if no area section exists in your design buffer. If you are allowing RMSDESAB to define areas for an indexed file by default, RMSDESAB assigns a bucket size value for each area it defines. However, if you choose to define areas explicitly and specify a bucket size value for each area, you should accept the default for the file section and set the bucket size value in each area section.

- In Section 2.8, add the following correction to the explanation of the error message "?DES-F-VOR":

  You entered a value in response to an attribute prompt that was not in the legal range of values for that attribute, or the values you entered resulted in a calculation that caused an overflow for RMSDESAB. If the value was not within the legal range, the error message is followed by a display of the incorrect value.

- In Table 4–1, add the following switch and description to the table of RMSCNV switches:

/ER[:filespec]	Continue processing after encountering an exception record. If a file specification is provided, then write the primary keys of exception records into the specified file. If no file specification is provided, then output the exception records to the terminal. Default: Stop processing and report RMS error code.

- In Section 4.3, add the following information to the description of RMSCNV switches:

  /ER[:filespec]
  Directs RMSCNV to continue processing when it encounters an exception record in the input file that cannot be written to the output file (see Section 3.4). If you specify a file specification, the exception records will be written to that file. If you do not specify a file specification, the primary key of each exception record is issued to the terminal. RMSCNV also issues exception record codes (see RMSIFL exception codes, Section 3.3.2).

  If you specify an exception file specification, RMSCNV creates the file as an RMS–11 Variable Fixed Control (VFC) sequential file upon encountering the first exception record. RMSCNV then writes the exception record with a 4-byte exception code to the fixed-control area of the record.

  By default, if you do not specify the /ER switch, RMSCNV stops processing upon encountering the first exception record and issues an error message indicating the type of exception record.

- In Table 5–1, add the following information to the description of RMSDSP switches:

/BR	Briefly displays attributes.
/SU	Supersedes existing output file.

- In Section 5.2, include the asterisk (*) and percent sign (%) in the description of wildcard characters permitted in the input file specification.
- In Section 5.3, add the following information to the discussion of RMSDSP commands:

  /BR
  Directs RMSDSP to issue basic displays for indexed files (see Section 5.4, Example 5–3) and container files (see Section 5.4, Example 5–6).

  /SU
  Directs RMSDSP to supersede any existing output file with the same name and version number as the output file specification. If this switch is not supplied and the version numbers are the same, RMSDSP issues the following error message:

```
?DSP-F_OPNINP, Error opening DDnn:file.dat as output
-RMS-E-ER$FEX, File already exists
```

- In Table 6–1, add the following information to the table of RMSBCK switches:

/NV	Creates a new version of the output file.

- In Section 6.2, in the discussion of RMSBCK command line format, include the asterisk (*) and percent sign (%) in the description of wildcard characters permitted in the input file specification.
- In Section 6.3.2, add the following information to the description of RMSBCK output switches:

  /NV
  Directs RMSBCK to create a new version of the disk output file if a file currently exists with the same version number as the input file. The current file is not deleted. If you do not specify this switch and a file currently exists with the same file name and version number as the input file specification, RMSBCK issues the following fatal error message:

```
?BCK-F-CREOUT, Error opening ddnn:file.dat;n as output
-RMS-E-ER$FEX, File already exists
```

- In Table 7–1, add the following information to the table of RMSRST switches:

/NV	Creates a new version of output file.

- In Section 7.2, in the discussion of the RMSRST command line format, include the asterisk (*) and percent sign (%) in the description of wildcard characters permitted in the input file specification.
- In Section 7.3.2, add the following information to the discussion of RMSRST commands:

/NV
Directs RMSRST to create the next higher version number if the expanded input file has the same version number as an existing output file. If this switch is not used and the file name and version number are the same, RMSRST issues the following error message:

```
?RST-F-CREOUT, Error opening ddnn:file.dat;n as output
-RMS-E-ER$FEX, File already exists
```

3.17 VAX–11 RSX System Library

This section describes changes to the VAX–11 RSX System Library (SYSLIB).

3.17.1 High-Level Language Interface Routines

Previously, when the high-level language interface to the CNCT, SDRC, SPAWN, or VSRC directives specified an asynchronous system trap (AST), the AST was not completed properly. This occurred because a corrupted general register was used by the interface routines. Note, however, that the MACRO–11 versions of the directives were not affected.

This problem has been corrected.

3.17.2 Named Directory Support

Tasks built with VAX–11 RSX Versions 2.2 and 2.3 failed when they were run on a PDP–11 RSX system and attempted to open a file in a named directory. This problem occurred because the File Control Services (FCS) routines in the SYSLIB.OLB file supplied with these versions of VAX–11 RSX did not include support for named directories.

This problem was corrected: the FCS routines in the SYSLIB.OLB file supplied with VAX–11 RSX Version 2.4 include support for named directories. Tasks built with this version of VAX–11 RSX now work on PDP–11 RSX systems and on VAX–11 RSX.

3.17.3 A.JUMP and A.MODE Symbols

Under VAX–11 RSX, the A.JUMP and A.MODE symbols are defined in the following files:

- FCSGBL in LB:[1,1]SYSLIB.OLB
- FSROF$ in LB:[1,1]RSXMAC.SML

3.18 Task Builder

This section describes changes to the Task Builder (TKB).

3.18.1 Building Drivers

Previously, drivers built under VAX–11 RSX sometimes failed to load on an RSX system with the error "not a valid driver task image". This happened because the default TKB switch /CM caused the driver size to be incorrectly rounded. The problem has been corrected.

3.18.2 Contiguous Space Requirements for TKB

Previously, TKB required contiguous space on the work file device (usually SYS$DISK) in order to open its work file. If contiguous space was not available, TKB failed and display the following error message:

```
TKB -- *FATAL*-Unable to open work file
```

This problem has been corrected; TKB no longer requires contiguous space on the work file device (WK).

3.18.3 Linking to Libraries

Previously, libraries built under VAX–11 RSX that linked to another library sometimes failed with the error "library references overlaid library". This happened because the default TKB switch /CM caused the library size to be incorrectly rounded. The problem has been corrected.

3.18.4 New Error Messages

The following TKB error messages are new for VAX–11 RSX Version 2.4:

- Cluster library element, element-name, is not resident overlaid
- Incompatible OTS module

3.18.5 TKB /FO Switch

TKB has the following new switch for VAX–11 RSX Version 2.4:

/FO

The /FO switch causes TKB to include the overlay run-time system (OTS) fast-mapping module FSTMAP in the task image. (See Section 3.18.6.)

3.18.6 OTS Fast Map Routine

The overlay run-time system (OTS) is a set of routines that is automatically included in any overlaid task built by TKB. One of these routines, MARKS, controls the unloading of overlays as the task executes. Another routine, the OTS Fast Map routine, uses the RSX–11M–PLUS fast-mapping facility to map windows for autoloaded memory-resident overlays. The OTS Fast Map routine is called by the MARKS routine when you specify the TKB /FO switch.

The OTS Fast Map routine enables you to use fast mapping for autoloaded memory-resident overlays, thereby increasing the speed of overlay mapping by approximately 10 times. You save time by using fast mapping whenever possible instead of the mapping directives CRAW$ and ELAW$ after the initial loading and mapping of an overlay.

Tasks that use the fast-mapping facility can be built on VAX–11 RSX and transported to an RSX–11M–PLUS system, but they are not executed successfully on VAX–11 RSX, which does not support this feature.

3.18.7 RNDSEG Option

The RNDSEG option causes TKB to round the size of a named segment up to the nearest Active Page Register (APR) boundary while building a resident library. This option is new for VAX–11 RSX Version 2.4.

3.18.8 Specifying a Task Partition

Previously, the Get Partition Parameters (GPRT$) directive always returned a successful response, consisting of a partition named GEN with a base address of 40000. If you specified a partition that did not exist in the system at taskbuild, the base address and the length of the partition were not required.

Changes made to the GPRT$ directive in Version 2.0 made it possible to get information about regions existing in the system. Therefore when you build a task and specify a partition, the Task Builder under VAX–11 RSX performs as it does under RSX–11. That means if you do not specify the base address and the length, and the partition is not a known (existent) region, the Task Builder returns the following error message:

```
TKB -- *DIAG* - Invalid partition/common specified.
```

On VAX–11 RSX, a known partition or region is a valid image file of a resident library or a resident common. The name of that partition or region must be defined as a logical name that translates to the full image file specification, if the image file resides in a user directory. (The logical name is not required if the image file resides in LB:[1,1].) See the *VAX–11 RSX Compatibility Mode Reference Manual* for more details on defining RSX–11 regions.

3.19 Terminal Driver

This section describes changes to the VAX–11 RSX terminal driver.

3.19.1 IO.ATA Function

The following is a list of extended capabilities added to the IO.ATA function:

- The IO.ATA function now supports the parameter2 value in the following format:

QIO$C IO.ATA[!TF.NOT], . . . ,<[ast], [parameter2], [ast2]>

The value of the parameter2 is passed into the high byte of the stack at each unsolicited character asynchronous system trap (AST). This feature can be used to identify terminals in a multiterminal environment.

- The IO.ATA function now supports the TF.NOT subfunction for the notification of unsolicited input. Unsolicited input causes an AST and entry into the AST service routine in the task. When the terminal driver receives unsolicited terminal input (except CTRL/C) and you use the TF.NOT subfunction with IO.ATA, the resulting AST serves only as notification of unsolicited terminal input; the terminal driver does not pass the character to the task. Upon entry to the AST service routine, the high byte of the first word on the stack identifies the terminal that caused the AST (parameter2 in the IO.ATA function).

  Using the TF.NOT subfunction allows a task to monitor more than one terminal for unsolicited input without the need to read each terminal continuously for possible unsolicited input. Note that the TF.NOT subfunction cannot be used with the CTRL/C AST (ast2 in IO.ATA); an unsolicited CTRL/C character deletes the type-ahead buffer.

  If TF.NOT is specified, after the AST has been affected, the AST becomes "disarmed" until a read request is issued by the task. A read with timeout that specifies a timeout count of zero (TF.TMO and tmo=0) move all unsolicited characters in the user's buffer. If TF.NOT is not specified with IO.ATA, every unsolicited character causes an AST.

  See the *RSX–11M–PLUS I/O Driver's Reference Manual* for more information on the TF.NOT subfunction.

3.19.2 TC.RAT Function Correction

Previously, the terminal characteristic TC.RAT functioned in a manner that was opposite to the RSX–11M–PLUS version of the characteristic. This problem has been corrected; Table 3–2 describes the current TC.RAT functions.

Table 3–2 TC.RAT Functions

TC.RAT	RSX–11M–PLUS	VAX–11 RSX
0	1-character buffer	Type-ahead disabled
1	36-character buffer	Type-ahead enabled

Users of the TC.RAT characteristic should note two minor differences between VAX–11 RSX and RSX–11M–PLUS characteristics: first, setting TC.RAT equal to zero not only sets the buffer length to one character, but also slaves a terminal and disables type-ahead capability in VAX–11 RSX. Second, if your application requires that the type-ahead buffer be set to a specific length, you can modify the values of system generation parameters TTY_TYPAHDSZ or TTY_ALTYPAHD, as described in the *VMS I/O User's Reference Volume*.

3.19.3 Terminal Characteristics for SF.GMC and SF.SMC Requests

Table 3–3 lists the terminal characteristics for SF.GMC and SF.SMC requests that were added to the VAX–11 RSX terminal driver for Version 2.1:

Table 3–3 Terminal Characteristics for SF.GMC and SF.SMC Requests

RSX–11 Bit Name	VMS Code	Meaning
TC.8BC	TT$M_EIGHTBIT	Pass eight bit on input
TC.HSY	TT$M_HOSTSYNC	Host to terminal synchronization
TC.NBR	TT$M_NOBRDCST	Disable broadcast
TC.PTH	TT2$M_PASTHRU	Pass through enable
TC.TSY	TT$M_TTSYNC	Output flow control

3.20 New VAX–11 RSX Error Messages

If you use a high-level language subroutine incorrectly or an error is detected during the loading of an overlay segment, the task terminates by means of a breakpoint (BPT) instruction. The task aborts and one of the following error messages is displayed:

```
%RSX-F-ERRLANGIN, error in high level language interface
%RSX-F-ERRLODOVR, load overlay failure
```

Both errors are followed by the following message:

```
%RSX-F-IMAGETERM, \Process name,\ terminated,
\date\ \time\
```

4 VAX–11 RSX Restrictions

This chapter describes software restrictions for the following features of VAX–11 RSX Version 2.4:

- MCR
- Task Builder (TKB)
- Backup and Restore Utility (BRU)
- Indirect
- Login command files
- Memory management on VAX–11 RSX and RSX–11
- Device names
- PIP wildcard characters
- RMS–11
- Sharing data between tasks
- Spawn directive
- SYSCOMMON system disks
- RSX–11 SYSGEN and NETGEN
- Redefining the Default Device
- Undocumented error codes

4.1 MCR SET UIC

Use of the MCR syntax for the SET/UIC=[g,m] command requires a VMS Version 5.4 or later system. RSX–11M, RSX–11S, and RSX–11M–PLUS system and network generations also require a VMS Version 5.4 or later system because they use the MCR SET /UIC=[g,m] command.

4.2 Task Builder

This section describes restrictions that apply to the Task Builder (TKB) under VAX–11 RSX.

4.2.1 Building Memory-Resident Overlays in a Co-Tree

Tasks that have memory-resident overlays in a co-tree and that are built with the default TKB switch /CM do not always have their overlay segments put on 512-byte boundaries. As a result, these tasks run correctly on VAX–11 RSX but fail when run on RSX–11M or RSX–11M–PLUS systems.

To avoid this problem, build such tasks with the TKB /-CM switch. This will allow the tasks to run on both VAX–11 RSX and RSX–11M and RSX–11M–PLUS systems.

This problem was corrected in VAX–11 RSX Version 2.3, but the correction caused undesirable side effects, so it has been withdrawn in VAX–11 RSX Version 2.4.

4.2.2 OTS Fast Map Restrictions

The OTS Fast Map routine, introduced with VAX–11 RSX Version 2.4, uses the RSX–11M–PLUS fast-mapping facility. A task must not use the IOT instruction for any purpose except fast mapping. (For more information on the fast-mapping facility, see the *RSX-11M–PLUS and Micro/RSX Executive Reference Manual*.)

Tasks that use the fast-mapping facility can be built on VAX–11 RSX to be transported to an RSX–11M–PLUS system but are not executed successfully on VAX–11 RSX, which does not support this feature.

Caution: Be sure to use both the Task Builder and system library supplied for your system to ensure that the correct OTS modules are incorporated in your task. For example, if you use the VAX–11 RSX Version 2.4 Task Builder with an incompatible (older) version of the system library to build an autoloaded overlaid task, you are issued the following new fatal error message:

TKB--*FATAL*-- Incompatible OTS module

Conversely, if you use the VAX–11 RSX Version 2.4 system library with an older version of the Task Builder, the OTS module FSTMAP is automatically included in any task built, but the module is not accessible. However, the FSTMAP module does not affect the execution or performance of your task.

4.3 Backup and Restore Utility

This section describes restrictions to the Backup and Restore Utility (BRU) on VAX–11 RSX.

4.3.1 Multidisk Restore Operation

Before you use BRU to perform a multidisk /IMAGE:RESTORE operation under VAX–11 RSX, set the PHY_IO privilege. This allows BRU to issue a physical I/O operation that sets the volume valid bit when the volume in the device is replaced.

4.3.2 APPEND Qualifier

In BRU operations with multivolume savesets, you can only use the BRU /APPEND qualifier with the first volume. This helps ensure the integrity of the backup set.

4.3.3 LENGTH Qualifier

The BRU Utility /LENGTH qualifier should not be used with cartridge tape devices such as the TK25 and TK50.

4.4 Indirect

This section describes restrictions to the Indirect Command Processor (Indirect) under VAX–11 RSX.

Indirect spawns a command line to MCR that is executed by a subprocess with the MCR CLI. Command exit status and process-context information changed by the command are moved to the main process when the command completes in the subprocess. Sections 4.4.1, 4.4.2, and 4.4.3 describe the restrictions imposed by this implementation.

4.4.1 Stopping Indirect Command Procedures

You cannot stop the execution of a CLI command by pressing CTRL/Y when the command is executed in a subprocess whose SYS$INPUT is a mailbox. Such a condition occurs in VAX–11 RSX when you perform either of the following actions:

- Run an indirect command procedure
- Enter a command in MCR mode

In either case, pressing CTRL/Y at the terminal does not stop the execution of the current command. It does, however, leave the subprocess in an indeterminate state and return control to the main process.

If you need to abort an indirect command procedure, we recommend that you also delete the subprocess in which the commands are running. The name of the subprocess, MCR or MCR.n, is displayed in an informational message after you enter MCR mode. To delete the subprocess, use the STOP command as follows:

```
$  MCR
%RSX-S-MCRMODENTER, executing MCR commands in process "MCR.1"
> CTRL/Y  INTERRUPT
>  STOP MCR.1
```

4.4.2 Logical Name Handling

The following list describes restrictions on logical name handling:

- If you redefine SYS$OUTPUT in a command procedure, the new definition remains in effect only through the execution of the command procedure. When the command procedure exits, SYS$OUTPUT reverts to the original definition specified for your process.
- If you create process-local logical name tables in a command procedure, the tables are created in the MCR subprocess but do not exist in your main process when the command procedure exits.

- If, during a command procedure, you create logical names in a process-local table (for example, a process-local table other than LNM$PROCESS), the logical names are not available after the command procedure completes. However, if you create logical names in the LNM$PROCESS table or any table that is not process-local, the logical names are available after the command procedure exits.

4.4.3 Process Context

The following list describes restrictions on the process context:

- Because MCR commands included in command procedures are actually executed in a subprocess, the output of a SHOW PROCESS command in a command procedure gives information on the subprocess. The process name, Process Identification Number (PID), and the deductible quotas are different from those of the main process in which Indirect is running. The remaining information looks the same.
- In a command procedure, the SET PROCESS /NAME command changes only the name of the subprocess being used to execute the MCR commands within the command procedure. When the command procedure exits, your main process has the same process name it had when you invoked the command procedure.
- If you include the following VMS-specific MCR commands from an indirect command procedure, changes in context are lost when the command procedure exits:
  - SET COMMAND
  - SET MESSAGE
  - SET RMS_DEFAULT
  - SET WORKING_SET

  All MCR commands included in an indirect command procedure are executed within a subprocess. Most of the commands that alter process-context information move the information to the main process to make it available after the subprocess exits. The four preceding commands do not do this.

Note: If you allocate a device during a command procedure and do not mount or deallocate the device during the procedure, the device is automatically deallocated when the command procedure exits.

4.4.4 Differences Between VAX–11 RSX and RSX–11M–PLUS Indirect

VAX–11 RSX Indirect does not support the following directives that are included in RSX–11M–PLUS Version 4.1 Indirect:

- .FORM
- .XQT
- .WAIT

The .FORM directive is not included because the FMS/RSX–11 driver is not supported under VAX–11 RSX. Complexities in correctly handling the asynchronous .XQT and .WAIT directives prevent their implementation in VAX–11 RSX Indirect.

A few Indirect directives have minimal effect in the VMS environment and, as a result, they have been set to standard values that allow most command files to operate as expected. These directives include the following:

- .IFACT/.IFNACT
- .IFINS/.IFNINS
- .IFLOA/.IFNLOA
- .TESTPARTITION
- .TESTDEVICE

The .TESTPARTITION directive can return information on RSX–11 regions, but partitions do not exist in the VMS environment.

The .TESTDEVICE directive returns the device-independent characteristics in the RSX–11 format, but the device-dependent characteristics it returns in words 2 and 3 of the device characteristics words are a VMS device-dependent longword.

A few of the special symbols are also slightly different due to the constraints and differences in the VMS environment. The following symbols, which can have varying values for RSX–11M–PLUS, have constant values for VAX–11 RSX:

- <BASLIN>
- <LOCAL>
- <PRIVIL>
- <FILER2>
- <MEMSIZ>
- <SYSTEM>
- <ACCOUN>
- <CLI>
- <FMASK>

Refer to the *VAX–11 RSX Compatibility Mode Reference Manual* for more details on indirect command file processing.

4.5 Login Command Files

This section describes restrictions on and information about login command files on VAX–11 RSX.

4.5.1 LOGIN.CMD File Nesting

The maximum nesting level of indirect command files for VAX–11 RSX is four. Since the LOGIN.CMD file is invoked at login time from a temporary indirect command file created by MCR, the nesting level in the LOGIN.CMD file cannot exceed three.

4.5.2 System Login Command File

The system logical name, SYS$SYLOGIN, points to the systemwide login command file. Unless you have changed it, SYS$SYLOGIN is defined as SYS$MANAGER:SYLOGIN. This definition causes SYLOGIN.COM to be invoked when you log in using DCL, and SYLOGIN.CMD when you log in using MCR.

You should create a systemwide login command file, SYS$MANAGER:SYLOGIN.CMD, that contains MCR and Indirect directives, to perform login functions when users log in using MCR.

4.6 Memory Management on VAX–11 RSX and RSX–11

This section describes features of VAX–11 RSX and RSX–11 memory management.

4.6.1 Compatibility Between VAX–11 RSX and RSX–11 Memory Management

Tasks that use memory management and that are developed and tested on VAX–11 RSX can be transported to RSX–11, and conversely. Also, VAX–11 RSX provides better performance for shared regions accessed as read only than it does for regions accessed for read/write operations. Therefore, tasks that do not write data in resident libraries or commons should access them as read only, by specifying any one of the following TKB options while the task is being built:

- LIBR = library:RO
- RESLIB = library/RO
- CLSTR = library:RO

4.6.2 Creating Regions

VAX–11 RSX static regions may be created in one of the following ways:

- By using the VMS Install Utility to make a compatibility mode image known to the system and by creating a global section (if the image has read-only sections or is a memory-resident library or common). For example:

```
$  INSTALL CREATE USER$:[DENNY]RESLIB.EXE/SHARED
```

Note: To create a static region with read-only access, enter the following command:

```
$  INSTALL CREATE USER$:[DENNY]RESLIB.EXE/NOWRITEABLE/SHARED
```

- By defining the name of a region as a logical name. This translates to the file specification of a compatibility mode image of a resident library or common. For example:

```
$  DEFINE RESLIB USER$:[DENNY]RESCOM.EXE
```

You can define VAX–11 RSX dynamic regions by issuing the Create Region (CRRG$) directive in a compatibility mode image.

4.6.3 Deleting Regions

VAX–11 RSX also allows you to delete a static region. To do this, use the VMS Install Utility (if the region was created by the Install Utility) by entering the following command:

```
$  INSTALL DELETE USER$:[DENNY]RESLIB.EXE
```

To deassign the logical name for the static region, enter the following command:

```
$  DEASSIGN RESCOM
```

4.6.4 Displaying Region Information

You can display information about RSX–11 regions as follows:

- Use the VMS Install Utility to display the names and the characteristics of the global sections that are associated to static or dynamic RSX–11 regions by entering either of the following commands:

```
$  INSTALL LIST /GLOB
$  INSTALL LIST /GLOB/FULL
```

Note: For compatibility with RSX–11 systems, VAX–11 RSX creates regions that have a zero length (R.GSIZ=0 in the Region Definition Block (RDB)), but it does not associate a global section with it. Therefore, you cannot display the names of such regions by using the VMS Install Utility.

- Use the DCL command SHOW to display the image files of resident libraries or commons that were defined (by logical names) as regions. The following example shows two commands you can use and the resulting screen output:

```
$  SHOW LOG RESCOM

RESCOM = USER$:[DIREC.LIBRARIES]RESCOM.EXE
$  SHOW LOG RES*

RESLIB = USER$:[DIREC.LIBRARIES]RESLIB.EXE
RESCOM = USER$:[DIREC.LIBRARIES]RESCOM.EXE
```

4.6.5 Region Checkpointing

VAX–11 RSX does not support the Checkpoint Common Region (CPCR$) directive. However, region checkpointing does exist. The only difference between VAX–11 RSX and RSX–11 checkpointing is when the checkpointing occurs (which cannot be controlled). VAX–11 RSX memory management creates a global section on behalf of the shared common region, using the compatibility mode image file as a section file. If the common region has been built with checkpointing allowed (using the /CP switch or the /CHECKPOINT qualifier) and is accessed for read/write operations, the section file is updated each time that pages from the common region are swapped out. At the end, when the common is detached and the region deleted, the section file contains all the modifications made in the common region.

Note: To ensure that a compatibility mode image file of a common region accessed for read/write operations is not being updated during the mapping session, build the common region to be non-checkpointable. For a non-checkpointable common region file, VAX–11 RSX creates a Copy on Reference global section, and the pages are swapped into the system's page file rather than into the section file.

4.7 Device Names

The following sections discuss retrictions about the use of device names and logical names in VAX–11 RSX.

4.7.1 Passing Device Names to Other Tasks

Some applications may pass information about devices (for example, SY or TI) to another task. For example, the MACRO–11 and TKB tasks pass a file specification to the CRF task when the /CR switch is supplied.

When a device name is passed to another task, that device name must be a name that the second task can use. In particular, the device name must be either a logical name that is shared by both tasks or a physical device name. If both tasks are in the same application, the device name can be a jobwide logical name.

In VAX–11 RSX, RSX–11 tasks do not often use physical device names. Because most VMS device names cannot be represented accurately in RSX–11 format, VAX–11 RSX returns a logical name if at all possible.

The logical names LB and SP are defined systemwide by the VAX–11 RSX startup command procedure and are accessible to all tasks on the system. If either name matches the VMS device name being used, VAX–11 RSX returns it to the RSX–11 task.

However, if the device name does not match either the LB or SP logical name, there is no guarantee that VAX–11 RSX will find a representation that it can use. Under these circumstances, VAX–11 RSX uses an algorithm that grants this control to the system manager and to the user. VAX–11 RSX translates the $$n logical names to find one that matches

the device. A $$n logical name can be defined in any logical name table that is included in the logical name search list, "LNM$FILE_DEV", and, therefore, may be systemwide, groupwide, jobwide, or process-local.

If no $$n logical name matches the device, VAX–11 RSX may use internal logical names, such as $I or SY. These are not actual VMS logical names; they are understood only by the AME, which assigns them. Therefore, these names should not be passed to a different task. If passed to a native mode task, the names are simply meaningless. If passed to another RSX–11 task, they will be meaningful, but probably will not mean the same thing to the second task. For example, each task has a device designated as TI; however, each task will take TI to mean its own terminal input device, even though the actual terminals may be different.

When the AME assigns a logical unit number (LUN) to SY, it attempts to find a VMS logical name (such as LB or a $$n name) that refers to the same device. If the AME finds the VMS logical name, then the RSX–11 pseudodevice name SY will have the RSX–11 physical device name LB, $$0, or whichever name was found. If the process default disk is not LB or a $$n name, the physical device name for SY is SY0.

If a $$n logical name is not defined for the process terminal, the physical name for all terminal pseudodevices (TI, CL, CO) will be $I0. If a $$n logical name is defined, it can be passed to another task provided that the $$n logical name is defined in a table to which both tasks have access. If SYS$INPUT, SYS$COMMAND, SYS$OUTPUT, or SYS$ERROR are actually files rather than a terminal device, $$n logical names cannot be used. (They cannot be shared with another task in any case.)

4.7.2 RSX–11 Device Names

On VAX–11 RSX, logical names that are used as RSX–11 device names are restricted to logical names that translate to device names or to device names and root directories. For example:

- Logical names that are valid RSX–11 device names:
  - $$0, which translates to 1DRA1:
  - $$0, which translates to DRA1:[ROOT.]
- Logical names that are not valid RSX–11 device names:
  - $$1, which translates to DRA1:[DIRECTORY]
  - $$1, which translates to DRA1:[DIRECTORY]FILE.NAME

Because of the preceding restriction, the default definition of WK has changed from SYS$SCRATCH to SYS$DISK.

4.8 PIP Wildcard Characters

Under VAX–11 RSX, the Peripheral Interchange Program (PIP) allows you to execute commands that specify wildcard characters for directories on a Files–11 Structure Level 1 disk. If you specify wildcard characters for directories on a Files–11 Structure Level 2 disk, PIP displays an error message similar to the following:

```
$  MCR PIP [*,*]/LI

PIP -- Cannot find directory file

SY0:[*,*]
```

You can use native mode utilities for file operations that require specifying wildcard characters for directories on a Files–11 Structure Level 2 disk.

4.9 RMS–11

This section describes restrictions on using the Record Management Services (RMS–11) on VAX–11 RSX. Suggestions for preventing or correcting problems are also included.

4.9.1 RMSCNV Utility

RMSCNV ignores user-provided area extension quantities when loading a file. The values that RMSCNV uses are large enough to reduce the number of file extensions in most cases.

4.9.2 RMSDES Utility

When the record size is greater than 1024 bytes, RMSDES selects area bucket sizes that are not large enough. Consequently, the file is successfully created, but is unusable. To bypass this restriction, specify the bucket size with the correct value before creating the file.

4.9.3 RMSIFL Utility

RMSIFL has the following restrictions:

- RMSIFL aborts with a memory management violation when more than one file is processed in the same RMSIFL session and the /DE switch is used. A temporary solution to this problem is to invoke RMSIFL for each file.
- RMSIFL uses a sort algorithm that, when called upon to sort on a key, does not preserve the first-in/first-out (FIFO) ordering of duplicates. For alternate keys, it is necessary to work around this problem by using RMSCNV. If your only concern is to order duplicates in the primary key, then you can use the RMSIFL /NOSO qualifier, as long as your input file is an indexed file or a file sorted on a primary key.

- When processing an indexed file with a large number of records, RMSIFL requires the use of temporary files during the sort phase. In some cases, RMSIFL aborts while attempting to create the temporary files and displays the following error message:

```
?IFL -- SORTS error code in octal: 4
?IFL -- Fatal RMS error - STS = - 832, STV=0
```

  Until RMSIFL upgrades to SORT–11 (in a future release of RMS–11), you should use the CONVERT Utility, which provides matching functionality and increased performance.

Note: Make sure you create your output file with prologue 1 or 2 if the file is to be used on a PDP–11 system.

4.9.4 Using QIOs on Assigned LUNs

In the handling of LUNs, VAX–11 RSX does not allow Queue I/O Request directives (QIOs) to be issued to devices that are assigned for RMS–11 operations. VAX–11 RSX performs a checking procedure similar to that performed by the ALUN$ directive. If there is a file accessed by RMS–11 on that device, the directive aborts and returns the error code IE.LNL (similar to the error code returned by the ALUN$ directive performed on a LUN in use for a file operation).

Some of the solutions that you can apply are as follows:

- Do not use QIOs for the same LUNs that are assigned to RMS–11 accessed devices.
- Use only RMS–11 functions on LUNs that are assigned to RMS–11 devices.

The RSX–11 TI device is not subject to this restriction.

4.10 Sharing Data Between Tasks

Applications that share data between two or more tasks through a shared region must synchronize their access to the shared region.

In a multiprocessing system, interlocked instructions can be used for synchronization. VAX processors provide interlocked native-mode instructions; however, none of the compatibility-mode instructions are interlocked. Therefore, compatibility-mode images cannot synchronize access to a shared region on a multiprocessing system through interlocked instructions.

In a single processor system, the fact that instruction execution is atomic (that is, an instruction cannot be interrupted by another task until it is complete) is useful for synchronization. Compatibility-mode instructions executed by the PDP–11 Instruction Set Emulator are not atomic, because a single compatibility-mode instruction is emulated by multiple VAX native-mode instructions. Therefore, compatibility-mode images on processors that use the PDP–11 Instruction Set Emulator cannot synchronize access to a shared region on a uniprocessor system by relying on atomic instruction execution.

4.11 Spawn Directive

This section describes restrictions on using the Spawn (SPWN$) Directive under VAX–11 RSX.

4.11.1 Spawning DCL With a Command Line

When a task issues the SPWN$ directive to spawn the DCL CLI, (DCL . . .) and includes a command line for DCL . . . to run a task, that task cannot perform any input from the terminal. VAX–11 RSX does not provide an interface between the task and terminal input. When MCR . . . is spawned, the mailbox driver serves as the interface mechanism, but when DCL . . . is spawned, a mailbox is created to provide the spawned command to DCL . . . and to stop the DCL process when the task exits. There is currently no provision for terminal input to go to the mailbox.

4.11.2 Spawning MCR With a Command Line

When a task issues the SPWN$ directive to spawn the MCR CLI (MCR . . .) and includes a command line for MCR . . . to run a task, that task is restricted from performing certain read QIOs to the terminal, as in the following examples:

- The interface between the task and terminal input is the mailbox driver. Therefore, if the task issues a QIO that is not supported by the mailbox driver, that QIO fails. For example, a one-character read does not terminate after only one character is typed but does terminate when RETURN is pressed, at which point only the first character is actually accepted.
- Another example of a QIO not supported by the mailbox driver is the SF.GMC/SF.SMC get/set multiple characteristics QIO. If the spawned task issues one of these QIOs, the QIO appears to succeed but actually has no effect. A set QIO does not change the characteristics of the terminal and after a get QIO, the contents of the receive buffer are unchanged.

Because of this restriction, you should spawn a task directly rather than spawning MCR . . . to run the task.

4.11.3 Spawning MCR With DEFCLI Set

If the DEFCLI or CAPTIVE flag is set in the User Authorization File (UAF), you cannot change your command line interpreter (CLI). Therefore, if MCR is not the default CLI and a task uses the Spawn directive to spawn MCR, the error code IE.ITS (directive inconsistent with task state) is returned.

4.12 SYSCOMMON System Disks

The Applications Migration Executive (AME) has been enhanced to support device name and file-handling operations. These operations perform compatibly with SYSCOMMON system disks or with any VMS searchlist as long as the logical name (and its translation strings) has been repetitively translated until either of the following conditions occur:

- Translations are marked with the TERMINAL attribute (LNM$M_TERMINAL).
- Strings that have no translations are reached on all branches of the search list.

Each of these final strings must contain only a physical device name, or a physical device name combined with a rooted directory name.

Note that, if a file is opened by using its file identification rather than its file name, or if a file is spooled using the PRINT$ routine, there is a slight possibility that an incorrect file will be found. This can occur if the terminal translations of the VMS logical name used as a device are on different physical devices and if the specified device identification number (DID) and file identification number (FID) exist on a device prior to the intended one. To avoid such an occurrence, DIGITAL recommends that you do not use search lists spanning multiple devices. However, file access by directory and file name (the normal case) is no more susceptible to this problem than native mode utilities are when you use search lists. In other words, if the directory and file name exist on a device prior to the intended one, both RSX–11 and VMS programs will find the same incorrect file.

4.13 RSX–11M–PLUS Version 4.3 System Generation and the Task Builder

The Task Builder (TKB) supplied with RSX–11M–PLUS Version 4.3 will cause a System Generation (SYSGEN) on VAX–11 RSX to fail because the host system on which the Task Builder (TKB) runs determines the Task Builder's behavior. On VAX–11 RSX systems, TKB assumes a default of /CM and a default task image fie type of .EXE. The System Generation fails because it expects the normal RSX defaults of /-CM and .TSK.

VAX–11 RSX Version 2.5 provides a special version of TKB, which supplies the behavior that System Generation expects, in LB:[1,54]TKB11MP43.TSK. If you are performing an RSX–11M–PLUS Version 4.3 System Generation on VAX–11 RSX, you must copy this file to your SYSGEN disk immediately before invoking SYSGEN. The following command copies the special version of TKB to the correct location on the SYSGEN disk:

```
$  COPY LB:[1,54]TKB11MP43.TSK ddnn:[3,54]TKB.TSK
```

In the preceding command line, ddnn: is the drive on which the SYSGEN disk is mounted.

4.14 RSX–11 SYSGEN and NETGEN

RSX–11M, RSX–11S, and RSX–11M–PLUS system and network generations are only supported on Files–11 Structure Level 1 target or baseline disk volumes.

RSX–11M, RSX–11S, and RSX–11M–PLUS system and network generations require VMS Version 5.4 or later versions.

4.15 Redefining the Default Device

The RSX logical names SY and WK are defined to be SYS$DISK (the device to which your default is currently set). Although you can redefine WK to be a valid device, SY is defined internally in the RSX AME and cannot be changed.

The following error typically occurs when the default device is set to a non-existent device.

```
%RSX-E-NOSUCHDEV, TKB-assigned device not found, SY0: lun=n
```

This message occurs if you erroneously redefine SYS$DISK, even though SY may be defined to be a valid device.

This is not a new restriction, but a clarification of VAX–11 RSX behavior.

4.16 Undocumented Error Codes

When you execute an RSX–11 directive (for example, SPWN$) some error codes returned by VMS system services to the AME cannot be translated to relevant RSX–11 error codes. Those error codes, such as LIB$_NOCLI, are simply translated to the RSX I/O error code, IE.VER (-4).

The Spawn directive may also return the RSX I/O error code, IE.NOD (-23), which is the translation of the VMS error code, SS$_EXQUOTA.

A Reporting Problems

This appendix describes the procedure for submitting a Software Performance Report (SPR). An SPR allows you to report any problems or questions concerning your system directly to DIGITAL.

An SPR can be used for:

- Software errors
- Documentation errors (when the Reader's Comments form is not appropriate)
- Questions
- Suggestions
- Follow-up on a previous SPR

An SPR cannot be used for:

- Software license and price policies
- Obvious hardware problems
- Logistical or clerical problems with kits, such as blank media, or failure to receive *The Software Dispatch*
- Problems with user-written software

In general, when you complete an SPR, use the following guidelines:

- Describe only one problem per form.
- Describe as accurately as possible the state of the system and the circumstances when the problem occurred.
- Illustrate the problem with specific examples.
- If the problem is a documentation error, specify the title of the manual and include the section and page number where the error occurred. Include the table or figure number if appropriate.

SPRs are assigned a priority of 1 to 5. An SPR that is assigned a priority of 1 receives the highest priority. The priorities are described as follows:

1 Most production work cannot be run.

Major system functions are unusable.
You cannot boot system.
Necessary peripherals cannot be used.

2 Some production work cannot be run.

Certain functions are unusable.
System performance has declined.
Installation does not have excess capacity.

3 All production work can be run with some user impact.

Significant manual intervention is required.
System performance has declined .
Installation has excess capacity.

4 All production work can be run with no significant impact on user.

Problem can be patched or bypassed easily.

5 No system modifications are needed to return to normal production.

SPR provides a suggestion, requests a consultation, or reports an error in the documentation.

You can submit the following categories of SPRs:

Problem/error
Contains a software problem. It is assigned a priority of 1 to 5. You receive an answer to this report.

Suggested enhancement
Contains a suggestion. It is assigned a priority of 5. You do not receive an answer to this report.

Other
Contains a suggestion or question. It is assigned a priority of 5. You may or may not receive an answer to this type of report.

Please supply the following information (in machine-readable form where applicable) when you report a problem:

System Problem	Information Needed
Crash	A copy of the output from the console terminal and the crash dump. If the crash is reproducible, please accurately describe the details and supply hardcopy or user source code when necessary.
Drivers	Controller/device information, software options, error log output, copy of device registers, and a sample program.
Utilities	A copy of your terminal output showing setup commands, before and after effects, and relevant file information.
Task Builder	A copy of your terminal output command files, the task map, and a dump of the first few blocks of the task image.
File system	Corrupted volume: output from the File Structure Verification Utility (VFY) and a dump of the volume. Improper results: the error code, a file header dump, and a sample program.

If a failure occurs when you are running privileged, add-on software, try to reproduce the failure without the additional software. Indicate on the SPR the behavior of the system with and without the add-on software.

The SPR process takes time. Therefore, if you have a critical problem, contact your local DIGITAL office.

B The DIGITAL Equipment Computer Users Society (DECUS)

DECUS, the DIGITAL Equipment Computer User's Society, is one of the largest and most active user groups in the computer industry. It is a not-for-profit association, supported and administered by DIGITAL, but actively controlled by its members. DECUS headquarters, located in Marlborough, Massachusetts, administers all international policies and activities. Members include individuals who have purchased, leased, ordered, or used a DIGITAL computer, or anyone who has a genuine interest in DECUS. Membership is free and voluntary.

B.1 DECUS Goals

The primary goals of DECUS are as follows:

- To advance the art of computation through mutual education and exchange of ideas and information
- To establish standards and to provide channels to facilitate the exchange of computer programs
- To provide feedback to DIGITAL regarding hardware and software customer needs
- To advance the effective use of DIGITAL computers, peripherals, and software by promoting the interchange of information

To further these goals, DECUS serves its members by holding symposia; by maintaining a program library; by publishing an association newsletter, technical newsletters, and books; and by supporting a number of Special Interest Groups (SIGs).

B.2 DECUS Activities

Local, regional, and national DECUS organizations give members the opportunity to meet other DIGITAL customers and employees in an informal setting. At local monthly meetings and national symposia, members can discuss their ideas, learn what others are doing, and give DIGITAL valuable feedback for future product development.

Often the national meetings in various countries also provide the stage for major new product announcements as well as a showplace for interesting developments in both hardware and software technology. At any meeting, members might describe ideas and programs they have implemented or the fine-tuning of a particular feature or application. Members present papers, participate in panel discussions, lead workshops, and conduct demonstrations for the benefit of other members.

B.2 DECUS Activities

Many of the technical papers and presentations from each symposium are published as a book, the *DECUS Proceedings*. Copies of the *DECUS Proceedings* are supplied to symposia attendees and can be purchased by DECUS members. DECUS also publishes newsletters focusing on special interests and a society newsletter.

DECUS has a Program Library, which contains over 1700 software packages that are written and submitted by users. A wide range of software is offered, including languages, editors, numerical functions, utilities, display routines, games, and other types of application software. Also available are library catalogs, which contain program descriptions and ordering information. The programs are available for a nominal service charge that covers the cost of reproduction and media.

Many DECUS members derive additional benefits from joining a DECUS Special Interest Group (SIG). SIGs often hold smaller meetings while attending regional and national meetings. They may also meet informally to discuss their special fields, such as operating systems, languages, processors, and applications. There are at least 25 SIGs in the United States alone. Many of the SIGs print newsletters and disseminate valuable technical information to members.

To obtain a membership form for DECUS, contact a DIGITAL sales representative or the nearest DECUS chapter office.

Index

A

A.JUMP symbol
  defined in VAX–11 RSX • 3–38
A.MODE symbol
  defined in VAX–11 RSX • 3–38
Account
  SYSTEM quotas • 1–2
  user quotas • 1–10
ACHN$ directive
  modifications • 3–13
Address limit
  compatibility mode • 3–7
ALLOCATION field
  RMS–11 • 3–35
Alternate record key field
  RMS–11 • 3–32
ALUN$ directive
  LUN reassignment modification • 3–12
  using template device • 3–13
ASN command • 3–23
AST
  See Asynchronous System Trap
ASTLM value
  required for SYSTEM account • 1–2
Asynchronous record operations
  using RMS–11 • 3–26
Asynchronous System Trap
  execution corrections • 3–6, 3–25, 3–38
ATRG$ directive
  modification • 3–13
ATTACH command • 3–16
AUTHORIZE utility
  modifying account quotas • 1–3

B

Backup and Restore Utility
  /APPEND qualifier restriction • 3–8
  creating UFDs • 3–9
  displaying version number • 3–9
  Files–11 Level 2 output volume
    copy restriction • 3–9
  /IDENTIFICATION qualifier restriction • 3–9
Backup and Restore Utility (Cont.)
  messages • 3–9
  multiple disk operations • 3–9
  performance enhancement • 3–9
  restoring multivolume savesets • 3–8
  /UFD qualifier • 3–9
Batch process
  name scanned for Radix-50 characters • 3–7
BIOLM value
  required for SYSTEM account • 1–2
BOM$ directive
  messages • 3–14
BRU
  See Backup and Restore Utility
BUCKETSIZE field
  RMS–11 • 3–36
BYTLM value
  required for SYSTEM account • 1–2

C

Caching
  device logical names • 3–3
CFT
  See Command Line Format
CLI
  See Command Line Interface
.CLOSE directive
  Indirect • 3–17
CMKRNL privilege
  for remove and delete procedure • 1–23
Command Line Format • 2–3
Command Line Interface
  changing in MCR mode or Indirect • 3–24
Compatibility mode hardware
  in VAX processors • 1–3
$CONNECT
  RMS–11 operation • 3–26
$CREATE
  RMS–11 operation • 3–26
Create operation
  FCS • 3–11
CTRL/Y command
  effect on compatibility mode task • 3–8

D

Data Terminal Emulator
  invoking • 2–2
  undocumented restrictions • 2–3
DCL command support • 2–1
Deadlock
  RMS–11 • 3–32
DEBUG command modification • 3–23
DECUS • B–1
DEFCLI parameter
  invalid for SYSTEM account • 1–2
Description file
  VAX–11 RSX • 1–7, 1–9
Device allocation
  RMS–11 error message • 3–34
Device logical names
  ASN command modification • 3–23
  caching and restricting • 3–3
  defined • 1–9, 3–2
  defining manually • 1–9
  for DTE device • 2–2
  passing between tasks • 4–8
  reassignment problem corrected • 3–12
Device mnemonics
  returned by GDVI$ directive • 3–10
DIOLM value
  required for SYSTEM account • 1–2
Directives, custom
  adding to VAX–11 RSX • 2–2
Directives, Executive
  See Executive directives
Directives, Indirect
  See Indirect Command Processor
Disk blocks
  contiguous
    not required for installation • 3–3
    not required for task building • 3–39
  file allocation
    RMS–11 error message • 3–34
  required for installation • 1–5
Disk Save and Compress utility • 3–10
DTE
  See Data Terminal Emulator

E

EMT instruction
  modification • 3–25
ENQLM value
  required for SYSTEM account • 1–2
/EO switch
  RMSCNV utility • 3–28
ER$FUL error • 3–34
ER$KEY error • 3–34
ER$MRS error • 3–34
ER$NOD error • 3–34
.ERASE SYMBOL directive
  Indirect • 3–16
Error code
  EI.NSF • 3–11
  IE.BDI • 3–11
Error message
  See Messages
/ER switch
  RMSCNV utility • 3–27, 3–36
Exception record, RMS–11
  See /ER switch
Executive directives • 3–12 to 3–15
  BOM$ messages • 3–14
  modified
    ACHN$ • 3–13
    ALUN$ • 3–12
    ATRG$ • 3–13
    EXTK$ • 3–14
    GPRT$ • 3–14, 3–40
    QIO$ • 3–15
    SPWN$ • 3–15
  task sizes returned • 3–14
EXIT_SUPERSEDE command
  RMSDES utility • 3–28
EXTENSION field
  RMS–11 • 3–35
EXTK$ directive
  task extension size • 3–14

F

Fast mapping
  OTS Fast Map routine • 3–39
FCS
  See File Control Services
File
  allocation
    RMS–11 error message • 3–34
  contiguous areas in RMS–11 • 3–33

File (Cont.)
ownership using Indirect • 3–17
specification
RMS–11 • 3–37
wildcards in RMS–11 • 3–37
File Control Services
common • 2–1
file creation correction • 3–11
modifications • 3–11
named directory support • 3–38
.PARSE routine modification • 3–11
Files, renamed
See VAX11 RSX
Files–11 Level 2
output volume copy restriction • 3–9
FILLM value
required for SYSTEM account • 1–2
Forms interface
MFT format • 2–3
/FO switch
TKB • 3–39
/FR qualfier
PIP command modification • 3–12
FSTMAP module
See Fast mapping

G

GBLPAGES parameter
See Global pages and sections
GBLSECTIONS parameter
See Global pages and sections
GDVI$ directive
device mnemonics returned • 3–10
GEN partition
task region • 3–14
Get Region Parameters directive • 3–15
Global pages and sections
determining • 1–4
modifying • 1–4
required for installation • 1–3, 1–4
GPRT$ directive
task size returned • 3–40
GREG$ directive
task size returned • 3–14
Group-global section
task region • 3–14

I

IE.BDI error code
modification • 3–11
IE.NSF error code
for nonexistent directory • 3–11
Indexed files
RMS–11 • 3–35
Indirect
See Indirect Command Processor
Indirect Command Processor • 3–16
accessing regions • 3–16
changing CLI in procedure • 3–24
deleting local symbols • 3–16
directives
.CLOSE • 3–17
.ERASE SYMBOL • 3–16
.PAUSE • 3–16
.TESTPARTITION • 3–16
.TRANSLATE • 3–16
error messages • 3–17
executing multiple command files • 3–17
file ownership • 3–17
general description • 4–3
logical name handling • 4–3
modifications • 3–15 to 3–18
process context • 4–4
process-local logical names
creating • 4–3
reassigning LUN • 3–12
restrictions • 4–3 to 4–5
stopping • 4–3
storing string values • 3–18
symbols
REGSEG • 3–18
REGSIZ • 3–18
using ATTACH command • 3–16
using RMSCNV /ER switch • 3–28
using SHOW TERMINAL command • 3–23
using SPAWN command • 3–16
Installation
contiguous space not required • 3–3
distribution media • 1–1
example • 1–13 to 1–17
messages • 1–18 to 1–21
on cluster • 1–10
on independent processor • 1–3
performing • 1–6 to 1–9
save sets • 1–1
SETPRV privilege required • 1–6

Installation (Cont.)
steps after • 1–9 to 1–10
steps before • 1–2 to 1–6
SYSTEM account values • 1–2
upgrade • 1–1, 1–11
Installation Verification Procedure • 1–8
messages • 1–21 to 1–22
running after installation • 3–3
running on cluster • 1–10
IVP
See Installation Verification Procedure

L

Library, system
See System Library
License, VAX–11 RSX
See VAX–11 RSX
License Management Facility • 3–5
LMF
See License Management Facility
Logical name tables
process-local
creating in command procedure • 4–3
Logical Unit Number
reassigning • 3–12
LOG_IO privilege
to define terminal attributes • 2–2
LOG_IO privilege, required
to invoke DTE • 2–2
LUN
See Logical Unit Number

M

MACRO–11
modifications • 3–18 to 3–23
MARKS
TKB overlay unloading routine • 3–39
MCR
See Monitor Console Routine
MCR mode
changing CLI in • 3–24
stopping CLI command • 4–3
MCR support for DCL commands • 2–1
Memory management
RSX–11 compatibility • 4–6
Messages
BOM$ directive • 3–14
BRU utility • 3–9
Indirect • 3–17
installation • 1–18 to 1–21
IVP • 1–21 to 1–22
MCR • 3–24
RMS–11 • 3–28
RSX$REMOVE_AND_DELETE.COM • 1–22 to 1–24
TKB • 3–39
VAX–11 RSX • 3–42, 4–14
format • 1–17
MFT
See Micro/RSX File Transfer Utility
Micro/RSX File Transfer Utility • 2–3
supersedes PFT • 2–3
Monitor Console Routine
commands
ASN • 3–23
ATTACH • 3–16
DEBUG • 3–23
SET TERMINAL • 3–23
SHOW TERMINAL • 3–23
SPAWN • 3–16
SUBMIT/LOG_FILE keyword • 3–25
help library • 2–3
installing • 1–8
messages
error • 3–24
informational • 3–24
modifications • 3–23 to 3–25
RSX images on VMS • 3–4
MS device mnemonic • 3–10
MU device mnemonic • 3–10
Multiple disk operations
BRU • 3–9
Multivolume saveset
BRU restriction • 3–8
restoring using BRU • 3–8

N

$$n logical names
See Device logical names
/NOCV switch
RMSRST utility • 3–31
/NV switch
RMSRST utility • 3–31

O

Online release notes • 1–2
$OPEN
  RMS–11 operation • 3–26
Open operation
  FCS • 3–11
OTS
  TKB overlay mapping routine • 3–39
Overlay
  memory resident
    and disk resident • 3–5
    problem corrected • 3–7
  run-time system routines • 3–39

P

Page length
  modifying systemwide value • 1–10
.PARSE routine
  FCS • 3–11
Partition
  defined • 3–40
.PAUSE directive
  Indirect • 3–16
PDP–11 Instruction Set Emulator
  corrections • 3–25
  VAX processors • 1–3
Peripheral Exchange Program
  /FR command modification • 3–12
  /PU command modification • 3–12
PFT
  See Profession File Transfer Utility
PIP
  See Peripheral Interchange Program
Privileges
  See VMS privileges
Process names • 3–6
Profession File Transfer Utility
  superseded by MFT • 2–3
/PU qualifier
  PIP command modification • 3–12

Q

QIO$ directive
  subfunctions
    IO.ATA • 3–40
    IO.RLB • 3–15
    IO.WLB • 3–15
    SF.GMC • 3–42
    SF.SMC • 3–42
Quotas
  SYSTEM account • 1–2
  user account • 1–10

R

R.MLB macro library
  restoring • 2–2
R0EXEC
  RMS–11 programming module • 2–4, 3–26
R0EXEV
  RMS–11 programming module • 2–5, 2–6, 3–26
R0EXSV
  RMS–11 programming module • 2–5
R0EXSY
  RMS–11 programming module • 2–4
Radix-50 characters
  in batch process names • 3–7
Record Management Services
  access methods • 3–26
  asynchronous record operations • 3–26
  building applications • 2–4 to 2–6
  building applications on VAX–11 only • 2–5
  $CONNECT operation • 3–26
  $CREATE operation • 3–26
  deadlock • 3–32
  documentation corrections • 3–32 to 3–38
  errors
    ER$FUL • 3–34
    ER$MRS • 3–34
    ER$NOD • 3–34
  EXTENSION field • 3–35
  fields
    ALLOCATION • 3–35
    alternate record key • 3–32
    BUCKETSIZE • 3–36
  files • 2–4
    contiguity • 3–33
    indexed, creating • 3–35
    specifications • 3–32, 3–37
  libraries
    memory resident • 2–6 to 2–7
    object • 2–4

Record Management Services
libraries (Cont.)
resident, task building against • 3–33
modifications • 3–26 to 3–38
new messages • 3–28
$OPEN operation • 3–26
task building against resident library • 3–33
UPDATE operation • 3–33
utilities
RMSBCK • 3–26, 3–32
RMSCNV • 3–27, 3–32, 4–10
RMSDEF • 3–28
RMSDES • 3–28, 4–10
RMSDSP • 3–29, 3–32
RMSIFL • 3–29, 3–32, 4–10
RMSRST • 3–30, 3–32
Region
checkpointing • 4–8
creating • 4–6
deleting • 4–7
displaying information • 4–7
Region parameters • 3–15
Regions
accessed in Indirect • 3–16
REGSEG symbol
Indirect string storage region • 3–18
REGSIZ symbol
Indirect string storage region • 3–18
Release notes
location • 1–9
online • 1–2
Reporting problems
See Software performance reports
RMS–11
See Record Management Services
RMS–11 to RMS 32 translator • 2–5
RMSBCK utility • 3–26, 3–32
command line format • 3–37
new switches • 3–27
/NV switch • 3–37
wildcards • 3–32
RMSCNV utility • 3–27, 3–32, 4–10
/ER switch • 3–27, 3–36
RMSDEF utility • 3–28
RMSDESAB utility
file contiguity • 3–33
RMSDES utility • 3–28, 4–10
new commands • 3–28
RMSDSP utility • 3–29, 3–32
/BR switch • 3–36
file contiguity • 3–33
RMSDSP utility (Cont.)
/SU switch • 3–36
wildcards • 3–32
RMSIFL utility • 3–29, 3–32, 4–10
RMSRES.EXE
installed at startup • 2–7
RMSRST utility • 3–30, 3–32
command line format • 3–37
new switches • 3–31
/NV switch • 3–37
wildcards • 3–32
RNDSEG
TKB option • 3–40
RSX$DEVICE_CACHE logical name • 1–10
modifying • 3–3
RSX$DEVICE_NAMES.COM
created • 3–2
modifying • 1–9
RSX$REMOVE_AND_DELETE.COM • 3–4
copying • 1–11
messages • 1–22 to 1–24
privileges required • 1–23
RSX$STARTUP.COM
in system startup • 1–9
invoking on cluster • 1–10
RSX–11M/M–PLUS RMS–11 Macro Programmer's Guide
corrections • 3–33
RSX–11M/M–PLUS RMS–11 User's Guide
corrections • 3–32
RSX–11M/M–PLUS RMS–11 Utilities Manual
corrections • 3–34
RSX–11 task
running on VAX–11 RSX • 2–1
transporting to VAX–11 RSX • 2–6
RSX compatible disks
VMS class required • 1–2

S

Save sets, VAX–11 RSX installation • 1–1
SAVE_SUPERSEDE command
RMSDES utility • 3–28
SETPRV privilege
to install VAX–11 RSX • 1–6
SET TERMINAL command
See SHOW TERMINAL command
SHOW TERMINAL command
default device • 3–23
SLP

SLP (Cont.)
See Source Language Input Program
Software performance reports • A–1
Source Language Input Program • 1–8
SPAWN command
/INPUT qualifier required • 3–16
SPR
See Software performance reports
SPWN$ directive
modification • 3–15
String values
stored by Indirect • 3–18
SUBMIT command
LOG_FILE keyword • 3–25
Synonym directories
obsolete • 3–4
SYS$COMMAND
SHOW TERMINAL default • 3–23
SYS$ERROR definition
correction • 3–8
SYS$LIBRARY
referencing files in • 3–4
SYS$LP_LINES page length value
modifying • 1–10
SYS$MESSAGE
referencing files in • 3–4
SYS$OUTPUT
redefining • 4–3
SYS$SYSTEM
defined by VAX–11 RSX • 3–4
referencing files in • 3–4
SYSCOMMON system disks • 4–13
SYSLIB
See System Library
SYSLIB.OLB
named directory support • 3–38
SYSNAM privilege
for remove and delete procedure • 1–23
SYSPRV privilege
for remove and delete procedure • 1–23
SYSTARTUP_V5.COM
modifying • 1–9
SYSTEM account values required • 1–2
System Generation • 4–13
System-global section
task region • 3–14
System Library
compatibility with Task Builder • 4–2
modifications • 3–38

T

Task
building
FCS problem correction • 3–11, 3–38
partition, specifying • 3–40
regions • 3–14, 3–15
segment rounding • 3–40
size
extension, required by EXTK$ directive • 3–14
real • 3–14
resultant • 3–14
with memory and disk overlays
loading • 3–5
Task Builder • 4–13
compatible library requirement • 4–2
contiguous space not required • 3–39
error messages • 3–39
/FO switch • 3–39
link problem correction • 3–39
loading error correction • 3–39
modifications • 3–38 to 3–40
OTS run time system • 3–39
RMS–11 resident library • 3–33
RNDSEG option • 3–40
Task names • 3–6
TC.RAT
terminal driver function • 3–41
Terminal driver • 3–40
requesting multiple characteristics • 3–42
subfunctions
IO.ATA • 3–40
TC.RAT • 3–41
.TESTPARTITION directive
Indirect • 3–16
TK50 tape drive • 3–10
TKB
See Task Builder
.TRANSLATE directive
Indirect • 3–16
TRAP instruction
modification • 3–25
TS11 tape drive • 3–10
TU80 tape drive • 3–10

U

UFD

UFD (Cont.)
  See User File Directory
UPDATE operation
  RMS–11 modification • 3–33
Upgrading VAX–11 RSX
  removing old version • 1–11
  restrictions • 1–1, 1–11
User account quotas • 1–10
User File Directory
  created by BRU • 3–9

V

VAX–11 RSX
  Executive directives
    modifications to • 3–12 to 3–15
  files
    listing • 1–9
    renamed • 3–2
    startup procedure • 1–9
  license
    loading on cluster • 1–10
    registering • 1–8
  login procedure modification • 3–18
  messages • 1–17 to 1–24
  regions • 3–16
  RMS–11 to RMS 32 translator • 2–5
  running RSX–11M–PLUS tasks • 2–1
VAX11RSX$DEVICE_NAMES.COM
  renamed • 3–2
VAX11RSX.COM
  disabling • 1–9
  renamed • 3–2
VAX CoProcessor/RSX
  FCS common with RSX • 2–1
VAX processors
  with Emulator • 1–3
VMS classes, required • 1–2
VMS privileges, required
  for installation • 1–6
  for remove and delete procedure • 1–23
  to define terminal attributes • 2–2
  to invoke DTE • 2–2
VMS versions
  compatible with VAX–11 RSX • 1–1

W

Wildcard characters
  in RMS–11 file specifications • 3–32