RSX-11M-PLUS and Micro/RSX System Management Guide

Order No. AA–JS14A–TC

digital™

software

RSX–11M–PLUS and Micro/RSX System Management Guide

Order No. AA–JS14A–TC

RSX–11M–PLUS Version 4.0
Micro/RSX Version 4.0

Digital Equipment Corporation Maynard, Massachusetts

First Printing, October 1979
Revised, November 1981
Updated, April 1983
Revised, July 1985
Revised, September 1987

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

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

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

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

The following are trademarks of Digital Equipment Corporation:

DEC
DEC/CMS
DEC/MMS
DECnet
DECsystem–10
DECSYSTEM–20
DECUS
DECwriter
DIBOL
EduSystem
IAS
MASSBUS
MicroPDP–11
Micro/RSX
PDP
PDT
RSTS
RSX
UNIBUS
VAX
VAXcluster
VMS
VT

ZK3085

HOW TO ORDER ADDITIONAL DOCUMENTATION
DIRECT MAIL ORDERS

USA & PUERTO RICO*

Digital Equipment Corporation
P.O. Box CS2008
Nashua, New Hampshire 03061

CANADA

Digital Equipment
of Canada Ltd.
100 Herzberg Road
Kanata, Ontario K2K 2A6
Attn: Direct Order Desk

INTERNATIONAL

Digital Equipment Corporation
PSG Business Manager
c/o Digital's local subsidiary
or approved distributor

In Continental USA and Puerto Rico call 800-258-1710.
In New Hampshire, Alaska, and Hawaii call 603-884-6660.
In Canada call 800-267-6215.
*Any prepaid order from Puerto Rico must be placed with the local Digital subsidiary (809-754-7575).
Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Westminster, Massachusetts 01473.

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

Contents

Part I: Setting Up and Shutting Down the System

Chapter 1 Introduction to System Management

Chapter 2 The Account File Maintenance Program

Chapter 3 Setting Up and Running the Queue Manager

Chapter 4 System Reconfiguration

Chapter 5 Virtual Monitor Console Routine (RSX-11M-PLUS Systems Only)

Chapter 6 The SHUTUP Program

Part II: Resource Monitoring and Control

Chapter 7 Resource Monitoring Display

Chapter 8 Memory Management

Chapter 9 The Console Logger (RSX-11M-PLUS Systems Only)

Chapter 10 Resource Accounting

Part III: System Problems and Performance

Chapter 11 Handling System Problems

Chapter 12 The I/O Exerciser

Chapter 13 The Bad Block Replacement Control Task

Chapter 14 I/O Queue Optimization

Chapter 15 Disk Data Caching

Part IV: Advanced Features

Chapter 16 Shadow Recording (RSX-11M-PLUS Systems Only)

Chapter 17 Command Line Interpreters

Chapter 18 The DCL Task

Chapter 19 Using the Catchall Task

Appendix A RSX-11M-PLUS UFD Conventions

Index

Examples

Figures

Tables

Preface

Manual Objectives

The *RSX-11M-PLUS and Micro/RSX System Management Guide* describes system programs and procedures that test, monitor, maintain, and customize RSX-11M-PLUS and Micro/RSX operating systems after system generation.

Intended Audience

This manual is written for system managers, operators, and system programmers who perform system management functions. It is not intended to be a tutorial for general users.

Structure of This Document

The *RSX-11M-PLUS and Micro/RSX System Management Guide* is divided into four parts: Setting Up and Shutting Down the System, Resource Monitoring and Control, System Problems and Performance, and Advanced Features. In addition, Appendix A contains a description of User File Directory (UFD) conventions.

Part I, Setting Up and Shutting Down the System, contains Chapters 1 to 6. Together, they provide essential information for setting up accounts, queues, and peripheral devices on your system. In addition, Part I includes a description of the system shutdown procedure (SHUTUP).

Chapter 1 provides a general overview of system management and serves as a guide to the remainder of the manual. It includes descriptions of general system concepts and utilities.

Chapter 2 describes, with examples, how to use the Account File Maintenance Program (ACNT) to create and maintain user accounts.

Chapter 3 describes how to set up the Queue Manager (QMG) for multistream line-printer spooling and batch processing.

Chapter 4 describes how to use the reconfiguration services (CON and HRC) to modify your system configuration. If you have an RSX–11M–PLUS operating system, you can isolate the system from the effects of faulty hardware by using CON and HRC. This chapter also includes a description of the Autoconfigure task (ACFPRE.TSK) supplied with pregenerated RSX–11M–PLUS and Micro/RSX systems.

Chapter 5 describes how to use the Virtual Monitor Console Routine (VMR) to alter a system image file on disk. You can modify parts of the system without repeating the system generation procedure by using VMR. This chapter applies to RSX–11M–PLUS operating systems only (excluding pregenerated systems).

Chapter 6 describes how to use the SHUTUP Program to shut down the system in an orderly fashion.

Part II, Resource Monitoring and Control, contains Chapters 7 to 10. After you set up your system, read Part II to learn how to monitor system usage and how to ensure a fair allocation of resources.

Chapter 7 describes how to use the Resource Monitoring Display (RMD) to display information about active tasks and available resources in your system.

Chapter 8 describes the tasks available for managing memory resources. It includes information on the Pool Monitor Task (PMT), secondary pool, and the Shuffler (SHF) task. However, note that PMT is supplied with RSX–11M–PLUS operating systems only (excluding pregenerated systems).

Chapter 9 describes how to use the Console Logger to control I/O to the console output device (CO) and to record system messages at a terminal or in a log file. This chapter applies to RSX–11M–PLUS operating systems only (excluding pregenerated systems).

Chapter 10 describes Resource Accounting, which creates a transaction file of system usage information.

Part III, System Problems and Performance, contains Chapters 11 to 15. These chapters contain information for correcting hardware-related problems and for improving system performance.

Chapter 11 provides an overview of the terminology and tools associated with detecting and correcting problems with your hardware. It also serves as an introduction to the rest of the chapters in Part III.

Chapter 12 describes how to use the I/O Exerciser (IOX) to detect and diagnose problems on the disk, terminal, and tape units in your system's hardware configuration.

Chapter 13 describes how the Bad Block Replacement Control Task (RCT) performs bad block handling and recovery on Mass Storage Control Protocol (MSCP) devices such as the RA81.

Chapter 14 describes I/O queue optimization, which improves the throughput of the disk subsystems used in your system.

Chapter 15 describes disk data caching, a feature that enhances I/O operations by reducing the number of physical I/O requests performed during disk operations.

Part IV, Advanced Features, contains Chapters 16 to 19. When you are familiar with the operating system, read Part IV to learn how to customize your system to fit your particular needs.

Chapter 16 describes Shadow Recording (SHA), which allows your system to maintain a copy of all information being written to a Files–11 disk. This chapter applies to RSX–11M–PLUS operating systems only (excluding pregenerated systems).

Chapter 17 describes command line interpreter (CLI) tasks, including how to write a CLI.

Chapter 18 describes how to task build and install the DIGITAL Command Language (DCL) task with various options. It also describes the Macro Metalanguage (MML), a set of macros that performs the mapping from DCL syntax to Monitor Console Routine (MCR) syntax. With this language, you can alter or add commands to extend DCL. This chapter applies to RSX–11M–PLUS operating systems only (excluding pregenerated systems).

Chapter 19 describes a catchall task (TDX). TDX allows you to run uninstalled tasks and abbreviate command names.

Appendix A describes the set of User File Directory (UFD) conventions observed by the operating system for files on disk. A section of file naming conventions is also included.

Associated Documents

Before you read this manual, you should be familiar with the following manuals:

- The *RSX–11M–PLUS MCR Operations Manual* provides information on the Monitor Console Routine (MCR) commands
- The *RSX–11M–PLUS Command Language Manual* and the *Micro/RSX User's Guide, Volumes 1 and 2,* provide information on the DIGITAL Command Language (DCL)
- The *RSX–11M–PLUS Utilities Manual* describes the utilities supported by the RSX–11M–PLUS and Micro/RSX operating systems

Although you do not need to read the *RSX–11M–PLUS System Generation and Installation Guide* before reading this manual, system generation logically precedes your use of the programs and procedures documented in this manual. In addition, the *RSX–11M–PLUS System Generation and Installation Guide* contains information on the system startup procedure (STARTUP.CMD).

Conventions Used in This Document

The following conventions are used in this manual:

Convention	Meaning
SET	Uppercase letters and words indicate that you should enter the letter or word exactly as shown.
file name	Lowercase letters and words indicate that you are to substitute a word or value. In command descriptions, lowercase letters indicate a variable whose actual value is determined when the command is entered or the message is issued. For example, the value of the variable task name depends on the name of the task associated with the command or message.
n_8 or #n n_{10} or n.	In general, the base of a number is indicated as a subscript to the number. For example: 77777_8 or 255_{10}. However, in command descriptions and examples, decimal values may be indicated by a trailing period (.). Octal values may be indicated either by the absence of a period or by a number sign character (#) preceding the number. For example, the following three values are equal: 255. 377 #377 Unless specified otherwise, all numeric values required in a command can be entered as decimal or octal.
[, . . .]	Square brackets around a comma and an ellipsis mark 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. Square brackets are also part of the User Identification Code (UIC) and User File Directory (UFD) syntax (for example, [group,member] or [name]). When you use a UIC or a UFD (in a file specification, for example), you must include the brackets.
{ }	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).
( )	Parentheses are used to enclose more than one argument in a command line. For example: `SET /CACHE=DUO:(DEFER_WRITES,DIRECTORY,LOGICAL)`

Convention	Meaning
@	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]`
. . .	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.
black ink	In examples, what the system prints or displays is printed in black.
	In interactive examples, what the user types is printed in red. System responses appear in black.
blue ink	Text in blue ink indicates that the information pertains to RSX-11M-PLUS multiprocessor systems only.
$\boxed{\text{xxx}}$	A symbol with a 1- to 3-character abbreviation indicates that you press a key on the terminal. For example, $\boxed{\text{RET}}$ indicates the RETURN key, $\boxed{\text{LF}}$ indicates the LINE FEED key, and $\boxed{\text{DEL}}$ indicates the DELETE key.
$\boxed{\text{CTRL/a}}$	The symbol $\boxed{\text{CTRL/a}}$ means that you are to press the key marked CTRL while pressing another key. Thus, $\boxed{\text{CTRL/Z}}$ indicates that you are to press the CTRL key and the Z key simultaneously. $\boxed{\text{CTRL/Z}}$ is echoed on some terminals as ^Z. However, not all control characters echo.
	Examples of terminal display in this manual show control key sequences as they are displayed on the terminal screen.

Summary of Technical Changes

The following sections list new and modified features for the RSX–11M–PLUS and Micro/RSX Version 4.0 operating systems. These new features are documented in this revision of the *RSX–11M–PLUS and Micro/RSX System Management Guide.*

New and Modified Features

The RSX–11M–PLUS and Micro/RSX Version 4.0 operating systems have the following new features:

- New VMR INSTALL keyword /DFB
- New VMR SET keyword /CHAR_LENGTH
- Data caching default values
- Deferred disk write requests (DFR) for data caching
- Larger extent sizes for data caching

New INSTALL Command Keyword /DFB

The INSTALL command keyword /DFB=YES allows the binding of Executive device logical unit number (LUN) assignments to be deferred until a task is loaded into memory. The /DFB=NO keyword allows binding to occur only at the time the task is installed.

The default is /DFB=YES.

New SET Command Keyword /CHAR_LENGTH

The SET command keyword /CHAR_LENGTH establishes the length of characters transmitted and received by terminals attached to the system through variable-speed multiplexers.

You can specify the character length as either 7 or 8 bits. If you do not specify a length, VMR displays the current setting. The default value for transmission is 8-bit characters.

This keyword is not valid for serial-line units.

Data Caching Default Values

Prior to this release, the SET /CACHE command used default values for any of the options not explicitly called out in the command line. Now, the SET command sets the default values for the data caching options at the time the cache is activated for a device. Subsequent commands change only the options specified in the command line; all other options retain the values established when the cache was activated.

Deferred Disk Write Requests for Data Caching

Deferred disk write requests (DFRs) are now a data caching option. If data caching is selected at system generation, you can enable DFRs for temporary files when you mount a disk or when you set device characteristics. DFRs are a useful way of speeding up I/O operations for temporary files on your system disk or on disks that are relatively slow writing data.

You can select the option by using one of the following commands:

Larger Extent Sizes for Data Caching

The maximum size of an I/O request for data caching has been increased from 15 blocks to 127 blocks. The following options support the larger extent size:

- [NO]DIRECTORY
- [NO]LOGICAL
- [NO]OVERLAY
- [NO]READ_AHEAD
- [NO]VIRTUAL

Note that extent sizes can be modified for cached devices by using the SET or MOUNT commands.

Changes to the Document

The following changes in organization are included in this revision of the *RSX-11M-PLUS and Micro/RSX System Management Guide*:

- File Transfer and Terminal Emulation (moved)
- Shuffler Task (moved)
- Secondary Pool (new addition)
- Catchall Task (new addition)

For a detailed description of the new organization for this manual, see the previous section entitled "Structure of This Document."

File Transfer and Terminal Emulation

Information on the Micro/RSX File Transfer utility (MFT) and the Data Terminal Emulator (DTE) is now contained in the *RSX-11M-PLUS Utilities Manual.*

Shuffler Task

Previously, the Shuffler (SHF) task was described in its own chapter. Information on the Shuffler is now contained in Chapter 8.

Secondary Pool

Information on secondary pool support is now provided in Chapter 8.

Catchall Task

Information on a Catchall Task (TDX), which is supplied with RSX-11M-PLUS and Micro/RSX operating systems, is now provided in Chapter 19.

Part I: Setting Up and Shutting Down the System

Chapter 1

Introduction to System Management

This chapter provides an introduction to the concepts and procedures associated with managing an RSX–11M–PLUS operating system or a Micro/RSX operating system that has the Advanced Programmer's Kit installed.

1.1 General System Concepts

Before you begin to set up and to modify your operating system, you should understand the features that it supports. The topics covered in this section will help you to understand the interaction of various system components and to optimize your system's operation after a system generation.

1.1.1 Memory Partitions and Regions

Partitions are contiguous areas of physical memory that are managed and dynamically allocated to regions by the Executive. Each partition accommodates as many tasks, device drivers, and common regions as possible at any one time.

The Executive brings a task into memory for execution and places the task in competition with other active tasks. When tasks are brought into memory, they are placed in a section of memory called a subpartition or "region." *Regions* are contiguous areas of memory in which executable tasks run. *Common regions* are code libraries and data commons that can be shared by tasks for more efficient use of physical memory. A task can run in any region large enough to contain it.

1.1.2 Primary and Secondary Pool

Primary pool (usually referred to simply as "pool") occupies part of the region in memory allocated to the Executive. When you run a task or perform some other action that requires pool, the system allocates pool space in units of memory called "packets" or "nodes." However, because the Executive (and consequently, pool) is permanently loaded in low memory, primary pool does not appear in the list of available partitions.

Primary pool contains the Executive's database and temporary data structures that are used by tasks while they execute. Because nearly all Executive functions require pool, a system can exhaust pool when the level of system activity is high (for example, if too many tasks are installed or too many volumes are mounted at the same time). When pool is depleted, the system does not appear to have failed; however, it does not function normally. If there are data lights on the front of the processor, they may flicker, but the system does not accept input. Under these conditions, the system often cannot display error messages because the Executive requires pool space to perform I/O. Once a system exhausts pool, you must restart the system.

Note

> You can prevent pool from being depleted in two ways. First, take the maximum amount of pool compatible with your system configuration during system generation. Second, use the Pool Monitor Task (PMT) and the Resource Monitoring Display (RMD) to monitor pool, and take appropriate action before pool gets too low. (For more information, see Chapter 8.)

Secondary pool is allocated from main memory and is extensible. However, when you extend secondary pool, you do so at the expense of main memory. Each time you increase the amount of secondary pool space, you decrease the amount of available memory. As a system manager, you need to create an appropriate balance between the two resources.

For more information on managing primary and secondary pool, see Chapter 8.

1.1.3 Memory Allocation

Instead of allocating a fixed amount of memory for specific requirements, the Executive allocates memory space as it is needed by waiting regions, as follows:

1. Uses a priority level of 1 to 250_{10} to determine a waiting region's use of a partition. (A higher number indicates a higher priority.) Each time a region is accessed, it is merged into a priority-ordered queue of regions waiting to be loaded into the partition.
2. Begins at the base of the partition and searches for a free, contiguous area large enough for the waiting region. The Executive examines the list of allocated areas in the partition. To calculate the size of the gaps between the areas, it orders the list of allocated areas according to their physical memory addresses.
3. Allocates space within the partition to a waiting region. When the Executive locates a sufficiently large gap, it allocates the space to a waiting region. To record the allocation, the Executive merges the Partition Control Block (PCB) for the region into the list of allocated areas.
4. Eliminates the region to free memory when the task is done executing. Each time a region exits and frees memory in the partition, the Executive examines the partition wait-queue and tries to allocate memory to the highest-priority waiting region.

In some instances, a waiting region cannot be loaded into a partition because there is not enough memory space available. The Executive keeps track of outstanding I/O on a region-by-region basis; if there is no I/O pending, the Executive moves common regions and task regions to increase the available memory space.

1.1.3.1 Checkpointing

Checkpointing preempts a lower-priority region when a higher-priority region can be brought in to make use of the freed memory. This optimizes the use of system resources while maintaining a priority-scheduling discipline.

Checkpointing in a partition occurs as the result of a *memory allocation failure*. That is, the Executive tries to allocate a contiguous area of a partition to a region and it cannot find an unoccupied memory area of sufficient size. When this happens, the Executive reexamines the list of allocated areas in the partition to determine whether it can form a free space of sufficient size by checkpointing one or more neighboring regions. Each region considered for checkpointing must be of a lower priority, it must be checkpointable, and it must have checkpointing enabled.

The Executive scans the list from the beginning, looking for a series of neighboring regions (possibly separated by gaps of free space), where each region satisfies the checkpoint criteria. If the memory occupied by a series of regions and free space satisfies the memory requirement for the higher-priority region, the Executive moves or *checkpoints* the lower-priority regions from memory to disk. Then, the Executive allocates the released memory to the higher-priority region.

The checkpointed region is immediately placed in competition with other regions waiting for residency. When the checkpointed region is reloaded, tasks mapped to that region continue processing from the point where they were interrupted.

1.1.3.2 Partition Fragmentation

Checkpointing cannot always free enough memory space to load a waiting region. This is a result of *partition fragmentation* and can be caused by any of the following events:

- Regions of various sizes entering and exiting memory.
- Device drivers loaded in the middle of a partition. Device drivers, and other regions which cannot be moved, divide a partition into two *partition sections*.
- Regions with nonbuffered, outstanding I/O.

  The Executive must wait for a region's I/O count to reach zero before the region can be checkpointed. While the Executive waits, tasks that map the region are blocked from execution. This prevents those tasks from issuing more I/O requests, which could delay the loading of the waiting region.
- Regions that are connected to the IP11 driver.
- Regions that are connected to an interrupt vector by the Executive directive Connect to Interrupt Vector (CINT$).
- Regions that have been fixed by the Executive for a memory parity error.

If the Executive is unable to load a waiting region, the Executive requests the Shuffler (SHF). The Shuffler moves regions in partitions to create the free space necessary to load a waiting region. The Executive requests the Shuffler only when all other attempts to load a waiting region fail.

1.1.3.3 Memory Compaction

The Shuffler is a privileged task designed to compact space in partitions when a memory allocation failure occurs. When the Executive is unable to load a waiting region into a partition, the Executive requests the Shuffler. The Shuffler searches memory for a partition that has at least one region waiting for space. When the Shuffler locates such a partition, it attempts to move or "shuffle" regions in the partition to create contiguous space.

The Shuffler shuffles common regions created by Executive memory management directives. Because the system maintains an outstanding I/O count on a region-by-region basis, the Shuffler can determine whether outstanding I/O is pending. When there is no I/O pending, common regions can be shuffled.

For more information on the Shuffler, see Chapter 8.

1.1.3.4 Disk Swapping

Swapping varies task priorities so that tasks of equal or close priority checkpoint each other periodically. This allows such tasks to share a portion of memory.

Swapping does not alter the checkpointing algorithm because a task can only checkpoint another task of lower priority; it cannot checkpoint a task of equal or higher priority. The priority with which you install a task (the *installed priority*) remains constant. However, the *memory priority* of tasks varies over time. The *memory priority* determines the order in which tasks are swapped to and from memory.

There are two parameters that control the swapping algorithm.

Parameters

Swapping interval

Determines how often the Executive scans the partition lists to modify the memory priority of resident tasks. A typical swapping interval is 0.5 second.

Swapping priority range

Specifies the absolute value of the range through which a task's priority varies from its installed priority. A typical value is 5. This value causes a task's memory priority to vary from P+5 to P–5, where P is the priority set for the task when it is installed.

On RSX–11M–PLUS operating systems, these parameters are set by commands in the command file SYSVMR.CMD. You can change them by using the MCR or VMR commands SET /SWPC and SET /SWPR. For a description of the MCR commands, see the *RSX-11M-PLUS MCR Operations Manual*. For a description of the VMR commands, see Chapter 5 of this manual.

The key element of the swapping algorithm is a byte in the task's header that maintains the swapping priority of the task. In a system that uses swapping, the Executive determines whether or not a nonresident task should checkpoint a resident task by making a comparison. It compares the running priority of the nonresident task with the sum of the running and swapping priorities (called the *effective priority*) of the resident task. If the running priority of the nonresident task is greater than the effective priority of the resident task, then the resident task is checkpointed.

Each time a task is read into memory (as the result of an initial task load or checkpoint read request), the Executive initializes the swapping priority byte in the task header. At each swapping interval, the swapping priority of each resident task is reduced by 1, until it reaches its lowest priority. If there is a possibility that checkpointing within a main partition might occur based on the new priorities, the Executive executes its partition allocation algorithm for that partition.

When you specify the swapping parameters, consider how your system is used. For example, in a highly interactive system, a high checkpoint rate may reduce the need for swapping. Editors and other interactive tasks usually run at a higher priority than tasks devoted to computations; however, they may also be checkpointed more frequently. While such tasks wait for terminal input, they can be checkpointed out of memory by lower-priority tasks. Then, when the input is complete, the interactive task can return to memory by checkpointing the lower-priority task.

In general, from the time a task is loaded into memory, the average time for the task to be checkpointed by another task of the same running priority is roughly equal to the product of the two swapping parameters (swapping interval and swapping priority range). Tasks with the same running priority tend to get the same amount of time in memory. Tasks whose running priorities differ by less than the swapping priority range tend to receive different amounts of time in memory, with the higher priority tasks getting more time.

1.1.4 Dynamic Checkpoint Space Allocation

Dynamic checkpoint space allocation allows all task checkpointing operations to use common system checkpoint files. Thus, you can install tasks as checkpointable without having to allocate checkpoint space in the task image file, and you can run multiple copies of a checkpointable task. By allocating checkpoint space dynamically, the system reduces the number of tasks that require memory residency at any given time.

To improve memory usage in large systems, make all or most of the tasks in the system checkpointable. The DCL command SET DEVICE /CHECKPOINT_FILE (MCR command Allocate Checkpoint Space or ACS) establishes and eliminates system checkpoint files. You can establish one checkpoint file for each mounted Files–11 volume.

When the Executive requires checkpoint space, it selects devices in the order in which you allocate the checkpoint files. Therefore, if you issue multiple SET DEVICE (ACS) commands, specify the fastest devices first. For example, a system may have both an RS04 fixed-head disk and an RP06 movable-head disk. If you determine that limited checkpoint space (400 blocks) can be allocated on the RS04, allocate additional space (2000 blocks) on the RP06 to ensure that checkpoint allocation failures do not occur.

As the Executive allocates and deallocates checkpoint space, the checkpoint files can fill up and become fragmented. When possible, the Executive reuses the restored checkpoint space. However, depending on the sizes of the checkpoint files and the amount of fragmentation, the Executive may fail to find space to fulfill a checkpoint request. In this case, a task must wait until additional memory or checkpoint space becomes available.

If a critical task must get into memory, it may be unacceptable for a checkpoint operation to fail. To ensure that any given task can always be checkpointed, allocate checkpoint space in the task image file. Then, if the Executive fails to find space for the task in the system checkpoint files, it can checkpoint the task to the preallocated space. There can be no checkpoint allocation failures in partitions in which all active tasks have preallocated checkpoint space.

Note

> This does not apply to tasks that use external headers (that is, headers that are placed in a reserved area immediately below the task image, instead of in pool). External headers require system checkpoint space.

For more information on the SET DEVICE command, see the *RSX-11M-PLUS Command Language Manual* or the *Micro/RSX User's Guide, Volume 1*. For information on the ACS command, see the *RSX-11M-PLUS MCR Operations Manual*.

1.1.5 Task Extension

The Executive directive Extend Task (EXTK$) allows tasks to increase or decrease their memory allocation during execution. For example, the MACRO-11 Relocatable Assembler (MAC) and the Task Builder (TKB) use the EXTK$ directive to maintain a memory-resident symbol table, until the table reaches a maximum size limit.

To modify the maximum size limit, use the DCL command SET SYSTEM /EXTENSION_LIMIT (MCR command SET /MAXEXT). To reduce the initial memory requirements for these tasks, install them with a small increment, such as 2048_{10}. For large assemblies and task builds, these utilities increase their size to the maximum limit, which also increases their speed. Then, before processing the next command, they return to their original size.

1.1.6 Round-Robin Scheduler

The *round-robin scheduler* ensures that all tasks of equal priority share the central processing unit (CPU). The scheduler alternates between *CPU-bound* tasks of equal priority (that is, tasks that have relatively high requirements for CPU time) and schedules tasks dedicated to I/O operations to run before any of the CPU-bound tasks. However, note that this affects only those tasks currently in memory (tasks that are in the Active Task List); tasks that are checkpointed or in the partition wait queue are not affected.

The scheduler provides access to the CPU for a predetermined length of time measured in *ticks*. A *tick* is a clock interrupt. The rate at which interrupts occur depends on the type of clock installed in your system. For a line frequency clock, the tick rate is either 50 or 60 per second, depending on the line frequency. For a programmable clock, a maximum of 1000 ticks is available; you can select the actual frequency during system generation (see the *RSX-11M-PLUS System Generation and Installation Guide*). To modify the time interval used by the round-robin scheduler, use the SET /RNDC command.

Round-robin scheduling does not affect the standard processor competition among tasks of different priorities. It also does not affect tasks that are not in the specified priority range. To modify the high and low limits of the priority range considered for scheduling, use the SET /RNDH and SET /RNDL commands, respectively.

1.1.7 Parity Support

If the Executive detects a main memory parity error within a task region, the Executive attempts to declare a parity error asynchronous system trap (AST) to that task. If the parity error occurs within a common region, the Executive attempts to declare a parity error AST to each task mapped to the shared region. In either case, if a task to which a parity error AST is declared does not have an AST recovery routine, the Executive aborts the task and issues a message to let you know that the task has been aborted.

When the Executive detects an error in a region, it locks the region into memory. This prevents that space from being reused. The Executive then invokes the FIXER task, which probes the region until it finds the segment containing the error.

When a parity error is detected within the Executive, or if the Executive detects a parity error while reading to or writing from a region, the system attempts to print the following message before halting:

```
***EXEC PARITY ERROR STOP***
```

For cache parity errors on the PDP–11/20, the half of the cache in which the parity error is detected may be disabled. If two errors in the same half are detected within the same minute, the cache is disabled. If less than one error per minute is detected and the system recovers by reading through to main memory, the cache is not disabled. When either one or both halves of the cache are disabled, the system operates in a degraded state. However, errors continue to be logged.

1.1.8 Multiple File Control Processors

For RSX–11M–PLUS and Micro/RSX systems with several high-performance disks, you can increase file system throughput by mounting each volume with a separate Files–11 Ancillary Control Processor (F11ACP). Multiple file control processors enhance the response to device requests by reducing the need for physical I/O operations each time a disk device is requested.

The F11ACP buffers the bit map and Master File Directory (MFD) for one device at a time. If you mount every volume with the same F11ACP, the F11ACP clears the buffer each time you request a different volume. Then, to obtain the bit map and MFD for a requested volume, the F11ACP must perform physical I/O to the disk. In contrast, if you mount each volume with a separate F11ACP, the ACP reads the bit map and MFD into memory only once. Subsequent requests to a volume are directed to the appropriate ACP, which buffers the bit map and MFD for that volume.

See the *RSX–11M–PLUS Command Language Manual*, the *RSX–11M–PLUS MCR Operations Manual*, or the *Micro/RSX User's Guide, Volume 1*, for information on the commands used to mount volumes with separate F11ACPs.

1.1.9 File System Options

By using the DCL commands INITIALIZE and MOUNT (MCR commands INI and MOU, respectively), you can modify file system parameters such as the default extension block count, the index file position, the memory buffers, and the mapping pointer count. (For more information on these commands, see the *RSX-11M-PLUS Command Language Manual*, the *RSX-11M-PLUS MCR Operations Manual*, or the *Micro/RSX User's Guide, Volume 1*.)

The following list describes some of the available options (MCR commands appear in parentheses):

- Default extension block count: MOUNT /EXTENSION (MOU /EXT) or INITIALIZE /EXTENSION (INI /EXT)

  When a disk file is created or extended, File Control Services (FCS) allocates a default number of additional disk blocks to the file (usually 5 blocks). For example, if a task writes 12 blocks of data in a file, three 5-block allocations are made, and the remaining three blocks are unused. You can use the /EXTENSION qualifier (/EXT keyword) to change the default number of disk blocks allocated.

  Each allocation of disk blocks requires several disk accesses to find free space, to allocate the space, and to mark the space as used in the file header. Therefore, if you increase the default above five blocks, you decrease the number of disk accesses needed to write files. In addition, you increase the number of unused disk blocks allocated while the file is open. Conversely, if you decrease the default below five blocks, you decrease the number of blocks allocated and not used and increase the number of file-system overhead disk accesses performed when extending files. (When the file is closed, however, the file system truncates the file to the number of blocks actually used.)

  A task can override this default extension block count when creating or extending a file. In addition, you can use the Peripheral Interchange Program (PIP) switch /TR to free unused space at the end of a file. Application tasks can override the file system defaults and perform optimal file extensions for the processing they perform. You can use these techniques, possibly with the /EXTENSION qualifier (/EXT keyword), to provide efficient disk allocation. (For more information on PIP, see the *RSX-11M-PLUS Utilities Manual*.)

- Index file position: INITIALIZE /INDEX (INI /INDX)

  The position of the index file in large volumes is important because an increase in the *seek time* (that is, the time it takes to search for the file) can decrease performance. Rather than having the index file at the beginning or end of a volume, position it at the midpoint of the volume (by using either the block number or the MIDDLE argument).

  For small volumes, such as a diskette, putting the index file at the midpoint limits the maximum size of the work files on the disk. In such cases, position the index file at the beginning or end of the volume.

- Memory buffers to speed directory searches: MOUNT /ACCESSED (MOU /LRU) or INITIALIZE /ACCESSED (INI /LRU)

  The /ACCESSED qualifier (/LRU keyword) specifies the number of buffers to be maintained in memory. The default is three buffers. The buffers contain only the most recently accessed directories; so, if your application is working with a small number of directories, three buffers may be sufficient. However, if many directories are being scanned frequently, access time will improve if you specify a number greater than three.

Note

> If the number of File Control Blocks (FCBs) and the number of directories in the least-recently-used (LRU) list exceeds the number of FCBs preallocated in the F11ACP, then maintaining extra buffers in memory will require additional pool space.

- Mapping pointer count: MOUNT /WINDOW (MOU /WIN) or INITIALIZE /WINDOW (INI /WIN)

  The /WINDOW qualifier (/WIN keyword) specifies the number of mapping pointers to be allocated for file windows. The default is seven pointers. These pointers refer to, or "point to," contiguous blocks of the file on the disk. By increasing the number of pointers, you can optimize access to fragmented files. However, note that the additional pointers require secondary pool space. If you need to increase the amount of space available in secondary pool, initialize the disk with fewer pointers to files with little or no fragmentation.

In addition, the MCR command SAVE has a /MOU keyword that allows you to use the MOUNT options on the system disk. For more information on the /MOU keyword, see the *RSX-11M-PLUS MCR Operations Manual.*

1.1.10 Overlapped I/O Completion

Overlapped I/O completion support causes the execution of I/O completion code for each I/O request to be postponed until the next request has been initiated. If I/O requests are in the driver's queue, the Executive completes the I/O processing while the physical device services the next request. Thus, if the last request in the driver's queue does not require a device interrupt (that is, if it occurs synchronously), multiple I/O requests to the same device may complete in an order other than the order in which they were requested.

1.2 Installing and Generating Your System

To install an RSX-11M-PLUS system, refer to the *RSX-11M-PLUS System Generation and Installation Guide*. The guide includes information on generating, bootstrapping, and saving your system. To install a Micro/RSX system with the Advanced Programmer's Kit, refer to the following manuals:

- *Micro/RSX Base Kit Installation Guide for Diskettes* or the *Micro/RSX Base Kit Installation Guide for Tape*
- *Micro/RSX Advanced Programmer's Kit Installation Guide for Diskettes* or the *Micro/RSX Advanced Programmer's Kit Installation Guide for Tape*

1.3 Setting Up Your System

After performing system generation, you can modify your system to suit your application. The following sections provide a brief overview of the files, tasks, and utilites that are available for establishing your system environment.

1.3.1 Using System Management Files

RSX-11M-PLUS and Micro/RSX operating systems are supplied with a variety of system management files that you can use to set up your system. In particular, refer to the *RSX-11M-PLUS System Generation and Installation Guide* or the *Micro/RSX System Manager's Guide* for information on the following files:

File Specification	Description
LB:[1,2]STARTUP.CMD	The system startup command file is invoked each time you boot the system. By using MCR commands in the STARTUP.CMD file, you can install tasks or perform other system initialization duties. To execute parts of the file conditionally, include commands to the Indirect Command Processer. (For information on Indirect, see the *RSX-11M-PLUS Indirect Command Processor Manual* or the *Micro/RSX User's Guide, Volume 1*.) If you have an RSX-11M-PLUS system, note that you can also use VMR to install tasks and change terminal characteristics (see Section 1.3.5). If you have a pregenerated RSX-11M-PLUS system (RL02 distribution kits) or a Micro/RSX system, modify SYSPARAM.DAT instead of STARTUP.CMD (see the following description).
LB:[1,2]SYSPARAM.DAT	The system configuration data file contains statements that are used by STARTUP.CMD. By modifying this file, you can alter the system startup parameters for pregenerated RSX-11M-PLUS and Micro/RSX systems.
LB:[1,54]SYSVMR.CMD	(RSX-11M-PLUS operating systems only.) The VMR command file installs certain privileged and nonprivileged tasks when you create the system image file during system generation (SYSGEN). By editing SYSVMR.CMD, you can modify parts of the system without having to repeat SYSGEN.
LB:[1,2]SYSLOGIN.CMD SYS$LOGIN:LOGIN.CMD	The system and user login command files, respectively, execute automatically each time a user logs in to the system. To set up a particular environment each time an account is used, modify the login command files. Note that LOGIN.CMD executes from the user's default device and directory (SYS$LOGIN).
LB:[1,2]SYSLOGOUT.CMD SYS$LOGIN:LOGOUT.CMD	The system and user logout command files, respectively, execute automatically each time a user logs out of the system. You can use the SYSLOGOUT.CMD file to purge files and to delete temporary files.
LB:[1,2]LOGIN.TXT	The login information file is displayed by the HELLO (LOGIN) task when you log in to the system. Edit this file to provide installation news and notices to system users.

File Specification	Description
LB:[1,2]BATCH.TXT	The batch job information file is included at the beginning of each batch job log file.
Online help files	Files with the file type HLP are located in directory [1,2]. They provide information on commands, tasks, and utilities. You can modify these files to add information that is specific to your system or application.

The *RSX-11M-PLUS System Generation and Installation Guide* and *Micro/RSX System Manager's Guide* include information on designing your own system management command files. Also, note that you can use command files to create new system commands. (For more information, see the description of the catchall task in Chapter 19.)

1.3.2 Using the Account File Maintenance Program

The Account File Maintenance Program (ACNT) allows you to create and maintain an account file. Each account entry contains a variety of information, such as an account number and password, the default device (SY), the user's first and last name, and the default command line interpreter (CLI). By using ACNT, you can add and delete accounts, modify accounts, list the accounts, examine an individual account, and sort the account entries by User Identification Code (UIC).

Chapter 2 provides examples and additional information on using ACNT.

1.3.3 Setting Up Printers and Batch Queues

The Queue Manager (QMG) controls the system printers and the print jobs that are passed to them. You can specify the number of copies to be printed, the time the job is to be released for printing, whether the file is to be deleted after printing, and which printer you want to use. Also, you can write your own output processors, which pass jobs to devices other than line printers. QMG displays information about the queues and alters the characteristics of jobs in the queues.

QMG can support as many as 16 output queues (including the default PRINT queue) and 16 output processors. You can initialize the print processors with specific attributes, such as printing a specified number of flag pages before each print job, accepting or rejecting print jobs that require lowercase printing, and handling different types of printed forms.

QMG also supports batch processing. A system can have as many as 16 batch queues and 16 batch processors.

Chapter 3 describes how to set up and maintain QMG. For information on using QMG for print and batch processing operations, see the *RSX-11M-PLUS Batch and Queue Operations Manual* or the *Micro/RSX User's Guide, Volume 1.*

1.3.4 Modifying Your System Configuration

The reconfiguration services consist of two tasks (CON and HRC) and a driver (RD:). With these services, you can alter or display information about various system resources, such as devices and memory. The main purpose of the reconfiguration services is to allow you to isolate faulty hardware so that it does not affect the system adversely.

Chapter 4 describes the functions of CON, HRC, and RD:. It includes descriptions of the reconfiguration commands and examples of how to use them.

If you are using a Micro/RSX or pregenerated RSX–11M–PLUS system, you can change your system configuration by modifying the file LB:[1,2]ACFPAR.DAT. For more information, see Chapter 4.

1.3.5 Modifying the System Image File (RSX–11M–PLUS Systems Only)

The Virtual Monitor Console Routine (VMR) is a task that accepts a subset of the Monitor Console Routine (MCR) commands. You use VMR to make the same changes to the system image file on disk that you would make to the running system with MCR. For example, you can set the size of pool, create partitions in memory, load drivers, install tasks, or change terminal characteristics. By using VMR, you can configure part of a system image file before you bootstrap it; then, as soon as you bootstrap the system, it will be ready for use.

Chapter 5 describes VMR and presents each of the VMR commands in alphabetical order.

1.4 Shutting Down the System

SHUTUP allows you to shut down the system in an orderly fashion. Although turning the system off does not damage the software, it can affect your application programs and files. Before halting the system, SHUTUP performs cleanup functions, such as the following:

- Logging out all logged-in terminals
- Submitting the user-written command file LB:[1,2]SHUTUP.CMD to Indirect for execution
- Stopping QMG, Console Logger, and Error Logger (if present)
- Deallocating checkpoint space
- Dismounting mounted devices

If some of the terminals on your system are located at a distance from the processor and the console terminal, or if the system is used for many different purposes, run SHUTUP before you turn off the system or make backup copies of files. To reduce the possibility of interrupting users, SHUTUP issues warning messages before logging them out of the system. When you run SHUTUP, it prompts you for the number of minutes to wait before shutdown, the number of minutes between shutdown messages, and the number of minutes to wait before disabling logins.

Chapter 6 contains more information and an example of the SHUTUP program.

1.5 Monitoring and Controlling System Resources

As soon as your system is set up and running, begin monitoring system usage. Then, when you become familiar with the needs of the people who are using the system, modify the availability of resources to correspond to their needs.

1.5.1 Resource Monitoring Display

The Resource Monitoring Display (RMD) provides information about the active tasks in your operating system and the availability of system resources. This information includes a list of the active tasks, their location in memory, the amount of memory they occupy, and the available pool space. RMD generates dynamic displays on video terminals and "snapshot" displays on hardcopy terminals.

The following displays are available:

- Memory
- Active Task
- Task Header
- System Statistics
- I/O Count
- General Cache
- Detailed Cache

RMD also supports a Help display that tells you how to switch display pages. Each display page has a setup page from which you can alter the contents of the associated display page. You can access any RMD display and alter the contents of the display from the MCR or DCL command line.

Chapter 7 contains detailed information on RMD and includes examples of RMD displays.

1.5.2 Pool Monitoring Support (RSX-11M-PLUS Systems Only)

Pool monitoring support controls the use of the system's pool. This support monitors pool levels, restricts use of pool, and notifies you when pool is near depletion. This support consists of two parts: the RSX-11M-PLUS Executive pool monitor code and the privileged Pool Monitor Task (PMT).

Chapter 8 describes how to enable pool monitoring support, the conditions that affect PMT, PMT defaults, and how to abort PMT. It also includes an annotated example of PMT output for extreme fragmentation and information on modifying secondary pool.

Micro/RSX systems and pregenerated RSX-11M-PLUS systems (RL02 distribution kits) do not include PMT.

1.5.3 The Console Logger (RSX-11M-PLUS Systems Only)

The Console Logger consists of a driver (CODRV) and a task (COT...). Together, they handle I/O to the console output device (CO) and record time-stamped system messages on a terminal, a log file, or both. If you have an RSX-11M-PLUS distribution kit (excluding RL02 distribution kits), you can select Console Logger support during system generation.

By using the MCR command SET from a privileged terminal, you can start and stop Console Logging, disable the log file or the console terminal, and reassign the log file and console terminal. Chapter 9 describes the Console Logging commands and presents sample terminal sessions.

1.5.4 Resource Accounting

Resource Accounting provides a transaction file of system usage information. Information is gathered on each user in a system and on total system utilization. This information can be used to bill individual users for the resources used and to measure overall system performance.

By analyzing the data in the transaction file, you can obtain statistics about individual users (such as the amount of CPU time a person uses) or overall system usage (such as the number of users logged in on the system per day). The information in the transaction file can be displayed, or you can write a program to access the file and analyze the data.

Chapter 10 describes how Resource Accounting works and contains an example of transaction file output.

1.6 Preventing the Loss of Information

RSX-11M-PLUS and Micro/RSX provide several utilities for preventing the loss or corruption of data and for maintaining peripheral devices. The sections that follow describe these utilities briefly and provide references to additional sources of information.

1.6.1 Backing Up Files

BACKUP is a DCL command that allows you to back up from and restore files to Files-11 volumes. Its purpose is to safeguard against the loss or corruption of data. When you backup a volume, you produce a copy that can be used if any of the files on the original volume are lost or corrupted. For more information on the BACKUP command, see the *RSX-11M-PLUS Command Language Manual* or the *Micro/RSX System Manager's Guide*.

To perform backup and restore operations, the BACKUP command uses the Backup and Restore Utility (BRU). In the process of copying, BRU also reorganizes and compresses files for efficient storage and use. For more information on BRU, see the *RSX-11M-PLUS Utilities Manual*.

1.6.2 Shadow Recording (RSX-11M-PLUS Systems Only)

Shadow Recording (SHA) allows your system to back up all information as it is being written to a Files-11 disk. It does so by creating two identical disks called a *shadowed pair.* More than one pair of disks may be shadowed, but shadowed disk pairs cannot overlap. The first disk of the pair, the primary disk, is the original disk that exists whether or not Shadow Recording is active. You can make any disk on your system, including the system disk, the primary disk of a shadowed pair.

You can use Shadow Recording to produce an online or "hot" backup copy of your system disk. During Shadow Recording, the Executive writes the same data to the secondary disk that it writes to the primary disk. However, the Executive reads the primary disk first. Then, if a read error occurs on the primary disk, the Executive reads the secondary disk. Thus, by duplicating your system disk, you are able to prevent loss of data if a disk error occurs. In addition, you can recover from an error faster because you do not need to halt the system to correct the errors. All of this occurs transparently to you.

Chapter 16 describes Shadow Recording, how to prepare your system for it, and how to control it. The chapter also provides information on the Shadow Recording commands and error messages.

1.7 Detecting and Correcting Hardware Problems

To detect and resolve problems associated with the devices in the hardware configuration, you can use a combination of the following: Error Logging, the I/O Exerciser (IOX), and, for DU-type disks, the Bad Block Replacement Control Task (RCT).

1.7.1 Error Logging

The Error Logging System monitors the performance of hardware devices. Error Logging handles mass-storage device (disk and tape) errors and memory errors. Since Error Logging is part of the operating system, it is most effective for hardware errors that allow the system to continue functioning.

Note

> To use Error Logging on a Micro/RSX system, you must install the Extended Error Logging option.

For more information on Error Logging, see the *RSX-11M-PLUS and Micro/RSX Error Logging Manual.*

1.7.2 The I/O Exerciser

The I/O Exerciser (IOX) detects I/O problems on the disk, tape, and terminal units in your system's hardware configuration. IOX exercises Files-11 disks, non-file-structured disks, magnetic tapes, DECtapes, and cassettes.

The IOX Command Language allows you to specify and control the type of exercise appropriate for the units in your system. Chapter 12 describes each IOX command in alphabetical order and provides examples that explain how to set parameters for different types of IOX exercises.

1.7.3 The Bad Block Replacement Control Task

The Bad Block Replacement Control Task (RCT) handles bad block replacement and recovery on Mass Storage Control Protocol (MSCP) disks such as the RA80. Bad block handling on MSCP disks consists of four stages: detecting, notifying, replacing, and revectoring. The disk controller (UDA50) detects bad blocks and notifies the driver (DUDRV). The driver activates RCT; then, RCT performs the bad block replacement functions that enable the controller to revector (redirect) I/O from the bad block to the replacement block. Note that all these actions are transparent to you; you do not interact with RCT.

RCT also performs replacement and recovery on MSCP disks that went off line during bad block replacement or before the contents of a write-back cache were copied to the disk.

Note

> Some MSCP devices perform bad block replacement themselves; such devices do not use RCT.

Chapter 13 describes bad block replacement functions and recovery techniques.

1.8 Recovering from a System Failure

When the system fails or "crashes," all activity stops. When you type, nothing prints on your terminal. If you are using a Micro/RSX system, the red lights underneath the diskette drives are not lit, the drives are silent, and the Run light on the front panel of the MicroPDP–11 is not lit.

RSX operating systems have certain built-in safeguards, which protect the data on mass-storage devices and ease the diagnosis of system faults. If the operating system detects a failure within itself, it stops executing or "crashes." The system prints one or more short messages on the console terminal and then halts the processor (which causes the Run light on the front panel of a MicroPDP–11 to go out).

If you are unable to determine the cause of a system crash, restart the system. If the failure was caused by an electrical power failure or by a system user running a faulty privileged program, this solves the problem. However, if the system crashes a second time, a defective device or corrupted system disk is probably at fault. To correct the problem, contact DIGITAL Field Service.

The following sections describe two tools for recovering from a system failure: the Executive Debugging Tool (XDT) and the Crash Dump Analyzer (CDA).

1.8.1 Using the Executive Debugging Tool

You can use the Executive Debugging Tool (XDT) to determine the cause of a system failure. When a software fault forces the system to trap to XDT, XDT has exclusive control of the system, and all other system activity is suspended. You can then use XDT commands and operators to examine registers, memory locations, and system data structures to locate the software fault that caused the trap.

If you have an RSX–11M–PLUS system (excluding RL02 distribution kits) and you want to include support for the memory-resident version of XDT, select XDT during system generation. If you have a pregenerated RSX–11M–PLUS system or a Micro/RSX system, modify the XDT=option statement in LB:[1,2]SYSPARAM.DAT to load the loadable version of XDT each time you restart your system. (For information on modifying SYSPARAM.DAT, see the *RSX-11M-PLUS System Generation and Installation Guide* or the *Micro/RSX System Manager's Guide.*)

If you do not include support for XDT during system generation (RSX-11M-PLUS systems only) or during system startup (RL02 distribution kits or Micro/RSX systems only), you can still use the LOAD command to load XDT interactively, as follows:

```
$
```

However, note that if you plan to use XDT, you must load it into memory before a system failure occurs. By default, XDT is not loaded.

Note

> If you are using a Micro/RSX system, you must have the Micro/RSX Advanced Programmer's Kit installed to use XDT.

For more information on XDT, see the *RSX-11M-PLUS and Micro/RSX XDT Reference Manual.*

1.8.2 Using the Crash Dump Analyzer

The Crash Dump Analyzer (CDA) is a specialized utility that helps you establish the cause of system "crashes." CDA reads the contents of a memory dump created by the crash dump routine of the Executive. Then, it uses the data in the Executive symbol table file (RSX11M.STB) to format the binary input of the memory dump into readable analysis listings.

To get a listing (or "dump") of the contents of memory, CDA requires the crash device driver and a crash device. If you have an RSX-11M-PLUS system, include support for CDA during system generation. If you have a Micro/RSX system or a pregenerated RSX-11M-PLUS system, modify the CRASH_DEVICE=option statement in LB:[1,2]SYSPARAM.DAT to load a crash device each time you restart your system. (For information on modifying SYSPARAM.DAT, see the *RSX-11M-PLUS System Generation and Installation Guide* or the *Micro/RSX System Manager's Guide.*)

Note

> If you are using a Micro/RSX system, you must have the Micro/RSX Advanced Programmer's Kit installed to use CDA.

You can also load the loadable crash dump driver into your system, using the DCL or MCR LOAD command. (See the *RSX-11M-PLUS Command Language Manual*, or the *RSX-11M-PLUS MCR Operations Manual.*) However, note that if you plan to use a crash device, you must load the driver into memory before a system failure occurs. By default, a crash device is not loaded.

CDA is a nonprivileged task that any user can run. However, understanding the "crash dump" requires a knowledge of assembly language programming and the Executive data structures. For more information, see the *RSX-11M-PLUS and Micro/RSX Crash Dump Analyzer Reference Manual.*

1.9 Improving System Performance

RSX-11M-PLUS and Micro/RSX operating systems support two features that increase the efficiency of I/O operations: I/O queue optimization and disk data caching.

1.9.1 I/O Queue Optimization

RSX-11M-PLUS and Micro/RSX operating systems use one of the following methods of I/O queue optimization: Nearest Cylinder, Elevator, or Cylinder Scan. By selecting the method of I/O queue optimization that is best for your system, you can improve the *throughput* of disk subsystems; in other words, you can select the most effective method for processing the list of I/O requests in the queue for a particular disk device.

Chapter 14 describes each of the three methods of optimization and how to choose the one that is best for your system. The chapter also provides a list of error messages returned by the MCR command SET /OPT.

1.9.2 Disk Data Caching

Disk data caching enhances I/O operations by reducing the number of physical I/O requests performed to a disk.

Disk data caching decreases the number of physical I/O operations taking place on "cached" disk drives by using a cache partition in memory. Copies of disk data are placed in the cache partition, making the data available for memory-to-memory transfers instead of disk-to-memory transfers during an I/O request.

Chapter 15 describes all of the disk data caching options. You can disable, override, modify, and monitor disk data caching.

1.10 Customizing the System-User Interface

A command line interpreter (CLI) serves as the interface between you and the operating system. By default, your system is supplied with two CLIs: the DIGITAL Command Language (DCL) and the Monitor Console Routine (MCR). Both MCR and DCL include commands that invoke system tasks and utilities and set or display certain system characteristics. To customize your environment, write your own application CLI, modify your existing CLI (DCL only), or use a catchall task (such as TDX).

1.10.1 Creating Your Own CLI

A CLI is a task that services unsolicited commands meant for the operating system or an application. These commands are not prompted for by any task nor are they given to a task by a read operation on a terminal. A CLI is an efficient means of interfacing an application command processor to any number of terminals, because it does not involve processing characters with an Executive directive Queue I/O Request (QIO$) or an asynchronous system trap (AST).

Chapter 17 introduces the concepts of CLIs on RSX-11M-PLUS and Micro/RSX systems. The chapter includes enough information to allow you to write CLIs that are specific to your application. In addition, it includes a sample CLI (written in both FORTRAN and MACRO-11), which illustrates the CLI interface and clarifies the basic operations of a CLI.

1.10.2 Modifying the DCL Task (RSX-11M-PLUS Systems Only)

The DIGITAL Command Language (DCL) is a general language with a set of commands and required syntax. DCL is used on all RSX operating systems. If you have an RSX-11M-PLUS system (excluding pregenerated systems), you can add, modify, or remove DCL commands.

Chapter 18 explains the ways in which DCL can be installed on your system. It also describes how the DCL task parses and translates commands. In addition, the chapter explains the Macro MetaLanguage, which is a set of MACRO-11 macros used to define DCL.

1.10.3 Using a Catchall Task

RSX-11M-PLUS and Micro/RSX operating systems allow you to use a catchall task that "catches" commands that are not recognized by DCL or MCR. If MCR receives an unrecognized command, it searches for a task installed with the name ...CA. and passes the command line to the task.

Any task installed with the task name ...CA. is treated as a catchall task. Pregenerated RSX-11M-PLUS and Micro/RSX systems install a catchall task called TDX on your system for you. Once installed, TDX checks the typed command against its list of commands. If the commands match, TDX translates the command into a valid MCR command. When you use TDX, you can run uninstalled tasks and abbreviate command names.

Chapter 19 provides more information on installing and using TDX.

1.11 Communicating with Other Systems

A local RSX (RSX-11M/M-PLUS, Micro/RSX, or VAX-11 RSX) terminal can log in to and conduct an interactive session with an external computer system. This session is established by using the Data Terminal Emulator (DTE). The external system can be any type of RSX system (RSX-11M/M-PLUS system, Micro/RSX, and VAX-11 RSX) or a Professional personal computer. Once a local RSX terminal is logged in to an external system, the external system becomes the *host* system. The host system views the local RSX system as *remote*. During an interactive session, files can be transferred between the host system and the remote (local) system. Files are transferred by using the Micro/RSX File Transfer Utility (MFT).

The *RSX-11M-PLUS Utilities Manual* describes how to perform terminal emulation and file operations between an RSX system and an external system.

Chapter 2

The Account File Maintenance Program

The RSX–11M–PLUS and Micro/RSX operating systems provide an Account File Maintenance Program (ACNT) for privileged users to create and maintain an account file. ACNT allows you to create an account file, add and delete accounts, modify accounts, list the accounts, examine an individual account, and sort the account entries.

The account file contains entries for all User Identification Codes (UICs) authorized within the system. One UIC can have several users, but each user must have a unique password. When a user tries to log in, the system compares the HELLO or LOGIN command parameters (the UIC and password) to the account file. By making this comparison, the system determines whether or not the user can have access to the system.

Passwords are automatically encrypted when a new account is created. Once a password is encrypted, the password cannot be seen. Therefore, if you forget your password, you must get a privileged user to enter the ACNT program to assign you a new password.

Note

> Please be aware that no new user-written encryption routines are supported for the creation of new accounts starting with RSX–11M–PLUS Version 3.0. However, encryption routines written prior to Version 3.0 continue to be supported (for login purposes only).

Each account in the account file contains the following information:

- UIC (the account number)
- Default system device
- Password
- First name
- Last name
- Default file protection
- Default directory string
- Session identifier

- Account number
- Command Line Interpreter (CLI), if the system supports more than one CLI
- Whether the user's terminal is to be slaved after login
- Date and time of the user's most recent login
- Number of times that the user has logged in to the system

2.1 Changing Your Password

If you are a nonprivileged user, you cannot use ACNT to modify the account entry for your password. When a nonprivileged user attempts to run the ACNT program, ACNT prints the following message:

```
ACNT -- Cannot be run from a nonprivileged terminal
Use SET PASSWORD to change your password
```

If you have a nonprivileged account and you want to change your password, use the DCL command SET PASSWORD or the MCR command SET /PASSWORD to do so. Please note that if you want to use the SET PASSWORD command, you need to know your old password. If you do not remember your old password, then a privileged user must run ACNT to obtain a new password for you.

For more information on changing your password, see the *RSX-11M-PLUS MCR Operations Manual,* the *RSX-11M-PLUS Command Language Manual,* or the *Micro/RSX User's Guide, Volume 2.*

2.2 Using an Account File from Another System

If you have accounts on another system, you can use the account file from that system. You can use the account file from a previous version of the system or one from another system on which you have installed the current version.

If a previous account file does not contain all the fields that are present in the current release (for example, Version 1.0 of Micro/RSX does not include a default file protection field), ACNT takes the default for that field. The default for the file protection field is the same as the default protection for the volume.

To transfer an existing account file to the current system, use the following procedure:

1. Copy the existing account file to another device (tape or disk). For example, to copy the account file from the system disk (DU0) to another disk (DU2), use the following command line:

   ```
   $
   ```

2. Delete the current account file on the system receiving the new account file, as follows:

   ```
   $
   ```

3. Copy the account file from DU2 to the current system disk, as follows:

   ```
   $
   ```

2.3 Invoking ACNT

Use the MCR or DCL command RUN to invoke ACNT from a privileged terminal, as follows:

```
>.
```

While ACNT is running, the system account file is locked. Tasks, such as HELLO or LOGIN, cannot open the account file to verify account names and passwords. As a result, users are temporarily unable to log in to the system. This does not affect users that are currently logged in to the system; however, it does interfere with users who are attempting to log in. To minimize this interference, run ACNT after users have already logged in and exit ACNT as soon as you are finished.

After you enter the RUN command, ACNT identifies itself and displays the following list of options:

```
      RSX-11M-PLUS Account File Maintenance Program

Account Utility options are:
Add         Add an account to file
Delete      Delete an account file entry
Examine     Examine existing account
List        List account file
Modify      Modify account file
Sort        Sort account file
CTRL/Z      Terminate utility session
Enter option:
```

To select an option, type the first letter of the option name. For example, if you want to add an account, type A. ACNT responds by requesting further input or by displaying information for the option you select.

2.4 Creating the Account File

The CREATE option lets you create the account file. Create the account file immediately after generating your system to allow normal use of the system. The CREATE option allocates the account file LB:[0,0]RSX11.SYS and sets the file protection so that only privileged users and privileged tasks can access the account file.

You can copy and extend the account file by using the Peripheral Interchange Program (PIP; see Section 2.12). The new copy will have the default file protection; therefore, run ACNT immediately. ACNT detects the incorrect protection codes and restores the proper protection.

Example 2–1 shows how to create the account file on RSX–11M–PLUS systems.

Example 2–1: Creating an Account File

```
>
      RSX-11M-PLUS Account File Maintenance Program

Account Utility options are:
Add         Add an account to file
Delete      Delete an account file entry
Examine     Examine existing account
List        List account file
Modify      Modify account file
Sort        Sort account file
CTRL/Z      Terminate utility session
Enter option:
Enter maximum number of accounts:
```

At the Enter option: prompt, specify the maximum number of account entries to be listed in the file. To specify a decimal value, append a period to the number. The system then creates an account file, named LB:[0,0]RSX11.SYS, large enough to contain n_{10} accounts. In the previous example, the system creates an account file for 35_{10} accounts.

After the create operation completes, press the ESCAPE key to request a list of options. You can then add accounts to the account file.

2.5 Adding Account Entries

The ADD option lets you add accounts to the account file. ACNT prompts you for the following information:

- UIC (the account number)
- Default system device
- Password
- First name
- Last name
- Default file protection
- Default directory string
- Session identifier
- Account number
- CLI, if the system supports more than one CLI
- Whether the user's terminal is to be slaved after login

ACNT maintains the file in ascending order by UIC.

The prompts for the password, first name, and last name specify the maximum number of characters that ACNT accepts. Passwords can be up to 39 characters long; however, systems using the DECnet package can use only 8-character passwords. The valid characters for user names and passwords are as follows:

- A to Z, uppercase and lowercase
- 0 to 9
- ' (apostrophe)
- . (period)
- - (hyphen)
- $ (dollar sign)
- ! (exclamation point)

The default system device prompt requests a device name and unit number (optional, if the unit is 0). ACNT accepts logical, physical, and pseudo device names. The acceptance of logical and pseudo device names lets you move the user disk to another drive without having to change the default device in the account file. When you specify the device name, do not include a colon (:).

RSX–11M–PLUS systems include two additional prompts: one is for a session identifier and the other is for the user account number. The session identifier and user account number are used by the Resource Accounting program (see Chapter 10). The session identifier is also used by the HELLO task (see the *RSX–11M–PLUS MCR Operations Manual* or the *RSX–11M–PLUS Command Language Manual*).

Example 2–2 shows how to add an account.

Example 2–2: Adding an Account Entry

```
>                                               ❶
      RSX-11M-PLUS Account File Maintenance Program

Account Utility options are:                               ❷
Add         Add an account to file
Delete      Delete an account file entry
Examine     Examine existing account
List        List account file
Modify      Modify account file
Sort        Sort account file
CTRL/Z      Terminate utility session
Enter option:                                   ❸
Enter account (group, member):                  ❹
Default system device ( DDU ):                  ❺
Password (39 chars. or fewer):                  ❻
First name (12 chars. or fewer):
Last name (14 chars. or fewer):                 ❼
Default file protection ([SY,OW,GR,WO]):
[RWED,RWED,R,R]                      ❽
Default directory string ([group, member] or [name]):                 ❾
Session ident (3 chars. or fewer):              ❿
Account number (4 digits or fewer):             ⓫
Enter user CLI (default=MCR):                   ⓬
Slave terminal? [Y/N]:                          ⓭
Disable login/logout messages? [Y/N]:           ⓮
```

In this example, ACNT creates an account for Kathy Bean.

❶ Invokes the ACNT utility by using the RUN command.

❷ Lists the available options.

❸ Selects the ADD option to add an account entry.

❹ Specifies the UIC with a group number of 10 (a nonprivileged account) and a member number of 11.

❺ Specifies the default system device. ACNT accepts logical, physical, and pseudo device names. Specifying the pseudo device name SY0 allows you to move the system disk to another drive without having to change the default device in the account. Note that the device name does not include a colon.

❻ Specifies the password as BASEKITPL, which is encrypted by ACNT when the account is created.

❼ Creates the account for a user named Kathy Bean. Because the ACNT encryption routines require the user's last name to form the encrypted password, you must specify a last name to create a valid account entry. If you do not specify a last name, the account will not be created with multiuser protection.

❽ Specifies the default file protection as read, write, extend, and delete access for system and owner, and read-only access for group and world.

❾ Selects the user's last name (Bean) as the default directory string, which can be up to 9 characters long. Your response to this prompt is always interpreted as a named directory. To specify the default directory string as a numbered (nonamed) directory, press the RETURN key without specifying the UIC. By default, ACNT assigns the UIC to the default directory (in this example, [10,11]).

❿ Specifies WRT as the session identifier.

⓫ Specifies 23 as the account number.

⓬ Selects MCR as the user's CLI. MCR is the default for RSX-11M-PLUS systems; DCL is the default for Micro/RSX systems.

⓭ Sets the terminal to noslave status.

⓮ Enables login and logout messages.

After ACNT creates the account and finishes prompting for all the necessary information, it returns the following message:

```
Account entry and directory have been created.
```

2.6 Deleting Account Entries

The DELETE option eliminates individual accounts from the file. After you specify the account to be deleted, ACNT displays the account entry and requests a Y (Yes) or N (No) response for verification. If you type N, ACNT searches for another account with the same UIC. If you type Y, ACNT asks if you want to delete the account's User File Directory (UFD) and other files; type Y again to delete these files. If you type N in response to the deletion query, the account's files and UFD are left intact.

Before deleting the account's UFD, be sure that the files associated with the UFD are not protected to prevent the system from deleting them. If the user has set the protection to deny the system delete access, ACNT cannot delete those files; however, ACNT can still delete the UFD. So, if you type Y in response to the Delete UFD and files? prompt, and ACNT is unable to delete the files, ACNT will delete the UFD but not the files. If this happens, use the File Structure Verification Utility (VFY) to eliminate the files (see the *RSX-11M-PLUS Utilities Manual*).

Example 2-3 shows how to delete accounts from the account file.

Example 2-3: Deleting an Account Entry

```
>
      RSX-11M-PLUS Account File Maintenance Program

Account Utility options are:
Add         Add an account to file
Delete      Delete an account file entry
Examine     Examine existing account
List        List account file
Modify      Modify account file
Sort        Sort account file
CTRL/Z      Terminate utility session
Enter option:                                        ❶
Enter account (group, member):                       ❷

7-AUG-87 14:30:02 RSX-11M-PLUS Multiuser Account File List  Page 1

Owner = [010,011]           Login_defaults =  SY0:[BEAN]
L_name = BEAN               F_name = KATHY           Password = (ENCRYPTED)
Def_CLI = MCR               Session ID = WRT         Account # = 23
Total_logins = 24           Last_login = 7-Aug-87  13:01:50
Characteristics = NOSlave         Def_dir_string     NOSilent
Def_Protection = [RWED,RWED,R,R]

This entry ? [[Y/N]]:                                ❸
Delete UFD and files? [Y/N]:                         ❹
Account entry has been deleted                       ❺
```

This example deletes Kathy Bean's account, [10,11].

❶ Selects the DELETE option.

❷ Specifies the account [10,11] for deletion.

❸ Displays the account entry and prompts you for verification.

❹ Prompts you before deleting the account UFD and files. If you are sure the system has delete access to all the files associated with [10,11], type Y (Yes) in response to this question. Otherwise, exit ACNT and use the MCR or DCL command SET to change the file protection to SYSTEM:DELETE. Then, reinvoke ACNT and delete the account.

❺ Displays a message to indicate that the account has been deleted.

2.7 Examining Account Entries

To display the information for a single account, use the EXAMINE option. To display the information for a group of accounts or for all accounts in the account file, use the LIST option (see Section 2.8).

You can specify the account you want to examine by entering either the UIC in the format [g,m] or an account name. ACNT displays all accounts it finds that match the UIC or account name specified.

The EXAMINE option displays the following information for each account in the account file:

- UIC
- Last name
- User's CLI
- Number of times the user has logged in to the system.
- Whether the user's terminal is to be slaved after login.
- Default file protection.
- Default system device
- First name
- Session identifer and account number
- Date and time of the last login.
- Default directory string. If the default directory string is the same as the login default, ACNT displays a blank space.
- Status of the user's terminal as silent (or NOsilent) after login.

ACNT remains in Examine mode until you press the ESCAPE key to request a list of options.

Example 2–4 shows how to use ACNT to examine the accounts in the account file.

Example 2-4: Examining an Account Entry

```
>
      RSX-11M-PLUS Account File Maintenance Program

Account Utility options are:
Add         Add an account to file
Delete      Delete an account file entry
Examine     Examine existing account
List        List account file
Modify      Modify account file
Sort        Sort account file
CTRL/Z      Terminate utility session
Enter option:
Enter account (group, member):

7-AUG-87 14:30:02 RSX-11M-PLUS Multiuser Account File List  Page 1

Owner = [010,011]           Login_defaults =  SY0:[BEAN]  ❶
L_name = BEAN               F_name = KATHY            Password=(ENCRYPTED) ❷
Def_CLI = MCR               Session ID = WRT          Account # = 23 ❸
Total_logins = 24           Last_login = 7-AUG-87  13:01:50 ❹
Characteristics = NOSlave   Def_dir_string            NOSilent ❺
Def_Protection = [RWED,RWED,R,R] ❻
```

In this example, Kathy Bean's account is examined.

❶ Lists the UIC [10,11] and login defaults. The default system device is SY0 and the default directory string is [BEAN].

❷ Displays Kathy's last name and first name. The password is encrypted, so ACNT displays (ENCRYPTED) instead of the password itself.

❸ Displays the CLI (MCR), the session identifier (WRT), and the account number (23).

❹ Displays the total number of logins (24), as well as the date (7-AUG-87) and time (13:01:50) of the last login.

❺ Displays the terminal characteristics (the terminal is set to NOSlave and NOSilent). ACNT leaves a space in place of the default directory because it is the same as the login default displayed in the first line.

❻ Shows the default file protection. In this example, the file protection is read, write, extend, and delete access for system and owner, and read access for group and world. If you examine an account that does not use a default file protection, ACNT displays NODef_Protection.

After you examine an account or group of accounts, ACNT prompts you with the following message:

```
Key <RETURN> to continue, CTRL/Z for ACNT options menu:
```

If you want to examine another entry, press the RETURN key. If you want to select a different option or to exit ACNT, press CTRL/Z.

2.8 Listing Account Entries

The LIST option either displays on your terminal or lists in a file the account entries in the account file. You can display all the entries or only a specified group of entries. If you type F (for file), ACNT creates the file ACCOUNT.DMP, which contains all the account information you request with the LIST option. After you exit from ACNT, you can print the ACCOUNT.DMP file.

Example 2–5 shows how to list account entries.

Example 2–5: Listing Account Entries

```
>
      RSX-11M-PLUS Account File Maintenance Program

Account Utility options are:
Add         Add an account to file
Delete      Delete an account file entry
Examine     Examine existing account
List        List account file
Modify      Modify account file
Sort        Sort account file
CTRL/Z      Terminate utility session
Enter option:        ❶
  Only the old UNENCRYPTED type passwords can be printed
Print passwords (default no)? [Y/N]:      ❷
All accounts (default no)? [Y/N]:      ❸
Enter group number:      ❹
List on Terminal(default) or File(ACCOUNT.DMP) [T/F]:      ❺

7-AUG-87 14:30:02 RSX-11M-PLUS Multiuser Account File List  Page 1

Owner = [010,011]           Login_defaults =  SYO:[BEAN]
L_name = BEAN               F_name = KATHY          Password =
Def_CLI = MCR               Session ID = WRT        Account # = 23
Total_logins = 24           Last_login = 7-AUG-87  13:01:50
Characteristics = NOSlave   Def_dir_string          NOSilent
Def_Protection = [RWED,RWED,R,R]
```

This example lists only one account with the group number equal to 10. The member number is 11, and the owner is Kathy Bean.

❶ Selects the LIST option by typing the letter L.

❷ Shows that the password is not printed; the password field for the displayed entry is left blank. (Note that encrypted passwords cannot be displayed, even if you specify Y in response to the "Print passwords?" prompt.) This is the default.

❸ Lists only a selected group of accounts. This is the default.

❹ Specifies all accounts with the group number field of the UIC equal to 10.

❺ Displays the account on the terminal (and not in the file, ACCOUNT.DMP). This is the default.

After ACNT lists all the accounts you have specified, it returns the following prompt:

```
Key <RETURN> to continue, CTRL/Z for ACNT options menu:
```

To list more account entries, press the RETURN key. To exit ACNT or to select another option, press CTRL/Z.

2.9 Modifying Account Entries

The MODIFY option allows you to change the password, default system device, default file protection, first name and last name for an account, and the slave status of a terminal. You can also change the session identifier and the account number. If you include support for multiple CLIs during system generation, you can change the default CLI for an account.

After you specify the account you want to modify, ACNT displays an account entry and asks for verification. To indicate that the correct account was or was not specified, type either Y (Yes) or N (No), respectively. If you type N, ACNT searches the file for another entry with the same UIC. If you type Y, ACNT prompts for the entry details. If you do not want to change an item, press the ESCAPE key.

Example 2–6 shows how to modify account information.

Example 2-6: Modifying an Account Entry

```
>
      RSX-11M-PLUS Account File Maintenance Program

Account Utility options are:
Add         Add an account to file
Delete      Delete an account file entry
Examine     Examine existing account
List        List account file
Modify      Modify account file
Sort        Sort account file
CTRL/Z      Terminate utility session
Enter option:                                  ❶
Enter account (group, member):                 ❷

7-AUG-87 14:30:02 RSX-11M-PLUS Multiuser Account File List  Page 1

Owner = [010,011]           Login_defaults =  SY0:[BEAN]
L_name = BEAN               F_name = KATHY          Password = (ENCRYPTED)
Def_CLI = MCR               Session ID = WRT        Account # = 23
Total_logins = 24           Last_login = 7-AUG-87  13:01:50
Characteristics = NOSlave         Def_dir_string    NOSilent
Def_Protection = [RWED,RWED,R,R]

This entry ? [Y/N]:                          ❸

Type <ESC> to leave entry unchanged
Password (39 chars. or fewer):
Default system device ( DDU ):                 ❹
First name (12 chars. or fewer ):
Last name (14 chars. or fewer ):
Default file protection ([SY,OW,GR,WO]):
Default directory string ([group, member] or [name]):               ❺

New directory has been created                              ❻

Enter user CLI (default=MCR):
Slave terminal? [Y/N]:
Disable login/logout messages? [Y/N]:
Session ident (3 chars. or fewer):                    ❼
Account number (4 digits or fewer):
```

In this example, Kathy Bean's account is modified.

❶ Specifies the MODIFY option.

❷ Selects the account UIC [10,11]. Note that you can also specify the user's last name (in this example, Bean).

❸ Displays the requested entry and prompts you for verification.

❹ Leaves the password, system device, first and last name, and default file protection unchanged.

❺ Changes the default directory string to MEMOS.

❻ Displays a message to indicate that the default directory string has been changed.

❼ Changes the session identifier to DEV.

ACNT prompts you for changes to all the account fields; then, ACNT prompts you with the following message:

```
Key <RETURN> to continue, CTRL/Z for ACNT options menu:
```

Press CTRL/Z to return to the options menu. Then, if you want to exit ACNT, press CTRL/Z again.

2.10 Sorting Account Entries

The SORT option sorts the account entries in ascending order by UIC. Example 2–7 shows how to sort the account file.

Example 2–7: Sorting the Account File

```
>
      RSX-11M-PLUS Account File Maintenance Program

Account Utility options are:
Add         Add an account to file
Delete      Delete an account file entry
Examine     Examine existing account
List        List account file
Modify      Modify account file
Sort        Sort account file
CTRL/Z      Terminate utility session
Enter option:
```

After the sort operation is complete, ACNT returns the following message and then redisplays the ACNT menu:

```
Account file is now sorted by UIC
```

2.11 Exiting from ACNT

The EXIT option allows you to exit from the ACNT program. When you exit from ACNT, ACNT returns the following message:

```
Account file modification is complete
```

Note that this message is displayed even if you do not modify any of the account entries.

2.12 Increasing the Size of the Account File

The account file supplied with your system ([0,0]RSX11.SYS) can include a maximum of 36 accounts. If you need more accounts, create a larger account file. Then, run ACNT to reenter the accounts that were in the previous file and to add new accounts.

To create a larger account file, use the following procedure:

1. Run the ACNT utility and use the LIST option to obtain a list of the accounts in the current account file. Send the output to a file so that you can print it later.
2. Delete the current account file by using the following command line:

   ```
   $
   ```

3. Run the ACNT utility again. ACNT prints an identification line and the following messages:

   ```
       ***  RSX-11M-PLUS Account File Maintenance Utility  ***

   Creating Account file -- LB:[0,0]RSX11.SYS
   Enter maximum number of accounts:
   ```

4. Enter the maximum number of accounts that you expect to need. For example:

   ```
   Enter maximum number of accounts:
   ```

   To allow for future expansion, create a few more accounts than you currently need.

5. Use the ADD option to reenter the accounts that were in the previous file and to add new accounts.

   Because the passwords for the previous accounts are encrypted, you need to assign a temporary password for each account. Notify users of the temporary password and then advise them to alter their passwords (with the SET PASSWORD command) as soon as they log in to the system.

2.13 Error Messages

This section describes the messages that the ACNT utility displays on your terminal when it encounters errors.

ACNT—Account already exists

Explanation: You attempted to add an account with a UIC and password that already exists in the account file.

User Action: Use the LIST option to obtain a listing of all account entries in the account file. Then, use a UIC and password that do not appear on the list.

ACNT—Account file already exists

Explanation: You attempted to create an account file that already exists.

User Action: You cannot create a second account file. However, if you want to add an account entry to the account file, select the ADD option by typing A.

ACNT—Account file error

Explanation: The program detected an I/O error while processing the account file. This may indicate that the disk is too full or that another user was attempting to access the same file.

User Action: Use the DIRECTORY/FULL command to see if the disk is full. Also, be sure that no other (privileged) users are running ACNT. If the disk is full, purge or delete unnecessary files to create more space. If another user is running ACNT, wait until they are finished before reentering the RUN $ACNT command line.

ACNT—Account file full

Explanation: The account file is full; you cannot add further accounts.

User Action: Create a larger account file. (For information on increasing the size of the account file, see Section 2.12.)

ACNT—Cannot find account

Explanation: You attempted to delete, to examine, or to modify an account that does not exist.

User Action: Use the LIST option to check the account name and UIC. Then, try again specifying the correct account.

ACNT—Invalid default device name

Explanation: The default system device you specified is not a disk device.

User Action: Specify a disk as the default system device.

ACNT—Invalid option

Explanation: The option you specified in response to the Enter option: prompt is not a valid option. You may have typed the name of the option incorrectly.

User Action: Specify one of the options listed in the menu (type A, D, E, L, M, S, or press CTRL/Z).

ACNT—I/O Error

Explanation: The disk that has the account file is full or write-locked.

User Action: If the disk is full, delete files to provide space on the disk. If it is write-locked, enable write access.

ACNT—Listing file error

Explanation: When you use the LIST option, ACNT attempts to open a file named ACCOUNT.DMP. By default, ACNT searches for the file in your current UIC on the device SY:. This message indicates that ACNT cannot open the file; for example, the device is full or it does not contain a UFD for the current UIC. The device may be full, write-locked, or it may not contain your current default directory.

User Action: If the device is full, delete some files. If the device is write-locked, enable write access. If your current default directory does not exist on the current default device, create the directory, change your default directory to one that does exist, or change your default device to one that contains your default directory.

ACNT—Syntax or command input error

Explanation: You typed an invalid character.

User Action: Check the command syntax and reenter the information.

ACNT—Warning -- Device does not exist on current system. OK? [Y/N]:

Explanation: The specified default device is not known to the system.

User Action: If you enter Y (Yes) in response to the OK? prompt, ACNT retains the device name as a logical device name. If you create accounts by using logical device names for default devices, be sure to assign those logical names to actual devices before the accounts are used. If you enter N (No), ACNT prompts again for another device name.

ACNT—Workfile-dyn. mem. exhausted

Explanation: The system does not have sufficient pool space for the requested operation. This is usually a temporary condition.

User Action: Wait for pool to be restored; then, reenter the command. (For more information on monitoring and restoring pool, see Chapters 7 and 8, respectively.)

ACNT—Workfile-open file

Explanation: The disk containing the account file is full or write-locked.

User Action: If the disk is full, delete files to provide space on the disk. If the disk is write-locked, set default to a privileged UIC to enable write access.

ACNT—Workfile-virtual storage exceeded

Explanation: The workfile exceeds virtual memory or there is insufficient pool space. This is usually a temporary condition.

User Action: Wait for pool to be restored; then, reenter the command. (For more information on monitoring and restoring pool, see Chapters 7 and 8, respectively.)

Chapter 3

Setting Up and Running the Queue Manager

This chapter includes an illustrated sequence of steps for setting up the Queue Manager (QMG), print processors, and batch processors. It also includes a description of commands to QMG and a list of relevant error messages. This chapter does not include descriptions of the commands for print and batch processing operations. For information on using QMG to process print or batch requests, see the *RSX-11M-PLUS Batch and Queue Operations Manual* or the *Micro/RSX User's Guide, Volume 2.*

Except where noted, all commands in this chapter are privileged; you must have a privileged account to use the commands.

3.1 Introduction to the Queue Manager

Most systems include the Queue Manager (QMG) for multistream line printer spooling. *Line printer spooling* is the process of coordinating output directed to a line printer. Without QMG, output passes directly to the line printer. So, if many users and tasks direct output to the line printer at the same time, the printer may print three lines of one file and then two of another. However, when you use QMG, output is saved in separate files on a mass-storage device (usually a disk). Then, QMG *despools* or transfers the files from the disk to the printer, one at a time.

The Queue Manager also supports *transparent spooling* and *batch processing*. These features are defined as follows:

Transparent spooling — Allows you to specify the name of a spooled output device as the destination of a job (instead of using an output file specification). Transparent spooling applies to any task that creates an output file.

Batch processing — Allows you to submit a command file containing commands and data to the system for execution. Note that you can use QMG without batch processing, but you cannot have batch processing without QMG.

For more information on spooling and batch processing, see the *RSX-11M-PLUS Batch and Queue Operations Manual* or the *Micro/RSX User's Guide, Volume 2.*

3.2 Queue Management Components

The queue management subsystem consists of five major components. They are as follows:

A command interface task (QMGCLI)

Processes print requests, batch requests, and queue access commands. (For more information on the queue access commands, see the *RSX-11M-PLUS Batch and Queue Operations Manual* or the *Micro/RSX User's Guide, Volume 1*.) QMGCLI communicates with the Queue Manager (QMG) by sending it data packets.

The Queue Manager (QMG)

Controls the queuing and dequeuing of print and batch jobs. QMG communicates with QMGCLI and the despooler tasks to process requests.

The despool prototype task (LPP)

Attaches a device and despools print jobs to the device. It is an output despooler, which is also called a print processor. You must install a print processor for each device you want to be spooled (that is, for each device you place under the control of QMG).

With the exception of RL02 distribution kits, RSX-11M-PLUS systems include the source files for LPP on device SY: in directory [121,10]. You can use this as a model to write print processors (output despoolers) for your applications.

The batch processor prototype task (BPR)

Creates a virtual terminal that passes commands to the batch processor and to a system CLI (MCR, DCL, or a user-written CLI.) The BPR task can be installed as many as 16 times.

The Card Reader Processor (CRP)

Reads a deck of cards containing a batch job, copies the cards to a file, and requests QMG to create a batch job containing this file.

For information on using CRP, see the *RSX-11M-PLUS Batch and Queue Operations Manual.*

3.3 Setting Up the Queue Manager

You can set up the Queue Manager (QMG) with indirect command files or interactively.

There are two different methods for using indirect command files to set up QMG. The method you select depends upon the type of RSX operating system you are using, as follows:

- If you have a Micro/RSX or pregenerated RSX-11M-PLUS operating system, modify the system configuration data file LB:[1,2]SYSPARAM.DAT, which is used by the startup command file LB:[1,2]STARTUP.CMD. (For information on SYSPARAM.DAT, see the *RSX-11M-PLUS System Generation and Installation Guide* or the *Micro/RSX System Manager's Guide*.)
- If you have an RSX-11M-PLUS system (excluding RL02 distribution kits), use the prototype command file LB:[1,2]QMGSTART.CMD supplied with your system. This file includes commands for setting up the Queue Manager in a standard configuration. It sets up one print processor and one batch processor. By modifying the command file, you can tailor the Queue Manager to your installation.

If you use QMGSTART.CMD, invoke it during the system startup procedure by including the following command line in LB:[1,2]STARTUP.CMD:

```
@LB:[1,2]QMGSTART.CMD
```

For more information on altering system startup procedures, see the *RSX-11M-PLUS System Generation and Installation Guide* or the *Micro/RSX System Manager's Guide*.

The sections that follow explain how to bring QMG into your system interactively. Both MCR and DCL commands are included. Each of the steps includes a block diagram of the Queue Manager, processors, and devices as they exist at that point in the procedure. The parts of the subsystem added at each step are shown in red.

For a description of the individual commands and qualifiers, see Section 3.5.

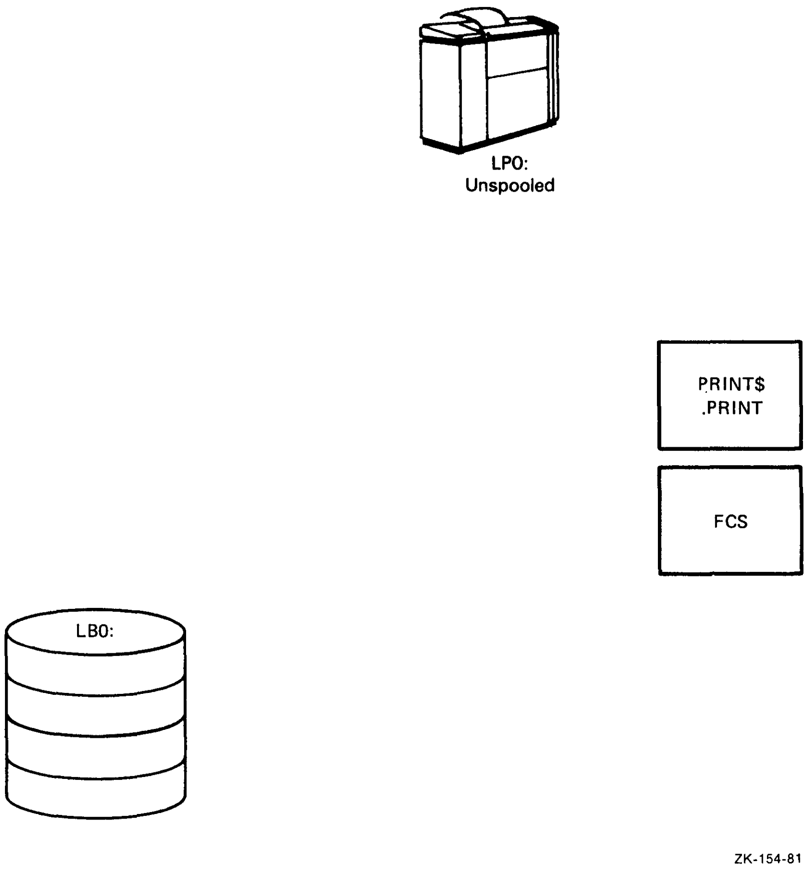
To bring QMG into your system, perform the following steps:

1. Install the Queue Manager as a checkpointable task, using the DCL command INSTALL/CHECKPOINT or the MCR command INSTALL /CKP=YES.

   ```
   DCL>
   ```

   ```
   MCR>
   ```

   The QMG task is located with the privileged utility tasks.



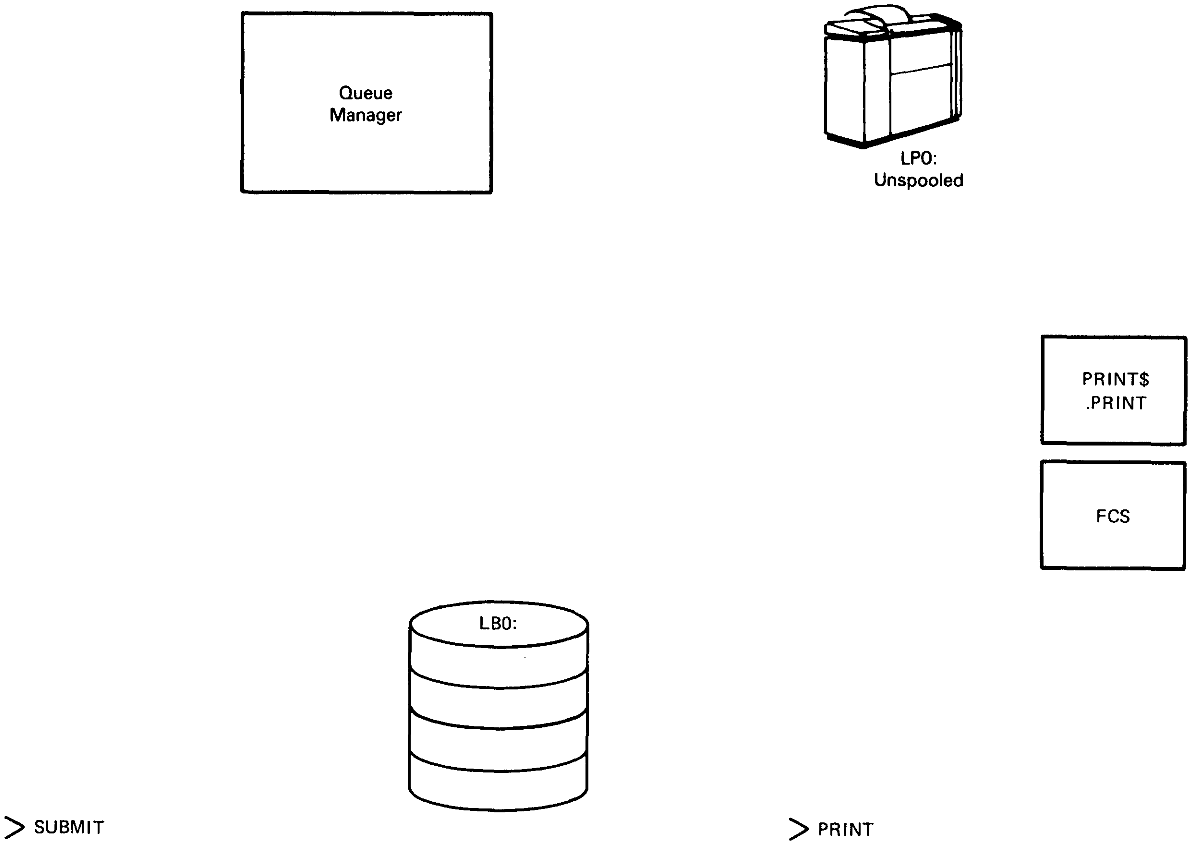
2. Start QMG by using the START/QUEUE/MANAGER (QUE /STA:QMG) command, which also initializes the default queues PRINT and BATCH.

   ```
   DCL>
   ```

   ```
   MCR>
   ```

   If the queue file [1,7]QUEUE.SYS does not exist, it is created on the spooling device SP: (usually assigned to the system library device, LB:) and all previous queue assignments are cleared.

   If the queue file already exists, all queue assignments are maintained but they are not assigned to processors. Therefore, you do not need to reinitialize queues (step 3), but you must reinitialize the processors (step 5).



ZK-155-81

3. Initialize a *device-specific queue* to serve the system tasks sending output to the device LP0: through QMG.

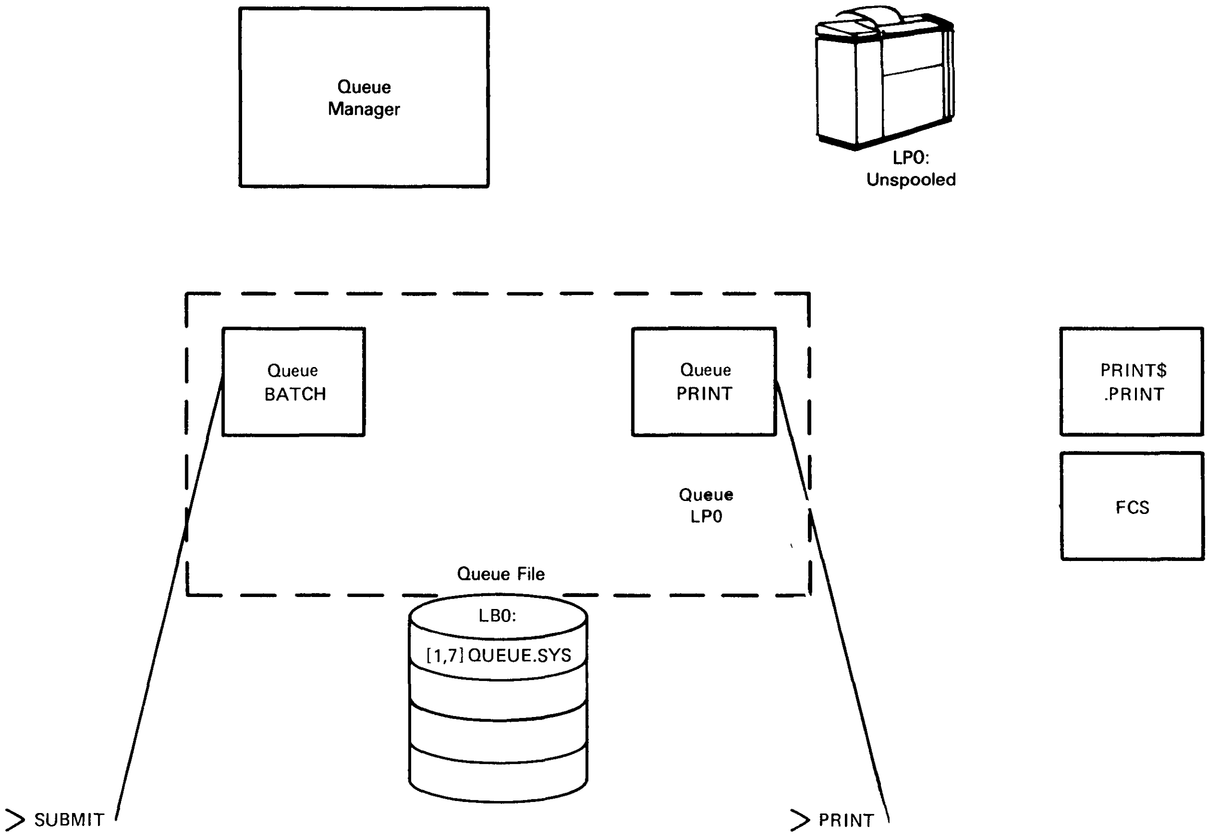
   ```
   DCL>
   ```

   ```
   MCR>
   ```

   Initialization names and creates a queue. The default print queue (PRINT) is initialized in the previous step, but you must initialize at least one more queue (even if your system has only one printing device). There must be a device-specific queue for every spooled output device. (PRINT is a general queue.) In addition, you must initialize the device-specific queue before you initialize the associated device.

   By convention, a device-specific queue is named after the device it supports. In this example, the device-specific queue LP0 supports the device LP0:.

   You can initialize as many as 14 additional print queues for a total of 16. One must be the default queue PRINT. The remaining 15 can be device-specific or general queues. Queues for application processors are counted as print queues.



ZK-156-81

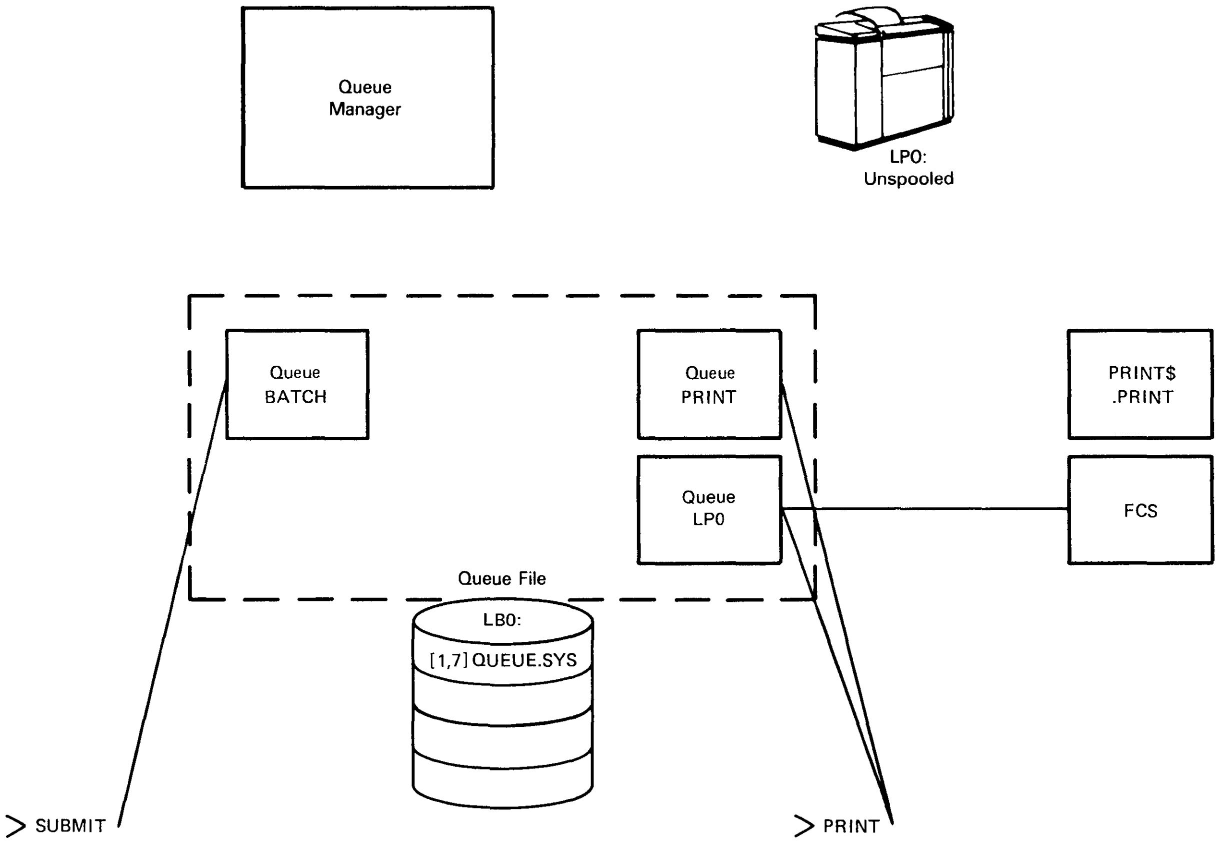
4. Install a print processor for each physical printing device that you are planning to initialize.

   ```
   DCL>
   ```

   ```
   MCR>
   ```

   Use the printer name to derive a name for the corresponding processor. For example, LP0: is the printer that corresponds to the processor LP0. The despooler task LPP.TSK installs automatically as LP0.

   The print processors are located with the privileged utility tasks.



ZK-157-81

If you have additional output devices, use the /TASK qualifier to install and to name print processors for them, as follows:

```
DCL>
```

```
MCR>
```

You can install as many as 15 print processors. Applications processors are considered print processors for this count.

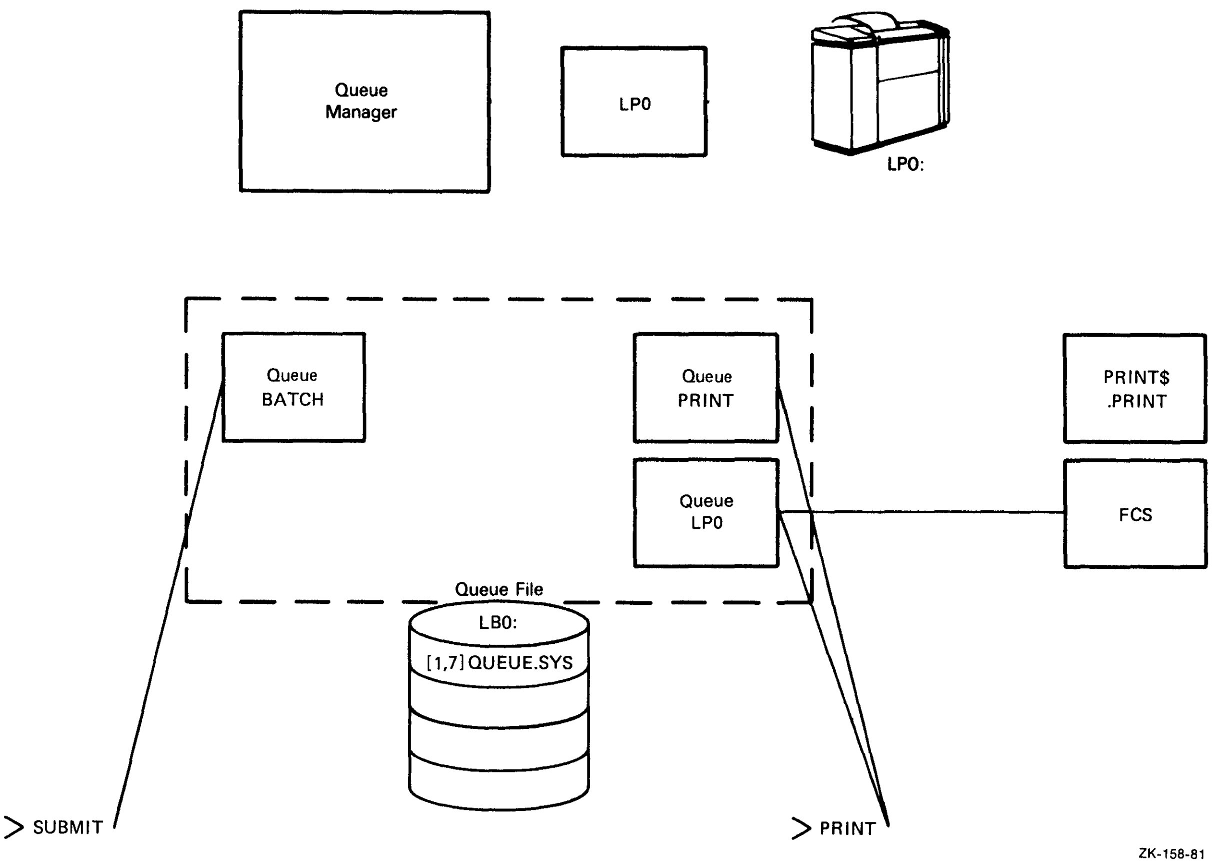
5. Initialize and set the desired attributes for the print processors.

```
DCL>
```

```
MCR>
```

The INITIALIZE command assigns the queue LP0 to the print processor LP0. Before you can initialize the print processor LP0, the queue LP0 must exist (see steps 2 and 3).

Initialization of a print processor sets the associated output device spooled.



If you plan to use a hardcopy terminal as both a terminal and a spooled device, use the DCL qualifier /SHAREABLE (MCR switch /SHR) to establish output to that terminal as shareable. For example:

```
DCL>

MCR>
```

When a terminal is *shareable,* it is owned by the print or batch processor only while it is printing a job. When the job completes, the processor detaches the terminal and you can use the terminal to log in to the system. The Queue Manager (QMG) does not spool jobs to the terminal if it is attached by another task. However, if you do not initialize the print processor as shareable, QMG attaches the terminal and makes it unavailable to any other task.

Note

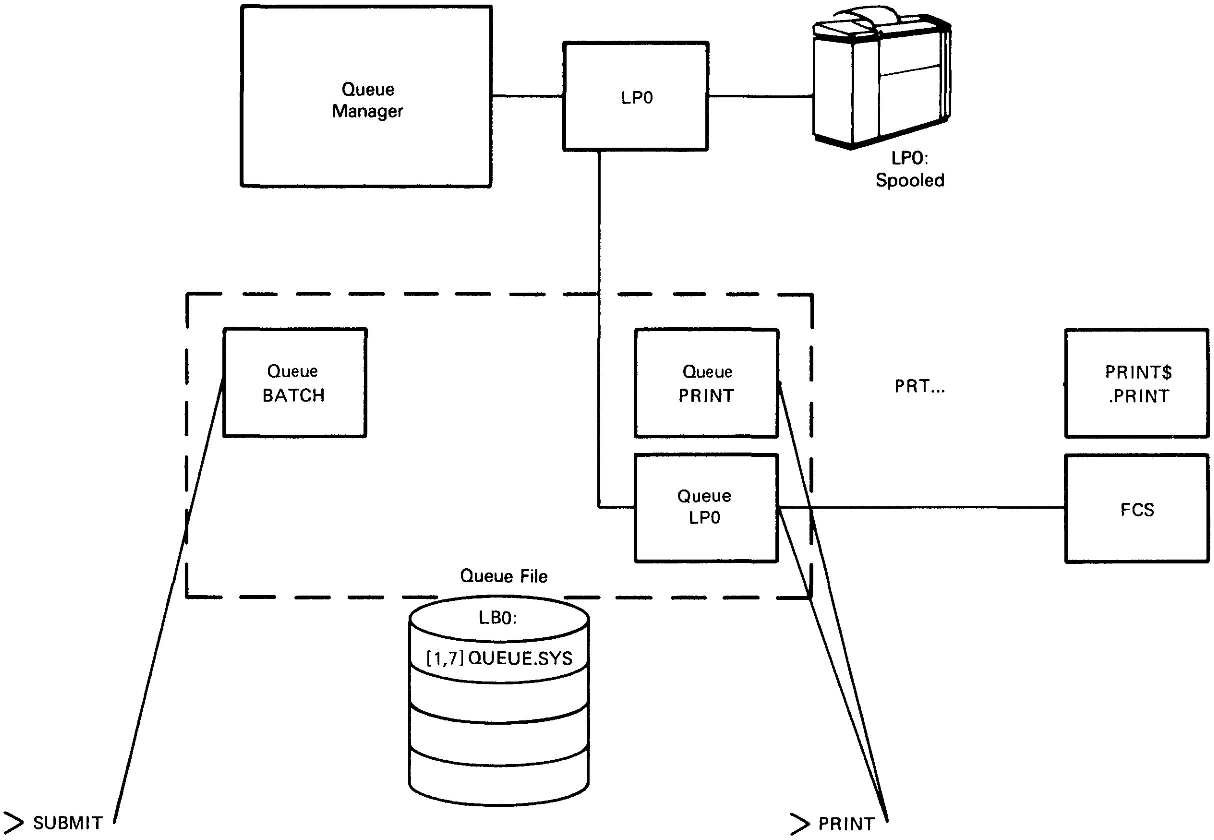
When you initialize a processor as shareable, transparent spooling to the associated device cannot occur.

6. Install the interface between QMG and other system tasks.

   ```
   DCL>
   MCR>
   ```

   The QMG interface is controlled by the task $QMGPRT.TSK. Install the task with the task name *PRT....* The task $QMGPRT is located with the nonprivileged utility tasks.

   For more information, see Section 3.4.

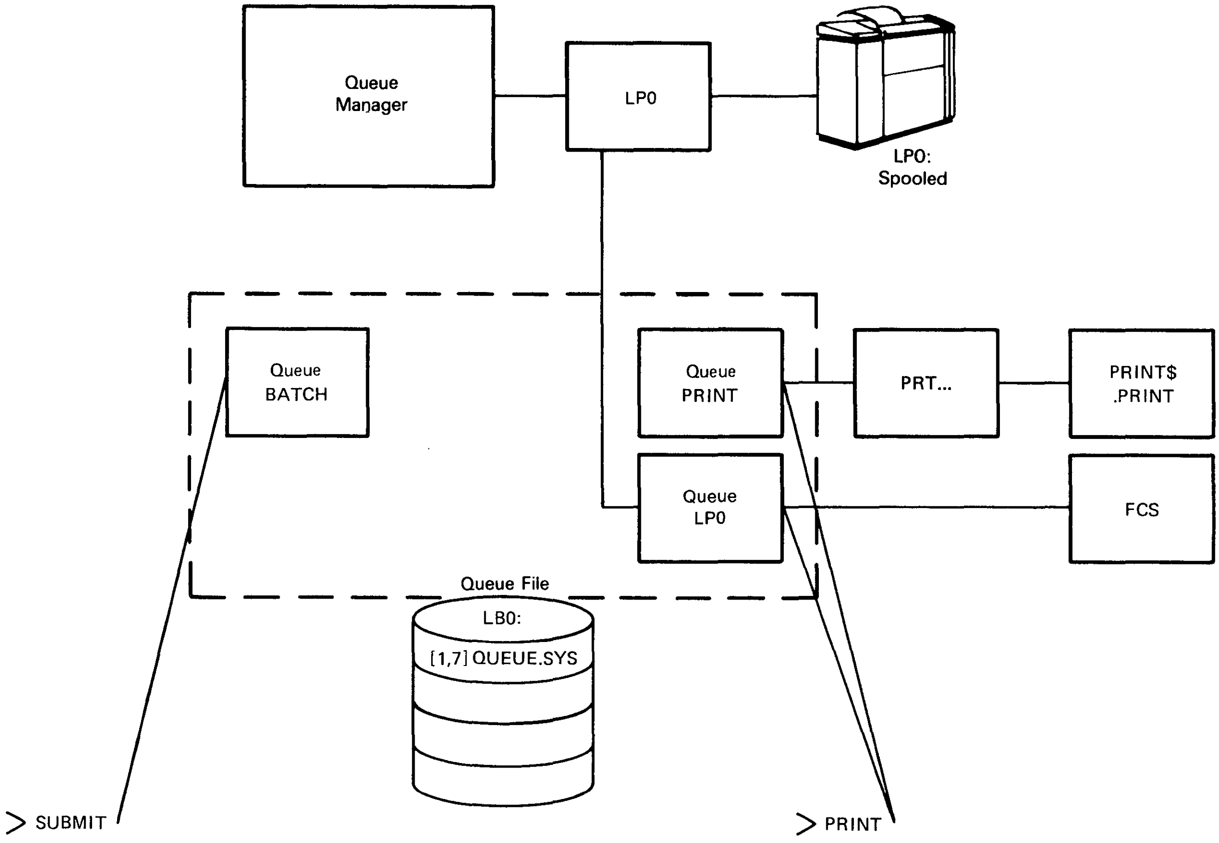


ZK-159-81

7. Assign the queues to the print processors to establish a connection between the two.

   ```
   DCL>
   MCR>
   ```

   QMG can send jobs to assigned processors only. If you initialize a queue without assigning it, jobs sent to the queue are not processed. You can assign each queue to more than one processor, and you can assign more than one queue to each processor.



ZK-160-81

This step is required only if you want to establish general queues. The three previous steps initialize a print processor and establish a device-specific print queue; your new printer is ready to be used. However, when you specify a device-specific queue (for example, LP0) in your PRINT command line, you do not have the option of sending your print job to any other processor (or corresponding printer). If LP0 is not available, your job must wait.

General queues can send print jobs to any available processor to which they have been assigned. Therefore, if you have more than one printer and you want print jobs to go to the first available printer (see Note), establish a connection between a general print queue (such as PRINT) and the new print processor.

Note

Different types of print jobs are established by the PRINT command that places them in a queue. Therefore, print jobs do not actually go to the "first available printer"; more specifically, they are sent to the *first available printer that has the same characteristics as those specified in the PRINT command line.*

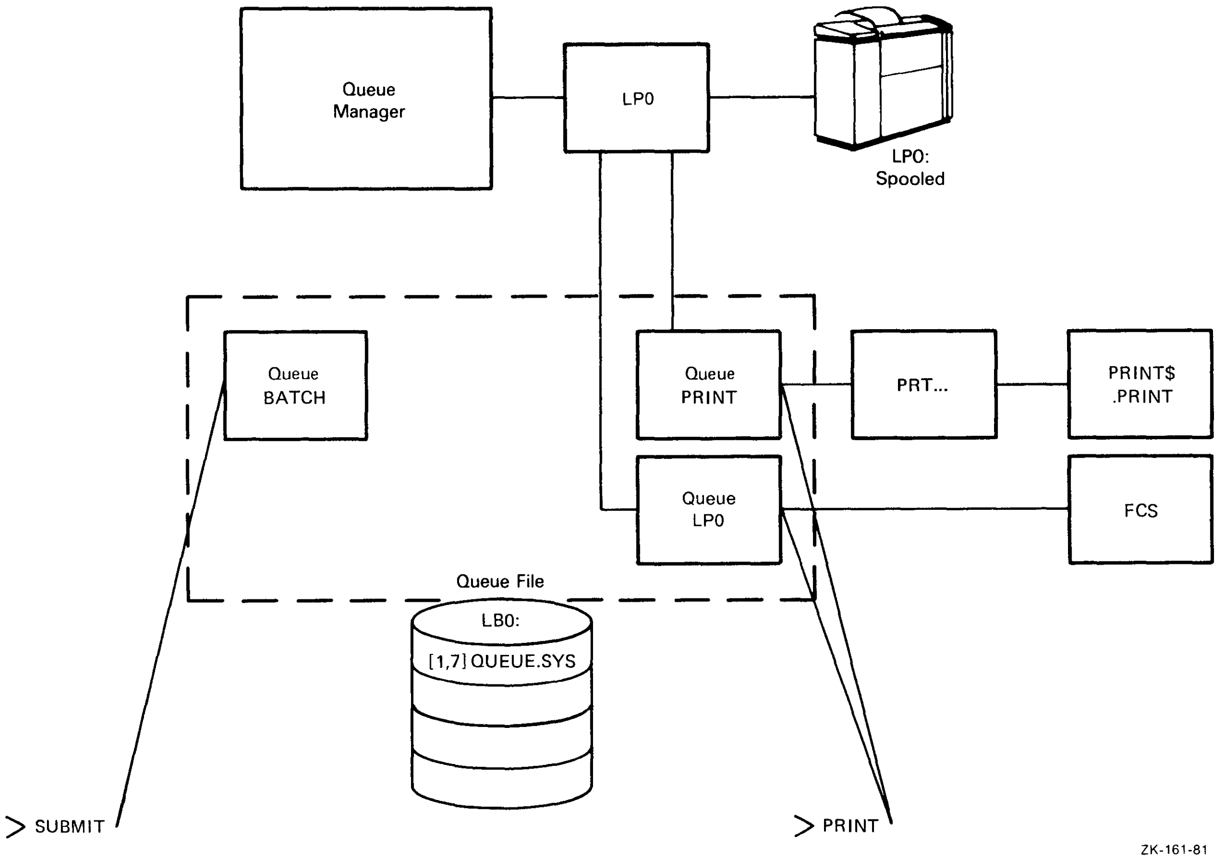
8. Install the batch processor (BPR).

```
DCL>

MCR>
```

The BPR task is located with the privileged utility tasks and installs automatically with the taskname BAP0. If you have additional batch processors, use the /TASK qualifier to install and name them, as follows:

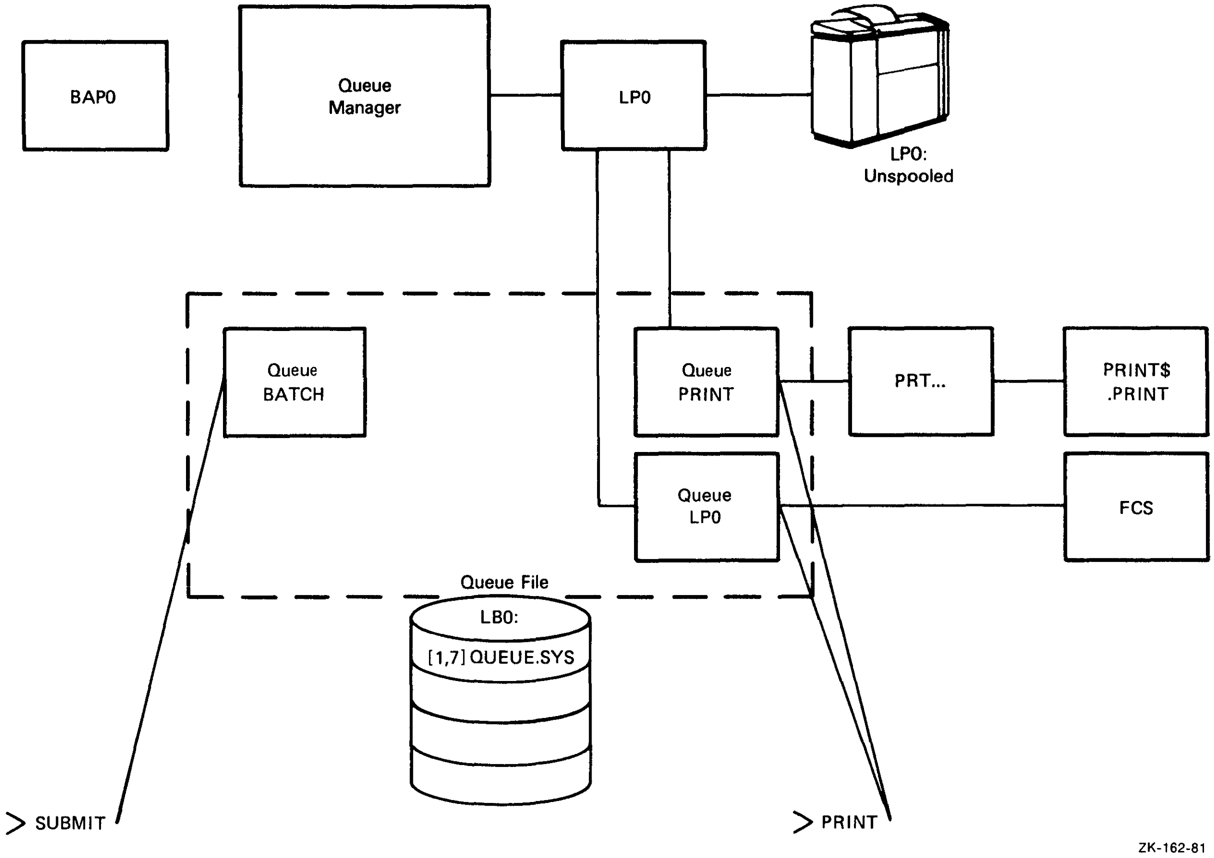
```
DCL>

MCR>
```

You can install as many as 16 batch processors with names in the form BAPnnn, where nnn represents one to three Radix–50 characters.



9. Initialize the batch processors.

```
DCL>

MCR>
```



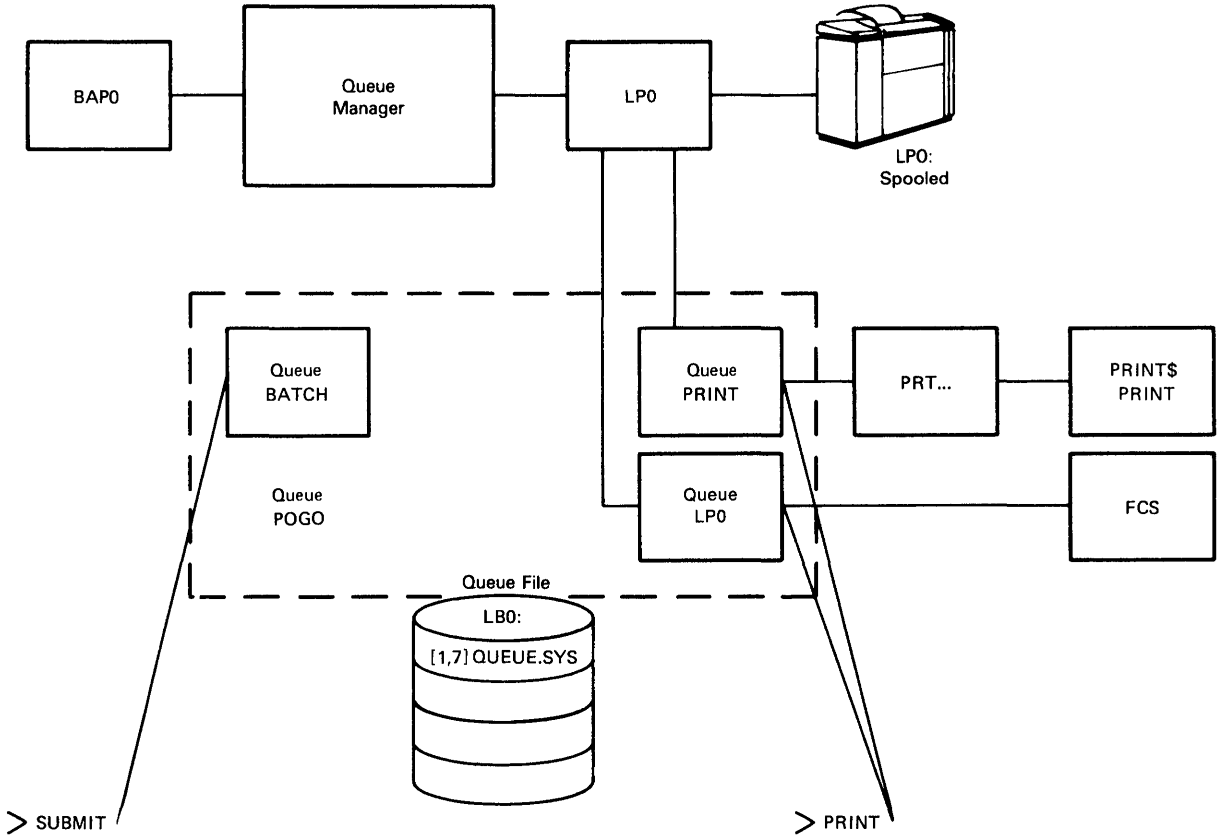
10. Initialize the batch queues.

```
DCL>
```

```
MCR>
```

The default queue BATCH is initialized with the command START/QUEUE/MANAGER, but you can initialize as many as 15 additional batch queues with names consisting of up to 6 alphanumeric characters.

Note that, unlike print processors, batch processors do not need device-specific queues. There are no peripheral devices associated with batch processors.

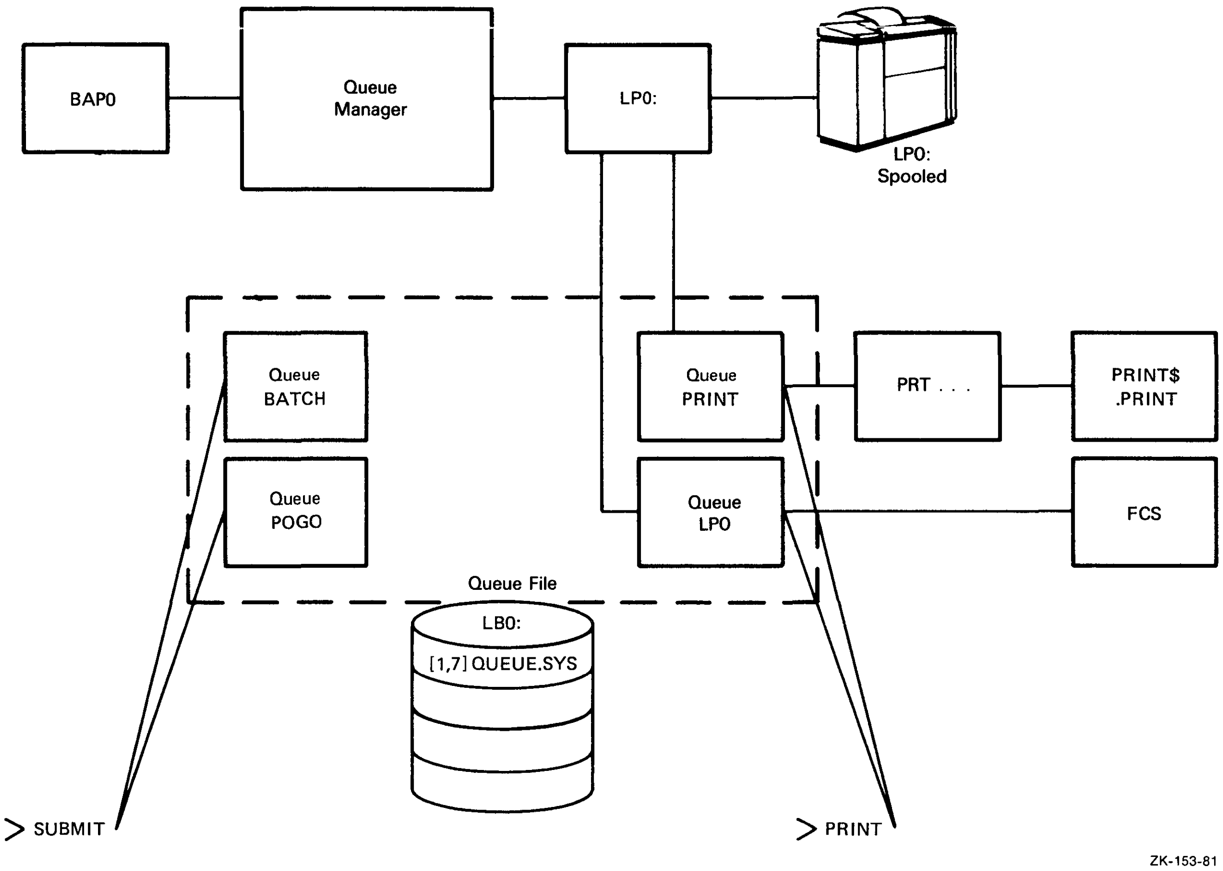


ZK-163-81

11. Assign one or more batch queues to the processor.

    ```
    DCL>
    MCR>
    ```

    If you initialize a queue but do not assign it to a processor, jobs can be queued but they are not processed.



3.4 Using the Queue Manager to Spool Listing Files Automatically

Certain utilities and compilers (such as the Task Builder (TKB) and the Fortran compiler (F77)) provide switches for spooling listing files to the Queue Manager (QMG) automatically. The listing file is placed in the PRINT queue upon completion of the task build or compilation. However, to provide support for this feature, the task QMGPRT.TSK must be installed.

RSX-11M-PLUS operating systems (excluding RL02 distribution kits) are distributed with the startup command file LB:[1,2]QMGSTART.CMD, which installs QMGPRT.TSK for you. However, to install QMGPRT.TSK on a Micro/RSX or pregenerated RSX-11M-PLUS system, you must create your own secondary startup command file. Include one of the following command lines in the file:

```
.IFNINS PRT... INS $QMGPRT/SLV=NO              (if your CLI is MCR)

.IFNINS PRT... INSTALL/NOSLAVE $QMGPRT         (if your CLI is DCL)
```

Then, add the following statement to the system configuration data file LB:[1,2]SYSPARAM.DAT:

```
STARTFILE=filename
```

Specify the full file specification (device, directory, and file name) for the secondary command file.

By default, after you install QMGPRT.TSK, listing files are spooled to the PRINT queue automatically. If you do not want a listing file printed, you must specify the appropriate switch when you enter the command line.

For example, when you use the LINK command with the /MAP qualifier, TKB creates a task image file (file type TSK) and a memory allocation file (file type MAP). If the system task QMGPRT.TSK is installed, TKB calls QMG to spool your map file to the PRINT queue. If you do not want the map file to be printed, you must specify the /NOPRINT qualifier in the LINK command line.

If you do not install QMGPRT.TSK, you cannot print map or listing files directly. For example, if you are using TKB, you cannot use the /SP switch. Instead, you must enter a separate PRINT command line. The TKB command line tells TKB to create a task file and a map file (without printing it); the PRINT command tells the system to print the map file.

For more information on using TKB, see the *RSX-11M-PLUS and Micro/RSX Task Builder Manual.* For information on using automatic spooling with other utilities, see the manual that corresponds to each utility.

3.5 Queue Manager Command Descriptions

DCL and MCR commands control the Queue Manager (QMG). The commands for both CLIs invoke the same functions, but the command syntax is different. Use the command syntax that is best for you. Both forms of syntax appear in the command descriptions; MCR commands or command elements appear in parentheses. In the examples, commands are identified by the explicit DCL> and MCR> prompts.

General Format for DCL Commands

```
DCL>DCLcommand/QMGqualifier[s] parameter[s][/qualifier[s]]
```

General Format for MCR Commands

```
MCR>QUE ddnn: /function[:option[s]][/sw[s]]
```

Processors, devices, and device-specific queues all have names derived from the name of the spooled device. Thus, if LP1: is the spooled device, the corresponding processor is called LP1. Likewise, the device-specific queue is called LP1.

In DCL command lines, when you specify the name of a processor, device, or queue, the colon (:) is optional. However, the colon is required in MCR command lines. In MCR, include the colon as part of a device name and to terminate a queue name. For example, in an MCR command line, use the name XPRESS: to refer to a queue named XPRESS.

The sections that follow describe the commands for setting up and running QMG. For information on using QMG (including the commands for print and batch processing operations), see the *RSX-11M-PLUS Batch and Queue Operations Manual.*

Table 3–1 provides an overview of the QMG commands. The command descriptions in the sections that follow appear in the order listed in the table.

Table 3–1: Queue Manager Command Summary

DCL Command	MCR Command	Function
INITIALIZE	QUE	
/QUEUE	/CR	Creates, names, and initializes a queue.
/PROCESSOR	/SP	Creates, names, and initializes a processor (print or batch) or despooler.
DELETE		
/QUEUE	/DEL:Q	Deletes a queue by name.
/PROCESSOR	/UNSP	Deletes a processor or despooler by name. Sets a device unspooled.
ASSIGN		
/QUEUE	/AS:	Establishes a path from a queue to a processor.
DEASSIGN		
/QUEUE	/DEA	Eliminates the path from a queue to a processor.
STOP		
/ABORT	/KIL	Deletes an active job on a processor.
/PROCESSOR	/STO	Stops a processor.
/QUEUE	/STO:QUE	Stops queues.
/QUEUE/MANAGER	/STO:QMG	Stops QMG and deletes all processors.
START		
/QUEUE	/STA:QUE	Starts a queue.
/PROCESSOR	/STA	Starts a processor or despooler.
/QUEUE/MANAGER	/STA:QMG	Starts QMG and initializes default queues. Creates the queue file LB0:[1,7]QUEUE.SYS. Clears all queue assignments.

3.5.1 INITIALIZE/QUEUE (QUE /CR)

The INITIALIZE/QUEUE (QUE /CR) command creates, names, and initializes a queue.

Formats

```
DCL>INITIALIZE/QUEUE queuename/qualifier
```

```
MCR>QUE ddnn:/CR:typ
```

DCL Parameter Qualifiers	MCR Queue Types
/BATCH	BATCH
/PRINT	PRINT

Parameters

queuename

Specifies the name of the queue to be initialized.

Queue names BATCH and PRINT are reserved, because the BATCH and PRINT queues are initialized by the START/QUEUE/MANAGER (QUE /STA:QMG) command.

Initialize a queue for every output device to which QMG is to send print jobs. These are called *device-specific* queues. Otherwise, any six alphanumeric characters make an acceptable queue name for a general queue. For example, if you have no processor named LP1, then you can use LP1 as a general queue name.

Initialize device-specific queues before you initialize the associated processor. After both the queue and the processor are initialized, the queue can accept print jobs. In addition, the named printer is set *spooled;* it is under the control of QMG.

ddnn

Specifies the device mnemonic (dd) and unit number (nn) for the output device.

DCL Parameter Qualifiers and MCR Queue Types

/BATCH (BATCH)
/PRINT (PRINT)

Tells QMG what kind of queue is being initialized. Queues for applications processors are considered print queues.

The default qualifier is /PRINT (PRINT). However, for clarity, specify one of the two qualifiers when you initialize a queue.

Notes

1. No more than 16 print queues (including queues for applications processors) can be initialized. This count includes the default PRINT and BATCH queues.
2. Once a queue is established, it can remain in the system across a system failure. However, to avoid an accidental loss, you should reestablish print queues each time you bootstrap the system.

INITIALIZE/QUEUE
QUE /CR

Examples

```
DCL>
```

```
MCR>
```

Creates, names, and initializes the print queue XPRESS. The queue now accepts print jobs, but it must be assigned to an output processor before jobs can be taken from the queue.

```
DCL>
```

```
MCR>
```

Assigns the queue initialized in the previous example to a print processor.

```
DCL>
DCL>
```

```
MCR>
MCR>
```

Creates, names, and initializes a print queue named LP0. This is a device-specific queue whose name is derived from the name of an output device. This queue must be initialized before the output despooler can be initialized. After both the queue and output despooler are initialized, the queue can accept print jobs and the named printer is set spooled.

3.5.2 INITIALIZE/PROCESSOR (QUE /SP)

The INITIALIZE/PROCESSOR (QUE /SP) command creates, names, and initializes an output despooler or batch processor. When you initialize a print processor, you set the associated output device *spooled.* A *spooled* output device is a hardware device (usually a line printer) and an associated processor that are under the control of the Queue Manager (QMG). The spooled device services only one request at a time and processes only those jobs that have been passed to it by QMG.

Processors, devices, and device-specific queues all have names derived from the name of the spooled device. Thus, if LP1: is the spooled device, the corresponding processor is called LP1. Likewise, the device-specific queue is called LP1:. However, when you specify the name of a processor, device, or queue in a DCL command line, the colon (:) is optional.

Before you initialize the processor, the task must be installed. After you initialize the processor, it "owns" the printer. All jobs sent to a processor are spooled to the printer that it owns.

Formats

```
DCL>INITIALIZE/processortype processorname[/qualifier]

MCR>QUE processorname:/SP[/switch]
```

DCL Processor Types	MCR Functions
/PROCESSOR	/SP
/PRINTER	/SP
/DEVICE	/SP
/BATCH_PROCESSOR	/BA
/CARD_READER	/INPUT
/INPUT	/INPUT
/APPLICATIONS_PROCESSOR	/SP/EX

DCL Qualifiers	MCR Switches
/BATCH_QUEUE:queuename	/QUEUE::queuename
/CONSOLE:ddnn:	/TERM:ddnn:
/FLAG_PAGE:n	/FL:n
/FORMS:n	/FO:n
/LOWERCASE	/LOW
/NOWARNINGS	/NM
/PRINTER_QUEUE:queuename	/QUEUE:queuename
/SHAREABLE	/SHR
/UPPERCASE	/NOLO

Parameters

/processortype

Specifies processor types, which are divided into the following groups:

- /PROCESSOR, /PRINTER, and /DEVICE initialize a print processor (also called an output despooler). They are synonyms; the effect of each is the same. However, the synonyms are provided to improve the clarity of commands and documentation. Use /PRINTER for printers and /DEVICE for other output devices. /PROCESSOR can be used for any output device.

The MCR function /SP performs the same function. There are no synonyms in the MCR form of this command.

Note that a device-specific queue must exist before the output despooler can be initialized (see Section 3.5.1).

- /BATCH_PROCESSOR (/BA) initializes a batch processor.
- /CARD_READER and /INPUT initialize a processor associated with a card reader device (also called an input despooler). They are synonyms; the effect of each is the same. The MCR function is /INPUT.
- /APPLICATIONS_PROCESSOR (/SP/EX) initializes a non-physical output despooler. Use this processor type when the destination of the output is a software application task or network. Also, use this type to initialize a user-written output despooler that currently has no physical device in the system.

  When you specify /APPLICATIONS_PROCESSOR (/SP/EX), the Queue Manager does not check for the existence of a physical device before initializing the processor. Thus, the Queue Manager can despool output to an applications task, to a network, or to a remote location.

  In MCR format, you must combine the /SP and /EX functions.

processorname

Specifies the names of the device, processor, and queue; the names are interchangeable. MCR users must include the colon (:), but DCL users can omit it. If you are initializing a batch processor, specify the name in the form BAPn, where n is a decimal number from 0 to 15.

You must initialize the device-specific queue before you initialize the print processor with the same name and unit number (see Section 3.5.1).

DCL Qualifiers and MCR Switches

With the exception of setting the number of flag pages, these qualifiers and switches have no effect on the printed output of the printer being initialized. Instead, these qualifiers define the kind of print jobs the processor can accept. (Different kinds of print jobs are established by the PRINT command that places them in the queue. For information on modifying the printed output of a print job, see the *RSX-11M-PLUS Batch and Queue Operations Manual* or the *Micro/RSX User's Guide, Volume 2*.)

A description of each DCL qualifier (MCR switch) follows in alphabetical order.

/BATCH_QUEUE:queuename (/QUEUE::queuename)
/PRINTER_QUEUE:queuename (/QUEUE:queuename)

Specifies default batch and print queues for jobs read from the card reader. The default queues are BATCH and PRINT, respectively.

/CONSOLE:ddnn: (/TERM:ddnn:)

Specifies which terminal will be used for messages that are displayed by the card reader processor. The default for ddnn is CO0:.

/FLAG_PAGE:n (/FL:n)

Specifies how many flag pages are to precede jobs and files printed by this processor. The value n must be 0, 1, or 2; the default is 0.

If the processor is being initialized to accept special forms, set this qualifier to 0.

To change the value, use the /FLAG qualifier to the START command (/FL switch to /STA option).

/FORMS:n (/FO:n)

Specifies the forms that the printer can accept. The value for n can range from 0 to 255 bytes. QMG truncates records longer than 255 bytes. The default is 0.

Users specify these values in their PRINT command lines. If an available printer does not support the requested forms, the printer stops and print jobs must wait. To restart the printer, change the value or values for /FORMS (/FO).

The lengths and widths of forms differ. If you have an RSX-11M-PLUS system, you can change both values in the forms table located in LB:[1,24]LPPBLD.CMD. The file contains an explanation of the procedure, which is briefly summarized as follows:

- To set the width, determine how many columns are needed to fill in the form. There are no standard widths for forms, but 80 and 132 are the most common.
- To set the length, determine whether your forms are of a standard length. All standard-length forms use form feeds, which are set on the hardware. All nonstandard forms use simulated form feeds, which are set through the software by replacing the form-feed character with the appropriate number of line feeds.

Determine which forms will be used on your system and assign each a number. Then, edit the file LB:[1,24]LPPBLD.CMD to set up the forms table for the print processor to correspond to the numbers you have assigned to the forms. After you complete the forms table, use the LINK (TKB) command to task build the processor.

The following table lists the standard form types and settings for RSX-11M-PLUS and Micro/RSX operating systems:

Form Type	Width	Length	Form Feed
0	132	66	Real
1	80	66	Real
2	132	2	Simulated
3	132	3	Simulated
(4–66)	132	(4–66)	Simulated

The values for these form types represent the number of lines that can be printed on each page before a form feed (that is, the page length). For example, if you want your printed page to be 10 lines long, specify /FORMS:10 in the command line. If you want your printed page to be 50 lines long, specify /FORMS:50.

The following table shows the forms requirements for a hypothetical installation. The form names describe different uses for these special forms.

Form	Assigned n	Width	Length
Accordion	0	132	66
Paychecks	1	40	10
Invoices	2	80	66
Stationary	3	80	66
Moon Diary	4	22	43
Star Log	5	112	113

By initializing the print processor with different values for /FORMS, you establish the width (80 or 132 columns) and the quality (draft or letter) of different printer modes.

The /FORMS qualifier also has special settings for certain types of printers. They are as follows:

Form Type	Width	Length	Printer Type
124	80	66	LN03 printers
125	80	66	LA50 printers
126	80 or 132	66	LA100 and LA75 printers
127	80 or 132	66	LN01 printers

To print a file after you have initialized a processor with one of these form types, select the correct printer mode by including one of the following form types in the PRINT command line:

Form Type	Printer Mode
3	Specifies 132-column, draft quality. Also, specifies landscape mode for LN01 and LN03.
4	Specifies 80-column, letter quality. Also, specifies portrait mode for LN01 and LN03.
5	Specifies 132-column, letter quality. Also, specifies landscape mode for LN01 and LN03.
6	Specifies 80-column, draft quality. Also, specifies portrait mode for LN01 and LN03.

For example, to print pages that are 80 columns wide and letter quality, use the following command line:

```
$
```

Note that letter-quality printing is a feature of the printer itself; the PRINT command line simply selects the feature by specifying the correct form type.

If you have processors attached to any form types other than 0 (for example, print processors attached to FORMS:1 and FORMS:2), always specify the forms type in the PRINT command line. If you do not, QMG assigns your job to the default form type (FORMS:0) and your job is not printed.

/LOWERCASE (/LOW)

Specifies the lowercase attribute for a printer (that is, the printer can accept print jobs with /LOWERCASE specified in the PRINT command line).

Note that QMG does not prevent you from initializing a printer with the lowercase attribute, even if the hardware does not actually support a lowercase character set. So, if a printer with the lowercase character set is temporarily unavailable, you can initialize another printer with the /LOWERCASE qualifier. Then, although the printer cannot print lowercase characters, it accepts print jobs with /LOWERCASE specified in the PRINT command line.

In general, do not use this qualifier unless the printer actually has both the uppercase and lowercase character sets.

Note

Due to lowercase-to-uppercase translation, the device driver for a printer affects the printer's output. For more information, see the description of the DCL command SET DEVICE in the *RSX-11M-PLUS Command Language Manual* or the *Micro/RSX User's Guide, Volume 2*, or see the description of the MCR command DEV in the *RSX-11M-PLUS MCR Operations Manual.*

/NOWARNINGS (/NM)

Reduces the amount of information that is displayed on the console terminal when the system is bootstrapped and STARTUP.CMD is invoked. For example, if you use the MCR command QUE LPO:/CR/NM, the Queue Manager (QMG) does not display the message "Queue already exists." The /NOWARNINGS qualifier (/NM switch) suppresses this message.

/SHAREABLE (/SHR)

Establishes the processor as shareable. The physical device is not "owned" by a processor if it is not printing a job. If you do not specify /SHAREABLE, the output processor attaches the device when it is initialized. No other job can access the device until it is detached, which happens when the device is despooled.

However, if you initialize the processor as shareable, the device is attached at the beginning of the QMG print job and is detached at the end of the job, leaving the device free for other uses. Note that transparent spooling to a shareable device cannot occur.

If you want to spool to the null device (NL:), specify the /SHAREABLE qualifier when you initialize the print processor.

/UPPERCASE (/NOLO)

Specifies the no-lowercase attribute for a printer (that is, the printer cannot accept print jobs with /LOWERCASE specified in the PRINT command line). Do not use this qualifier unless the hardware involved does not have the lowercase character set.

This is the default.

Restrictions

If you are initializing a processor for a physical device (such as a printer or a terminal), the system checks for the device and the device-specific queue. Before the processor is initialized, the following conditions must be met:

- The device must be part of the current system. That is, the device must be physically connected to the system and included in your system's configuration.
- For each print processor, there must be a device-specific queue with a name derived from the name of the output device. The processor cannot be initialized until the device-specific queue has been created.

These restrictions do not apply when you initialize batch processors or applications processors. Neither a physical device nor a device-specific queue is needed to initialize a batch processor or applications processor.

Notes

1. You can initialize up to 16 batch processors or 16 print processors (including applications processors).
2. You can initialize print queues whose names do not correspond to output devices. These are called *general queues.*
3. The /LOWERCASE and /UPPERCASE qualifiers do not modify the printer's output. Instead, they set up a "mask" that must be matched by the attributes of print jobs before they can be sent to the printer. In general, initialize printers with their actual attributes (for example, specify /UPPERCASE if the printer does not have the lowercase character set).

Examples

```
DCL>

MCR>
```

Sets LP0: spooled. Jobs can be passed to LP0: from QMG. You must initialize a device-specific queue named LP0 before you can initialize the processor for device LP0:.

```
DCL>

MCR>
```

Initializes a print processor to control LP1: and specifies that jobs run on this printer will be preceded by two flag pages. Jobs that specify /FLAG_PAGE in the PRINT command line will have two flag pages at the beginning of the job.

```
DCL>

MCR>
```

Initializes a print processor to control LP1: and states that the printer will only accept jobs specifying /UPPERCASE in the PRINT command line.

```
DCL>

MCR>
```

Creates, names, and initializes batch processor BAP0.

```
DCL>

MCR>
```

Creates, names, and initializes print processor LP0. The printer has the lowercase character set. Jobs will have two flag pages when printed.

DELETE/QUEUE
QUE /DEL:Q

3.5.3 DELETE/QUEUE (QUE /DEL:Q)

The DELETE/QUEUE (QUE /DEL:Q) command is a privileged command that deletes queues by name.

Formats

```
DCL>DELETE/QUEUE  queuename/ERASE
```

```
MCR>QUE queuename:/DEL:Q
```

Parameter

queuename/ERASE (/DEL:Q)

Specifies the name of a queue to be deleted. Only a privileged user can delete a queue by using this parameter. The default queues BATCH and PRINT cannot be deleted.

If the queue has jobs in it, it cannot be deleted immediately. Instead, the queue is marked for deletion until the last job is processed. Then, the queue itself is deleted; no new jobs can be entered into the queue.

You cannot delete device-specific queues until after you have deleted the processor with the same number (see Section 3.5.4).

Examples

```
DCL>
```

```
MCR>
```

Deletes the queue named MONA, if the queue is empty. If the queue has entries, it is marked for deletion and is deleted as soon as the last job is removed from the queue. In the MCR example, note that the colon in the queue name is required.

3.5.4 DELETE/PROCESSOR (QUE /UNSP)

The DELETE/PROCESSOR (QUE /UNSP) command deletes print processors, output despoolers, or batch processors by name. The command also sets the device unspooled.

The DELETE/PROCESSOR (QUE /UNSP) command counteracts the INITIALIZE/PROCESSOR (QUE /SP) command.

Formats

```
DCL>DELETE/processortype processorname
```

```
MCR>QUE processorname:/UNSP
```

DCL Processor Types	MCR Functions
/PROCESSOR	/UNSP
/PRINTER	/UNSP
/DEVICE	/UNSP
/BATCH_PROCESSOR	/UNBA
/APPLICATIONS_PROCESSOR	/UNSP

Parameters

/processortype

Specifies the processor types, which are divided into the following groups:

- /PROCESSOR, /PRINTER, and /DEVICE delete a print processor (also called an output despooler). They are synonyms; the effect of each is the same. However, the synonyms are provided to improve the clarity of commands and documentation. Use /PRINTER for printers and /DEVICE for other output devices. /PROCESSOR can be used for any output device.

  There are no synonyms in the MCR form of this command; the /UNSP switch performs the same function for all types of output devices.

- /BATCH_PROCESSOR (/UNBA) deletes a batch processor.
- /APPLICATIONS_PROCESSOR (/UNSP) deletes a nonphysical output despooler. Use this processor type when the destination of the output is a software application task or network. Also, use this type to delete a user-written output despooler that currently has no physical device in the system. This makes it possible for the Queue Manager (QMG) to despool output to an applications task, to a network, or to a remote location.

Note that you must delete the output despooler before you can delete the device-specific queue.

processorname

Specifies the name of the processor. For physical devices, the name of the processor is identical with the name of the device. Thus, the device LP1: is controlled by processor LP1:. DCL commands do not require that you include the colon (:), but you must include the colon in all MCR commands.

Batch processors have names in the form BAPnnn, where nnn is one to three Radix–50 characters.

DELETE/PROCESSOR
QUE /UNSP

Applications processors have names of six Radix–50 characters.

Examples

```
DCL>
```

```
MCR>
```

Deletes processor LP0:, the owner of line printer LP0:.

```
DCL>
```

```
MCR>
```

Deletes the batch processor named BAP2. Note the colon in the MCR command.

```
DCL>
DCL>
```

```
MCR>
MCR>
```

Shows the steps to follow to delete a device-specific queue.

3.5.5 ASSIGN/QUEUE (QUE /AS)

The ASSIGN/QUEUE (QUE /AS) command establishes a path from a queue to a processor.

This command can be used to redirect output from one processor to another or to group related processors, such as plotters, under a single queue.

Formats

```
DCL>ASSIGN/QUEUE queuename processorname
```

```
MCR>QUE processorname:/AS:queuename
```

Parameters

queuename

Specifies the queue that is to be assigned to the processor.

processorname

Specifies the processor to which the queue is to be assigned. When you assign queues to output devices, use the device name in the form ddnn.

DEASSIGN/QUEUE
QUE /DEA

3.5.6 DEASSIGN/QUEUE (QUE /DEA)

The DEASSIGN/QUEUE (QUE /DEA) command eliminates the path from a queue to a processor. It counteracts the ASSIGN/QUEUE (/AS) command.

Formats

```
DCL>DEASSIGN/QUEUE queuename processorname
```

```
MCR>QUE processorname:/DEA:queuename
```

Parameters

queuename

Specifies the queue that is to be deassigned from the processor.

processorname

Specifies the processor from which the queue is to be deassigned. When you deassign queues from output devices, use the device name in the form ddnn.

3.5.7 STOP/ABORT (QUE /KIL)

The STOP/ABORT (QUE /KIL) command deletes the active job on a given processor. Privileged users can delete any job; nonprivileged users can delete their own jobs. To delete the job, you need to know the name of the processor.

Formats

```
DCL>STOP/ABORT processorname
```

```
MCR>QUE processorname:/KIL
```

Parameter

processorname

Specifies the processor whose active job is to be deleted. Note that the MCR command format requires a colon (:) after the processor name.

You can stop jobs running on any processor under the control of QMG, including batch processors.

Notes

1. Use this command to stop a processor quickly (for example, if a job is printing only form feeds).
2. As soon as the active job is deleted, QMG passes the next eligible job to the processor. The command deletes only the job; it does not abort the processor.
3. You can also delete the active job on a card-reader processor with this command.

Example

```
DCL>
Processor?
```

Stops a print job on LP0. The currently active job is deleted from the queue and the next eligible job is queued.

3.5.8 STOP/PROCESSOR (QUE /STO)

The STOP/PROCESSOR (QUE /STO) command stops a processor.

Formats

```
DCL>STOP/processortype processorname[/qualifier]

MCR>QUE processorname:/STO[:opt]
```

DCL Processor Types	MCR Function
/PROCESSOR	/STO
/PRINTER	/STO
/DEVICE	/STO
/BATCH_PROCESSOR	/STO
/APPLICATIONS_PROCESSOR	/STO

DCL Parameter Qualifiers	MCR Options
/ABORT	:AB
/FILE_END	:EOF
/JOB_END	:EOJ
/PAUSE	(No equivalent)

Parameters

/processortype

Specifies the processor types, which are divided into the following groups:

- /PROCESSOR, /PRINTER, and /DEVICE stop a print processor (also called an output despooler). They are synonyms; the effect of each is the same. However, the synonyms are provided to improve the clarity of commands and documentation. Use /PRINTER for printers and /DEVICE for other output devices. /PROCESSOR can be used for any output device.
- /BATCH_PROCESSOR (/STO) stops a batch processor.
- /APPLICATIONS_PROCESSOR (/STO) stops a nonphysical output despooler. Use this processor type when the destination of the output is a software application task or network. Also, use this type to stop a user-written output despooler that currently has no physical device in the system.

  When you specify /APPLICATIONS_PROCESSOR, QMG does not check for the existence of a physical device before stopping the processor.

The MCR function /STO performs the same functions; there are no synonyms in the MCR form of this command.

processorname

Specifies the processor to which the queue is assigned. To stop print processors, use the device name in the form ddnn.

This command has no effect on the status of jobs in queues, but jobs are not dequeued while the processor is stopped.

DCL Qualifiers and MCR Options

Only one qualifier (or option) is permitted. The choice of qualifier (or option) here affects the restarting of jobs when the processor is started again.

/ABORT (:AB)
Stops the processor immediately. The current job is held in its queue.

/FILE_END (:EOF)
Stops the processor at the end of the current file.

/JOB_END (:EOJ)
Stops the processor at the end of the current job.

/PAUSE
Stops the processor at the end of the current line. There is no MCR qualifier; /STO is sufficient. This is the default.

Note

This command is counteracted by the START/PROCESSOR (QUE /STA) command. However, you cannot enter the START/PROCESSOR (QUE /STA) command until the processor actually stops.

STOP/QUEUE
QUE /STO:QUE

3.5.9 STOP/QUEUE (QUE /STO:QUE)

The STOP/QUEUE (QUE /STO:QUE) command stops queues. No jobs can be taken from the queue, but jobs can still be added to the queue. If a job is active at the time this command is issued, the job is allowed to complete processing, but no additional jobs are started.

Formats

```
DCL>STOP/QUEUE queuename
```

```
MCR>QUE queuename:/STO:QUE
```

Parameter

queuename

Specifies the queue to be stopped. Note the colon (:) in the MCR form of the command.

3.5.10 STOP/QUEUE/MANAGER (QUE /STO:QMG)

The STOP/QUEUE/MANAGER (QUE /STO:QMG) command stops the Queue Manager (QMG) and deletes all processors (including batch processors or output despoolers).

If you issue this command without the /ABORT (:AB) qualifier, QMG is marked for stop. As soon as all currently active jobs on all processors are done, QMG stops. No more jobs can be entered in the queue.

Formats

```
DCL>STOP/QUEUE/MANAGER[/ABORT]
```

```
MCR>QUE /STO:QMG[:AB]
```

Parameter

/ABORT (:AB)

Holds all active jobs and exits QMG immediately.

3.5.11 START/PROCESSOR (QUE /STA)

The START/PROCESSOR (QUE /STA) command starts a batch processor or output despooler.

Formats

```
DCL>START/processortype processorname [/qualifier[s]]
```

```
MCR>QUE processorname:/STA[:opt[s]]
```

DCL Processor Types	MCR Function
/PROCESSOR	/STA
/PRINTER	/STA
/DEVICE	/STA
/BATCH_PROCESSOR	/STA
/APPLICATIONS_PROCESSOR	/STA

DCL Qualifiers	MCR Options
/ALIGN	/AL
/BACKSPACE:n	:BA:n
/CONTINUE	(No equivalent)
/FLAG:n	/FL:n
/FORMS:n	/FO:n
/FORWARDSPACE:n	:FW:n
/NEXT_JOB	:NE
/PAGE:n	:PA:n
/RESTART	:RES
/TOP_OF_FILE	:PA:0

Parameters

/processortype

Specifies the DCL processor types, which are divided into the following groups:

- /PROCESSOR
- /PRINTER
- /DEVICE
- /BATCH_PROCESSOR
- /APPLICATIONS_PROCESSOR)

The effect of each is the same. These synonyms are provided to assure unambiguous commands and documentation. Use /PRINTER for line printers, /DEVICE for other output devices, /BATCH_PROCESSOR for batch processors, and /APPLICATIONS_PROCESSOR for applications processors.

In MCR format, use the /STA function.

processorname

Specifies the processor to be started. When starting a stopped output processor, use the device name in the form ddnn.

DCL Qualifiers and MCR Options

Some qualifiers (options) have meaning for starting either batch or print processors, and some have meaning for print processors only. The following list summarizes the restrictions:

- The qualifiers /CONTINUE (the default for /STA), /RESTART (/RES), /NEXT_JOB (/NE), and /TOP_OF_FILE (PA:0) are appropriate for both print and batch processors.
- Page-related qualifiers (:n) have meaning for print processors only. They have no effect unless the user specifies a page length qualifier in the PRINT command.
- /FORMS: (/FO:) and /FLAG: (/FL:) have meaning for print processors only.

A description of each qualifier follows. The qualifiers appear in alphabetical order.

/ALIGN (/AL)

Informs the print processor that the paper in the line printer is currently at the top of a form. Use /ALIGN (/AL) when you have stopped the print processor to change forms. The print processor maintains a count of how many lines it has descended on a form. The /ALIGN (/AL) qualifier clears that count for new forms.

/BACKSPACE:n (:BA:n)

Specifies that the processor start n pages back in the current job.

/CONTINUE

Specifies that the processor carry on from the point where it was stopped. MCR format does not require an option; this is the default action for the START/PROCESSOR (QUE /STA) command.

/FLAG:n (/FL:n)

Overrides the flag value set when the processor was initialized. The value n is the number of flag pages printed at the beginning of the job.

When the processor starts again, the Queue Manager (QMG) sends the next job that matches the attributes of the processor.

If you are restarting a processor to handle special forms, specify /FLAGS:0 (/FL:0) to avoid having a banner page printed on your special forms.

/FORMS:n (/FO:n)

Overrides the /FORMS value set when the processor was initialized. The value n can be any number from 0 to 255.

For more information on setting the /FORMS value, see Section 3.5.2.

/FORWARDSPACE:n (:FW:n)

Specifies that the processor start n pages forward in the current job.

/NEXT_JOB (:NE)

Specifies that the processor start at the beginning of the next job.

/PAGE:n (:PA:n)

Specifies that the processor start at page n in the job that was active when the processor was stopped.

START/PROCESSOR
QUE /STA

/RESTART (:RES)

Specifies that interrupted jobs be restarted from the beginning of the file that was being processed when the processor was stopped. This qualifier affects only jobs that were active at the time the processor stopped.

/RESTART (:RES) conflicts with the use of the /NORESTART qualifer in the PRINT or SUBMIT command lines. Use the command SHOW QUEUE to check for such jobs before you use this qualifier. To finish jobs started with the /NORESTART qualifier, use the /CONTINUE qualifer.

The effect of /RESTART (:RES) depends on the qualifier (or switch) used with the STOP/PROCESSOR (QUE /STO) command to stop the processor, as follows:

- If the qualifier was /PAUSE (the default for /STO), then any jobs active at that time are restarted at their beginning.
- If the qualifier was /FILE_END (:EOF), then jobs active at that time are restarted at their beginning. If the active file was the last one in the job, no jobs are restarted.
- If the qualifier was /ABORT (:AB) or if the system halted, the current active job is held in its queue. /RESTART (:RES) has no effect.
- If the qualifier was /JOB_END (:EOJ), no job is currently active; /RESTART (:RES) has no effect.

/TOP_OF_FILE (:PA:0)

Specifies that the processor start at the top of the current file in the current job.

Notes

1. Choose carefully between the /RESTART (:RES) and /CONTINUE (MCR default) qualifiers. /RESTART may conflict with the desires of users with print or batch jobs active at the time a processor is stopped.

   For example, if a user specifies /NORESTART (/RES) in a PRINT command line, and you stop and restart the processor on which the print job is active, the user expects the job to continue printing from the point at which it was stopped. However, if you start the processor with the /RESTART (:RES) qualifier, the job is not continued. Instead, contrary to what the user wanted to happen, the job is restarted at the top of the file.

   If jobs with the no-restart attribute are active when you stop a processor, specify the /CONTINUE (MCR default) qualifier to restart the processor. /CONTINUE continues printing the current job from the point at which it was stopped.

2. If you restart a processor to handle special forms, set the /FLAGS qualifier (:FL option) to 0 to avoid having a banner page printed on your special forms.

Examples

```
DCL>
```

```
MCR>
```

Starts print processor LP0:. Printing continues from the point at which it was stopped.

```
DCL>
```

```
MCR>
```

Starts print processor LP0:. The processor inserts two flag pages at the beginning of the job and at the beginning of each file if the user specifies /FLAG_PAGE in the PRINT command line.

```
DCL>
```

```
MCR>
```

Starts batch processor BAP0. Batch processing continues from the point at which it was stopped.

```
DCL>
```

```
MCR>
```

Starts the batch processor BAP0.

```
DCL>
```

```
MCR>
```

Starts print processor LP1:. The job active at the time the processor was stopped is restarted.

START/QUEUE
QUE /STA:QUE

3.5.12 START/QUEUE (QUE /STA:QUE)

The START/QUEUE (QUE /STA:QUE) command starts a queue that has previously been stopped. START/QUEUE (QUE /STA:QUE) counteracts STOP/QUEUE (QUE /STO).

Formats

```
DCL>START/QUEUE queuename

MCR>QUE queuename:/STA:QUE
```

Parameter

queuename

Specifies the queue to be started. Note the colon (:) in the MCR format.

3.5.13 START/QUEUE/MANAGER (QUE /STA:QMG)

The START/QUEUE/MANAGER (QUE /STA:QMG) command starts the Queue Manager (QMG), initializes the default queues PRINT and BATCH, and creates the queue file LB0:[1,7]QUEUE.SYS (unless it already exists). The command also clears all queue assignments.

If the queue file exists, all queues still contain their jobs, but the jobs are not assigned to processors.

START/QUEUE/MANAGER (QUE /STA:QMG) counteracts STOP/QUEUE/MANAGER (QUE /STO:QMG).

Formats

```
DCL>START/QUEUE/MANAGER
```

```
MCR>QUE /STA:QMG
```

Examples

```
DCL>
```

```
MCR>
```

Starts QMG and initializes the default queues PRINT and BATCH, if they are not already initialized. (However, the default queue BATCH is neither accessible nor displayed.) In addition, the command clears all assignments of queues to processors.

Any jobs that were active at the time QMG was stopped are held when QMG is started again. All other jobs retain the status they had when QMG was stopped; all processors must be reinitialized.

```
DCL>
```

```
MCR>
```

Stops QMG and deletes all processors.

```
DCL>
```

```
MCR>
```

Clears all assignments of queues to processors.

START/QUEUE/MANAGER
QUE /STA:QMG

```
DCL>
DCL>
DCL>
DCL>
DCL>
DCL>
DCL>
DCL>
```

```
MCR>
MCR>
MCR>
MCR>
MCR>
MCR>
MCR>
MCR>
```

Re-creates the network of queues and processors.

```
DCL>
```

```
MCR>
```

Releases a held job that was active at the time the system halted.

3.6 Error Messages

The following sections provide a list of error messages and suggestions for appropriate responses. The messages are divided into three groups: messages from the Queue Manager (QMG), messages from the output despooler, and messages from the batch processor (preceded by the mnemonic BPR). Each group is described separately. Within each section, the messages are listed in alphabetical order according to the text of the message.

3.6.1 Error Messages from Queue Manager Commands

This section lists error messages, describes the meaning of the messages, and suggests actions to correct the errors. Each error message is preceded by the first three letters of the command that failed. For example:

```
QUE -- JOB DOES NOT EXIST
```

In the following list of messages, the letters com represent the first three letters of the command that encountered the error.

com—Bad command line

Explanation: The command line could not be understood. You may have entered the command line incorrectly.

User Action: For a description of the valid command syntax, see Section 3.5. Then, reenter the command line correctly.

com—Bad modify value

Explanation: You specified an incorrect value as the argument to a command qualifier.

User Action: For a description of valid command qualifier values, see Section 3.5. Then, reenter the command line correctly.

com—Bad processor name

Explanation: The processor name includes invalid characters or too many characters.

User Action: Reenter the command with a valid processor name. Depending on the type of processor, valid names are as follows:

- Print processors must be named after the device they control (for example, the processor LP2 controls the printer LP2:).
- Batch processors must have names in the form BAPn.
- Applications processors can have any name consisting of alphanumeric characters.
- The maximum number of characters allowed in a processor name is six.

com—Bad queue name

Explanation: You included invalid or too many characters in the queue name.

User Action: Specify a queue name with no more than six alphanumeric characters.

com—Bad spool device type

Explanation: You attempted to set a device spooled that is not recognized by QMG (for example, a disk device).

User Action: Select a device that can be set spooled (for example, a line printer or a terminal). If you have associated the processor with an invalid device (such as a disk), use the DCL command REMOVE to remove the processor task. Then, reinstall it with a valid output device type.

com—Cannot delete device's queue

Explanation: You attempted to delete a queue before you deleted the processor attached to it.

User Action: First, delete the processor; then, delete the queue.

com—Directive failure

Explanation: There is an unexpected system error.

User Action: Rebootstrap the system and reenter the command line. If the error persists, either the QMG software is corrupt or there is a problem with the hardware. Submit a Software Performance Report (SPR) to DIGITAL.

com—File entry not in job

Explanation: You tried to delete or modify a nonexistent file in a job.

User Action: Type the SHOW QUEUE command to determine the correct file entry number. Reenter the command line with the correct entry number.

com—Illegal argument value

Explanation: You specified an illegal argument for a qualifier in the command line.

User Action: For a description of valid command qualifier values, see Section 3.5. Then, reenter the command line correctly.

com—Illegal or contradictory qualifier

Explanation: You specified an incorrect qualifier in the command line.

User Action: For a description of valid command qualifiers, see Section 3.5. Then, reenter the command line with the correct qualifier.

com—Illegal function code

Explanation: An unexpected system error or a user-written applications processor attempted to send an invalid function to QMG.

User Action: Reenter the command line. If the error persists, check the applications processor for coding errors.

com—Illegal or nonexistent intermediate device

Explanation: You assigned the pseudo device name for the spooling device SP: to an invalid device (such as a terminal or printer) or a nonexistent device.

User Action: Assign the pseudo device name SP: to a valid disk device that is known to the system (such as LB:).

com—Intermediate device not mounted

Explanation: The spooling device SP: is not mounted. You cannot use a device that is not known to the system.

User Action: Mount the SP: device; then, reenter the command line.

com—Invalid entry number

Explanation: You specified an entry number that was either too long or nonnumeric.

User Action: Enter the SHOW QUEUE command to find the correct entry number; then, reenter the command line with the correct number.

com—Invalid file spec syntax

Explanation: You entered the file specification incorrectly.

User Action: Reenter the command line with the correct file specification syntax.

com—Invalid job syntax

Explanation: You specified a job name incorrectly or with too many characters.

User Action: Reenter the command with the correct job name.

com—I/O error
I/O Error code nn

Explanation: This indicates a system error.

User Action: Reenter the command line. If the error persists, check the I/O error code number that accompanies this message. This code number is explained in the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual* and the *Micro/RSX User's Guide, Volume 2*. Take the action suggested by the error code.

com—Job already exists

Explanation: This indicates a system error or a user-written applications processor problem.

User Action: Reenter the command line. If the error persists, check the applications processor for coding errors.

com—Job does not exist

Explanation: You specified a job that does not exist. You may have specified the job name or directory incorrectly.

User Action: Type the SHOW QUEUE command. Determine the name and status of the job. Check for the proper command syntax and retype the command.

com—Job name not allowed

Explanation: You specified a name or an entry number with a command that does not use either qualifier.

User Action: Reenter the command line without specifying a job name.

com—Message not sent

Explanation: This indicates a system error. The system may lack sufficient resources to handle your request.

User Action: Wait for a few minutes; then, reenter the command.

com—Multiple functions detected

Explanation: You specified incompatible functions in a single command line.

User Action: Enter the functions in separate command lines.

com—Mutually exclusive switch/value

Explanation: You entered two or more conflicting qualifiers in the same command line.

User Action: For a description of valid command qualifiers, see Section 3.5. Then, reenter the command line with the correct combination of qualifiers.

com—No file name given

Explanation: You did not specify a file name in the command line.

User Action: Reenter the command line and include a file name. If you are unsure as to the placement of the file name in the command line, see Section 3.5 for the correct syntax.

com—No pool space

Explanation: The system does not have enough pool space available to execute the specified command.

User Action: Wait for tasks to exit before you reenter the command.

com—No such command

Explanation: You specified a nonexistent command.

User Action: See Section 3.5 for the correct command syntax.

com—No switches allowed

Explanation: You specified a qualifier with a command that does not accept qualifiers.

User Action: Reenter the command without any qualifiers.

com—Operation inconsistent with job state

Explanation: A command attempted an action inconsistent with the job state. For example, you attempted to abort a job that was not active.

User Action: Enter the SHOW QUEUE command to determine the status of the job. Also, check the command syntax. Reenter the command with the correct job name and syntax.

com—Privileged command

Explanation: You attempted to execute a privileged command from a nonprivileged terminal.

User Action: Enter this command from a privileged terminal only. If you have a privileged account, log in to the system with the name and password for that account; then, reenter the command.

com—Processor already exists

Explanation: You attempted to initialize a processor that is already initialized. You can initialize only one processor for each device that you want to set spooled; each processor is named after the device it controls.

User Action: To change the attributes for the existing processor, delete the processor (using the DELETE/PROCESSOR command). Then, reinitialize the processor with the INITIALIZE/PROCESSOR command and the appropriate set of qualifiers.

com—Processor directory full

Explanation: You attempted to create more than 16 processors; the maximum allowed by the Queue Manager is 16.

User Action: If you want to change the attributes for an existing processor, delete the processor (using the DELETE/PROCESSOR command). Then, reinitialize the processor with the INITIALIZE/PROCESSOR command and the appropriate set of qualifiers.

com—Processor does not exist

Explanation: You specified a device that is not set spooled.

User Action: Reenter the command with the proper device name.

com—Processor marked for delete

Explanation: The processor has been deleted by the DELETE/PROCESSOR/ERASE command and will exit after the current job is finished.

User Action: Wait for the processor to exit.

com—Processor task not installed

Explanation: You did not install the printer processor task (LPP.TSK) with the name of the device that you want to set spooled.

User Action: Use the MCR or DCL command INSTALL to install the task LPP.TSK with the name of the task that you want to set spooled.

com—QMG did not start

Explanation: QMG did not start when you issued the START/QUEUE/MANAGER command.

User Action: Check to be sure the directory LB:[1,7] exists. If not, create that directory; then, reenter the command.

com—QMG error

Explanation: QMG detected an error within itself or the system.

User Action: Rebootstrap the system and reenter the command. If the error persists, either the QMG software is corrupt or there is a problem with the hardware. Contact DIGITAL Field Service or submit a DIGITAL Software Performance Report (SPR).

com—QMG marked for exit

Explanation: QMG has been stopped and is marked to exit after the current job is finished.

User Action: No user action is required. However, if you want to restart QMG, wait until it exits completely and then restart it.

com—QMG not active

Explanation: The Queue Manager task QMG.TSK is not active; it is stopped, aborted, dormant, or not installed.

User Action: Wait for a few minutes; then, reenter the command. If QMG is stopped, restart it using the START/QUEUE/MANAGER command. If it has been aborted, use the DCL or MCR command INSTALL to reinstall it, as follows:

```
DCL>

MCR>
```

com—QMG not installed

Explanation: The task QMG.TSK is not installed in the system.

User Action: Use the INSTALL command to reinstall QMG.TSK as a checkpointable task (refer to the previous message).

com—Queue already exists

Explanation: You tried to create a queue that already exists.

User Action: To create an additional queue, reenter the command with a unique queue name.

com—Queue and processor not of same type

Explanation: You attempted to associate an incompatible queue and processor (for example, assigning a batch processor to a print queue).

User Action: Assign the processor to a queue of its own type.

com—Queue/device not allowed

Explanation: You specified an invalid queue or device.

User Action: Reenter the command with a valid queue name or device.

com—Queue/device not specified

Explanation: You did not specify a queue name or device in the command line.

User Action: Reenter the command line with a device or queue name.

com—Queue directory full

Explanation: You attempted to create 17 queues, but the maximum number allowed is 16.

User Action: To modify a queue, delete an existing queue (with DELETE/QUEUE) and initialize a new one.

com—Queue does not exist

Explanation: You specified a queue that does not exist in the current system.

User Action: Enter the SHOW QUEUE command to obtain a list of queues that are known to the system. Then, reenter the command with the name of an existing queue or create a queue with that name.

com—Queue file full

Explanation: The queue file [1,7]QUEUE.SYS on the pseudo device LB: (or SP:) is full. QMG has reached the maximum number that it can handle (that is, 999).

User Action: Wait for some jobs to finish; then, reenter the command.

com—Queue marked for delete

Explanation: You attempted to add a new job to a queue marked for deletion.

User Action: Send the job to another queue.

com—Redundant operation

Explanation: You requested an operation (such as deleting a queue) that has already been done.

User Action: No user action is required. This is an informational message.

com—Request failure on processor task

Explanation: You entered an INITIALIZE command, but QMG could not activate the processor due to a system error.

User Action: Reboot the system; then, reenter the command. If the error persists, either the QMG software is corrupt or there is a problem with the hardware. Submit a Software Performance Report (SPR) to DIGITAL or contact DIGITAL Field Service.

com—Spool device does not exist

Explanation: You specified a device to be spooled that does not exist in the system or is not loaded.

User Action: If you entered the wrong device name, reenter the command line with the correct one. If you need to spool to a device that is not currently in the system, load that device into the system. Then, reenter the command.

com—Spool device not available

Explanation: You attempted to spool to a device that is currently in use.

User Action: Wait until that device is available or spool to another device.

com—Spool device offline

Explanation: You attempted to spool to a device that is currently off line.

User Action: Place the device on line; then, reenter the command.

com—Spool device redirected

Explanation: The device to which you want to spool has been redirected to another device.

User Action: Use the REDIRECT command to redirect the spool device back to itself.

com—Spool device unloaded

Explanation: You specified a spool device that is not loaded in the system.

User Action: Load the spool device into the system.

com—Unable to open queue file

Explanation: QMG could not open the queue file.

User Action: Check to see if the queue file LB:[1,7]QUEUE.SYS is locked. If it is, use the UNLOCK command so that the file can be opened.

3.6.2 Error Messages from the Output Despooler

All of these messages indicate a FATAL exit status for the job; the job does not complete. These error messages from the output despooler task appear on the operator's console and in the listing on the output device. When these errors occur, the despooler task prints the job flag pages first and then the error messages on the first page of the listing.

The messages are preceded by the name of the output despooler (indicated by desp in the following list). This name is derived from the name of the output device owned by the despooler, such as LP0 or TT2.

I/O error codes are explained in the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual* and in the *Micro/RSX User's Guide, Volume 2.*

desp—Attempt to space nn pages beyond eof on-filespec

Explanation: The START/PROCESSOR command attempted to start beyond the end of the specified file.

User Action: Reenter the command line with a valid end-of-file (EOF).

desp—I/O Error nn on file- filespec

Explanation: The task encountered an error and returned the I/O error code nn. These codes are explained in the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual* and in the *Micro/RSX User's Guide, Volume 2.*

User Action: Look up the error and take corrective action.

desp—Job limit of nn pages exceeded on file- filespec

Explanation: The PRINT command included the /PAGES:nn qualifier and your job exceeded this limit.

User Action: Shorten the file or issue the PRINT command with a higher limit.

desp—Open error nn on file- filespec

Explanation: The task could not open the file named in the error message for output. The value nn is the I/O error code number. These codes are explained in the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual* and in the *Micro/RSX User's Guide, Volume 2.*

User Action: Check the directory listing for the state of the file. Look up the error; then, take corrective action.

desp—Print error nn on file- filespec

Explanation: The task could not print the file named in the error message. The value nn is the I/O error code number. These codes are explained in the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual* and in the *Micro/RSX User's Guide, Volume 2.*

User Action: Check the directory listing for the state of the file. Look up the error; then, take corrective action.

3.6.3 Error Messages from the Batch Processor

The batch processor (BPR) sends error messages to the system console terminal (CO:) and to individual batch logs. Messages that result from system problems are returned to CO:; these messages are discussed in this section. Errors that reflect difficulties in processing individual batch jobs are returned to the user's batch log. They are listed in the *RSX-11M-PLUS Batch and Queue Operations Manual.*

BPR—Batch job jobname still in progress

Explanation: This message indicates a system error. QMG attempted to start the batch job while another job was in progress.

User Action: Submit an SPR and include the console output as well as any other relevant information. If possible, reproduce the error.

BPR—Error during send to QMG

Explanation: This message indicates a system error.

User Action: Submit an SPR and include the console output as well as any other relevant information. If possible, reproduce the error.

BPR—Illegal error-severity code n

Explanation: This message indicates a system error.

User Action: Submit an SPR and include the console output as well as any other relevant information. If possible, reproduce the error.

BPR—Incorrect Emit Status Block (ESB) address returned by spawned task

Explanation: This message indicates a system error.

User Action: Submit an SPR and include the console output as well as any other relevant information. If possible, reproduce the error.

BPR—Log file open error

Explanation: BPR failed to open the log file.

User Action: Check the log file destination. The device may be write-locked or the specified directory may not exist.

BPR—Log file output error

Explanation: BPR failed to write to the log file.

User Action: Check the log file status. The device may not be available or the file may not be in the directory.

BPR—Spawn failure

Explanation: BPR failed to attach to an active BYE task on a virtual terminal. This message indicates a system error.

User Action: Submit an SPR and include the console output as well as any other relevant information. If possible, reproduce the error.

BPR—Unable to create virtual terminal

Explanation: BPR was unable to create a virtual terminals for one of the following reasons:

- The virtual terminal driver was not loaded.
- On an RSX–11M–PLUS system, support for virtual terminals was not included during system generation.
- When BPR attempted to create a virtual terminal, it exceeded the maximum limit for the number of virtual terminals allowed by the system. (On an RSX–11M–PLUS system, you can set the maximum limit during system generation.)

User Action: If you are using an RSX–11M–PLUS system (excluding RL02 distribution kits), repeat the system generation procedure and include support for virtual terminals or increase the maximum limit. If you are using a pregenerated RSX–11M–PLUS or Micro/RSX system, submit an SPR to DIGITAL.

Chapter 4

System Reconfiguration

Reconfiguration is the process of physically and logically connecting and disconnecting various system resources. If you have an RSX–11M–PLUS operating system (excluding pregenerated systems), you can use the reconfiguration services to add or to remove resources (such as memory and devices) while the system is on line.

Pregenerated RSX–11M–PLUS systems (RL02 distribution kits) and Micro/RSX systems do not support the reconfiguration services. Instead, they support a feature called *Autoconfigure.* When you bootstrap your system, the Autoconfigure task ACFPRE.TSK automatically identifies the devices in your configuration and creates the necessary data structures for those devices.

Section 4.1 describes how to use Autoconfigure to set up or to modify a pregenerated RSX–11M–PLUS or Micro/RSX system. The rest of the chapter discusses the components and commands of the reconfiguration services for RSX–11M–PLUS systems.

Some of the information in this chapter applies only to multiprocessor systems. Blue ink is used to distinguish this information.

4.1 Autoconfigure on Micro/RSX and Pregenerated RSX–11M–PLUS Systems

The first time you bootstrap your Micro/RSX or pregenerated RSX–11M–PLUS system, the system startup procedure STARTUP.CMD executes the Autoconfigure task LB:[3,54]ACFPRE.TSK. ACFPRE.TSK identifies the devices in your peripheral configuration (see Section 4.1.1), reads the configuration file LB:[1,2]SYSPARAM.DAT, and creates data structures that describe the devices. (Depending upon the configuration, this part of the startup procedure may take up to 1 minute to complete.) The startup procedure then loads the appropriate device drivers.

In general, you probably will not need to change your peripheral configuration each time you rebootstrap your system. So, to decrease the time required for system startup, ACFPRE.TSK stores the information that describes your current configuration in the file LB:[1,2]ACFPAR.DAT. For subsequent system startup procedures, STARTUP.CMD uses the information in ACFPAR.DAT (instead of running ACFPRE.TSK again).

If you add a peripheral device to your configuration, delete LB:[1,2]ACFPAR.DAT and rebootstrap the system. To determine and record the configuration information, the system startup procedure will run ACFPRE.TSK again.

4.1.1 Using Non-DIGITAL Standard Configurations

DIGITAL uses certain standards to determine the correct CSR addresses and interrupt vectors for peripheral devices on PDP–11 systems. If your devices are set up according to these standards (your DIGITAL Field Service Representative can verify this), ACFPRE.TSK can identify them. However, because the system allocates device CSR addresses in a fixed order in the I/O page, adding a peripheral may require changing the CSR and vector addresses of existing devices to comply with DIGITAL standards.

If your system does not meet DIGITAL's configuration standards (for example, certain peripheral options do not allow sufficient flexibility to meet the standards), or if you add a peripheral device and do not wish to reconfigure existing peripheral devices, edit ACFPAR.DAT to modify the configuration information accordingly.

4.1.2 Modifying ACFPAR.DAT

To modify the parameter file LB:[1,2]ACFPAR.DAT, edit the file and use the correct record formats. ACFPAR.DAT contains the following types of records:

- A CPU record
- A line frequency record
- Several controller records

The following subsections describe the corresponding formats for each type of record in more detail.

CPU Record

```
CPU=11/xx
```

The variable xx represents the model number of your PDP–11 processor. For example, 23 and 73 are valid model numbers for the MicroPDP–11/23 and MicroPDP–11/73 processors, respectively.

Line Frequency Record

```
LIN=n
```

To set the system clock, ACFPRE.TSK determines your power-line frequency. If necessary, you can edit ACFPAR.DAT to correct the clock frequency.

The variable n can be either 50 or 60 hertz (Hz). If your line frequency is 50 Hz and your system clock seems to lose time, n is set incorrectly to 60 Hz. Conversely, if your line frequency is 60 Hz and your system clock seems to gain time, n is set to 50 Hz. Specify a value for n that is equal to your actual line frequency.

Controller Records

```
CON=cname,CSR=csradd,VEC=vecadd,UNI=(numuni,lowuni)
```

Parameters

cname

Specifies the controller type and identification code. Valid controller types and their respective devices are as follows:

Controller Type	Devices
DU	MSCP-type devices
MU	TK50 and TU81 magnetic tapes
MS	TS11, TSV05, TU80, and TK25 magnetic tapes
DL	RL02 disks
YL	DL11 terminal intertaces
YV	DHU11 and DHV11 multiplexers
YZ	DZ11 and DZV11 multiplexers

The identification code is the sequential indicator of the controller from the DIGITAL hardware alphabet.

csradd

Specifies the control and status register (CSR) address for the device.

vecadd

Specifies the first vector address for the device.

numuni

Specifies the number of available units attached to the controller.

lowuni

Specifies the unit number of the lowest numbered device unit (that is, the first unit number).

The numuni and lowuni parameters are included for DL, DU, and MU device types only.

Example

Adds a record describing an RC25 controller with one master unit (an RC25 master unit consists of two drives) as the only extra peripheral on the system. The unit number plug on the drive indicates that it is units 4 and 5.

4.2 The Reconfiguration Services (RSX-11M-PLUS Systems Only)

The reconfiguration services allow you to bypass failed devices and isolate your system from the effects of faulty hardware. By reconfiguring your system, you can define a set of hardware resources that are accessible from the online system. Later, you can use the reconfiguration services to remove failed devices from the pool of allocated resources.

For example, after bootstrapping the system, you can place a failed disk drive off line. Then, use another drive that is already on line to take over for the disabled unit.

The sections that follow describe the components and basic concepts associated with the system reconfiguration services.

4.2.1 Reconfiguration Components—CON, RD:, and HRC

The reconfiguration services consist of three components: a command interface (CON), a loadable driver (RD:), and a privileged reconfiguration task (HRC). You must have enough space in memory to contain both CON and HRC at the same time; otherwise, CON commands fail.

The three components of the reconfiguration services are defined as follows:

The command interface (CON)

Receives and parses the commands that you enter to reconfigure the current system. Then, CON creates and submits QIO packets to the reconfiguration driver (RD:).

The loadable driver (RD:)

Serves as an interface between a user task (either CON or an application task) and the HRC task; it is the gateway between tasks that generate reconfiguration requests and the actual implementation of those requests. RD: accepts QIO packets from a task and places these packets into the queue for HRC.

Before RD: passes the QIO packets to HRC, it performs privilege and address boundary checks. The checks ensure that the packets are using valid addresses and function codes. RD: performs additional checks on some functions to ensure that it is attached to them.

For more information on RD:, see the reconfiguration sources located in the directory [27,10] of the distribution kit (except the pregenerated RL02 and Micro/RSX distribution kits, which do not contain any sources). Important modules for the interface are CNHRC.MAC, HROOT.MAC, and HRSUB.MAC. Also, see the macro OLRDF$ in [1,1]EXEMC.MLB for a description of volume access rights and mounted volumes.

Note

Programs written to use the current version of RD: may have to be modified in the next release. You should centralize all code concerning RD: so that you can make modifications easily.

The privileged reconfiguration task (HRC)

Performs the actual reconfiguration operations for the system. To process reconfiguration requests, HRC calls Executive routines, which then call the associated device drivers. HRC services the requests for configuration information by passing the desired information directly to the user buffer.

After HRC services the request, a status is returned (by means of I/O completion) to the requesting task. The status return is done to allow tasks other than CON to interface with HRC.

The database for each system device contains status words that reflect device state. When you change a system device state, HRC modifies the appropriate database status word either to grant or to inhibit device access. Section 4.2.2 discusses system device states in more detail.

4.2.2 System Device States

The system reconfiguration services control the configuration state (on line or off line) of the hardware devices. Therefore, they also control device access. A system device can assume one of the following four states:

Online

Indicates the device is on line; access is granted.

In the online state, a device is *logically on line* for access. In other words, an access path exists between the device and an online processor. A call to the driver's controller or to the entry points of the unit status is successful.

The system grants access to online devices only. To access an online disk or tape volume and to perform virtual I/O (for example, read or write operations to files), the volume must also be mounted. (See the *RSX-11M-PLUS MCR Operations Manual,* the *RSX-11M-PLUS Command Language Manual,* or the *Micro/RSX User's Guide, Volume 2* for a description of volume access rights and mounted volumes.)

Offline

Indicates the device is off line; access is denied.

When a device is in the offline state, the system denies access to the device. Attempts to access an offline device result in an I/O error (IE.OFL). (See the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual* for lists and explanations of error codes.) Taking a controller off line also takes all of its associated units off line, which places the units in the *marked for online* state.

Marked for online

Indicates the device will be brought on line when an access path exists; access is denied.

The marked for online state indicates that a request to bring a device or controller on line has been made, but part of the access path to the device or controller is off line. Access attempts yield the error code IE.OFL. HRC places a device marked for online state logically on line when it brings the entire access path on line.

Marked for offline

Indicates the device is in an indeterminate state; access is denied.

The marked for offline state indicates that the system encountered an unexpected error while placing a specific device logically on line. Before you attempt or retry an online operation, issue an explicit OFFLINE command.

4.2.3 Placing Devices Logically On Line and Off Line

Before HRC places a device logically on line, it verifies that the device driver is loaded. Then, it ensures that the controller is online by testing access to the device. If the controller can access the device (that is, if an access path exists), HRC modifies the appropriate status word to reflect that the device is on line. If the controller or device cannot be accessed, HRC marks the device for online status. Then, when the access path is brought on line, HRC places the device on line.

Before HRC places a device off line, it attempts to protect against loss or corruption of data by verifying that the change in state (from online to offline) will not interfere with active I/O. HRC considers a device active and rejects offline commands if the device has *context*. The reconfiguration services define *device context* as a condition that indicates (or permits) I/O activity on the device. The following is a list of conditions that define device context in more detail:

- A task is attached to the device (for example, the print despooler attaches the line printer).
- The device (or an associated device) is a logged-in terminal.
- The device (or an associated device) contains a mounted volume.
- For the DT07 bus switch, a port is logically connected (or *linked*) to a processor.
- For memory boxes, the memory within the box is logically on line (multiprocessor systems only).
- A disk device has data caching enabled.

You cannot take a controller off line if any of the units on the controller have context. (Dual-port disks that support load sharing are an exception, because they they can switch paths dynamically. Consequently, you can place one path with context off line as long as the alternate path is on line. For more information on load sharing, see the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver.*)

4.3 Invoking the Command Interface

You can reconfigure an RSX-11M-PLUS system interactively from a terminal or by using indirect command files. The reconfiguration services act as an interface between your terminal and the system. To use the reconfiguration services, invoke the command interface CON and then issue the commands.

There are two ways to invoke CON interactively:

1. Use the following *one-line format* to invoke CON, to execute a single command, and to return control to the Monitor Console Routine (MCR):

   ```
   MCR>
   ```

2. Use the following *direct task format* to invoke CON and obtain a task prompt (CON>):

   ```
   MCR>
   CON>
   ```

   Press CTRL/Z to exit from the direct task format.

You can also enter commands through indirect command files. An indirect command file that contains reconfiguration commands does not require terminal interaction when the file is executed. Thus, indirect command files are a convenient means of creating configuration command sequences that create different configurations or restore previous configurations.

Indirect command files can be nested to a maximum depth of three.

Examples

```
MCR>
```

Invokes CON, executes the reconfiguration commands contained in the indirect command file CONFIG.CMD, and returns control to MCR.

```
CON>
```

Executes the commands in the file CONFIG.CMD and returns control to CON.

4.4 Entering CON Commands

To enter a CON command, type a command line in response to the default prompt (>) or to the explicit command interface prompt (CON>). A command line consists of three components: a command name, required parameters, and a line terminator.

It is not necessary to use the entire command name when you enter a command line; CON requires only the first three characters of the name. However, to make the intent of the command input clear to someone reading a printed copy, type the command name in full. For example, CON accepts either of the following command lines:

```
CON>
```

```
CON>
```

In the previous example, dev-spec is the target device specification.

A space or tab signals the end of the command name. Therefore, no spaces or tabs are permitted within the name itself.

4.5 CON Command Descriptions

This section describes each of the reconfiguration commands in alphabetical order. Note that there are no equivalent DIGITAL Command Language (DCL) commands for accessing CON. Also, some of the commands must be issued from a privileged terminal.

Table 4–1 briefly summarizes the reconfiguration commands.

Table 4–1: CON Command Summary

Command	Function
BUILD	Creates a command sequence in an internal buffer that, if executed, duplicates the current system configuration.
CLEAR	Erases the command sequence created with the BUILD command (but not the file created with the LIST command) that is stored in the internal buffer by CON.
DISPLAY	Displays the configuration and status of the hardware devices in the current system.
ESTATUS	Emits the current status of a specific device.
HELP	Displays the help text for CON.
IDENT	Displays the current versions of CON and HRC and the date and time the two tasks were built.
LINK	(Multiprocessor systems only.) Logically connects a port of a switched bus run to a processor.
LIST	Displays the results from a BUILD command or puts the command sequence into a specified file. A file type of CMD is recommended because that is the default type for an indirect command file.
OFFLINE	Removes a device without context from the active set of devices in the current configuration.
OFFLINE MEMORY	(Multiprocessor systems only.) Removes the MKA11 memory box with the highest addresses from the system. Note that you must first ensure that tasks are not using the memory in the memory box.
ONLINE	Attempts to place a device logically on line.
ONLINE MEMORY	(Multiprocessor systems only.) Adds an MKA11 memory box to the online configuration. Note that you must expand the partition structure to use the additional memory.
SET	Alters the control and status register (CSR) or interrupt vector address values of a device (and, on multiprocessor systems, activates or inhibits the sanity timer and its alarm.)
SWITCH	(Multiprocessor systems only.) Logically disconnects a port of a switched bus run from a processor and then connects the port to another processor. The command is equivalent to a LINK command followed by an UNLINK command.
UNLINK	(Multiprocessor systems only.) Logically disconnects a port of a switched bus run from its respective processor.

CON also accepts two switches, /HE and /NOMSG. These switches are used alone in the CON command line. Their functions are as follows:

Switch	Function
/HE	Displays the help text for CON.
/NOMSG	Suppresses all information displays and error messages returned by CON. You must exit from and then reenter CON to have the messages displayed again.

4.5.1 BUILD

The BUILD command creates a sequence of commands that, when executed, duplicates the current system configuration. The sequence of reconfiguration commands that result from a BUILD command is held in a buffer in memory. When this sequence of commands executes, the reconfiguration commands restore the system to the exact state that existed when you entered the BUILD command.

Format

```
CON>BUI[LD]
```

Example

```
CON>
```

Creates a sequence of commands that duplicates the current system configuration. To examine the sequence of commands that is created, use the LIST command (see Section 4.5.8).

4.5.2 CLEAR

The CLEAR command erases from the internal buffer the current sequence of commands created by the BUILD command. The command does not delete the file created by the LIST command (see Section 4.5.8).

Format

```
CON>CLEAR
```

Example

```
CON>
CON>
```

Removes from the internal buffer the command sequence that resulted from the previous BUILD command.

4.5.3 DISPLAY

The DISPLAY command provides current system configuration information about the device units, device controllers, processors, memory boxes, and memory box controllers.

Format

```
CON>DIS[PLAY] [keyword(s)] [FOR string]
```

Keywords

ALL
ATTRIBUTES
CONTROLLERS
FULL
UNITS

Parameters

keyword(s)

Specifies the type of configuration information that is to be displayed.

The DISPLAY command allows more than one keyword per command line. However, some combinations (such as DISPLAY UNITS FULL FOR DB) are meaningless. The DISPLAY command ignores inappropriate keywords.

If the command omits a keyword, the display contains the logical names and status flags for every device unit and device controller in the current configuration.

FOR string

Allows a selective display of a particular device or device group. When you specify this parameter, CON displays the requested information for the devices specified in the string.

You can also use one or two wildcard characters (*) in the string. Use one wildcard to find out which devices are connected to the controller you have specified after the wildcard. Use two wildcards to find out which devices have the attribute you have specified after the wildcards (either ONLINE or OFFLINE).

The FOR string is an optional parameter.

Keyword Descriptions

ALL

Displays the same information as the FULL keyword with the addition of interrupt vector addresses and CSR assignments for device controllers.

ATTRIBUTES

Displays interrupt vector addresses and CSR assignments for device controllers.

CONTROLLERS

Displays every device controller in the current configuration.

FULL

Displays the status flags for every device unit and device controller in the current configuration. The status flags indicate the following:

Status Flag	Meaning
Accpath	The device has an online access path.
Context	For a tape or disk (mountable device), the device contains a mounted volume. For a terminal, a user is logged in. Other devices are attached.
Driver	For a device unit, the driver is loaded. For a device controller, a driver is available.
Massbus	The device is connected to an RH-type MASSBUS controller.
Multiport	The device is dual-ported (a dual-access device) between two controllers (for example, a dual-ported RP06 drive).
Offline	The device is off line.
Online	The device is on line (or marked for on line).
Prv_Diag	(Used with the OFFLINE and ONLINE keywords.) The device is marked for on line or off line, but before any action is taken, the corresponding controller must be brought on line or taken off line.
Unknown	The CSR assignment for the device is 160000_8 and thus cannot be accessed.

UNITS

Displays every device unit in the current configuration. The display equates each device unit with its respective controller.

Examples

```
CON>
DB0:                     Offline,Multiport,Massbus,Accpath,Driver
 PORT DB0A RHB0:         Online
 PORT DB0B RHC0:         Online,Current,Accpath
DB1:                     Offline,Multiport,Massbus,Accpath,Driver
 PORT DB1A RHB1:         Online
 PORT DB1B RHC1:         Online,Current,Accpath
```

Displays information about all of the DB-type devices configured into the system.

DISPLAY

```
CON>
DB0:        RHB0:,RHC0:
DB1:        RHB1:,RHC1:
DR0:        RHA0:
DR1:        RHA1:
MM0:        RHD00:
MM1:        RHD11:
CO0:
TT0:        YLA0:
TT1:        YZA0:
TT2:        YZA1:
TT3:        YZA2:
VT0:
DK0:        DKA0:
DK1:        DKA1:
DL0:        DLA0:
LP0:        LPA0:
LP1:        LPB0:
LP2:        LPC0:
NL0:
DM0:        DMA0:
DM1:        DMA1:
```

Displays all of the device units and controllers configured into the system.

```
CON>
```

Displays all of the RH-type controllers.

```
CON>
```

Checks to see which devices have the ONLINE attribute (port-specific status is not displayed).

4.5.4 ESTATUS

The ESTATUS command causes CON to exit with its exit status word reflecting the current state of a specified device. If you use this command with the Indirect Command Processor (Indirect; see the *RSX-11M-PLUS Indirect Command Processor Manual*), the command returns a 16-bit word describing device status. You can use the status word to obtain information about the configuration. Table 4–2 lists the values for the bits in the word for units and controllers; Table 4–3 lists the values for ports.

Table 4–2: Exit Status Values for Units and Controllers

Bit	Bit		
No.	**Value**	**Description**	**Display**
0	1	Exit status value:	
1	2	0 = Warning	
2	4	1 = Success	
		2 = Error	
		4 = Severe error	
6	100	Subcontroller device	
7	200	Controller device	
8	400	State:	
		0 = Device online	Online
		1 = Device offline	Offline
9	1000	Device is restricted to privileged and diagnostic functions.	Prv_Diag
10	2000	Multiport device	Multiport
11	4000	MASSBUS device	Massbus
12	10000	Device's CSR is 160000_8 (unknown)	Unknown
13	20000	Device has an online access path	Accpath
14	40000	Device has context	Context
15	100000	Driver is loaded for this device	Driver

Table 4–3: Exit Status Values for Ports

Bit No.	Bit Value	Description	Display
0	1	Exit status value:	
1	2	0 = Warning	
2	4	1 = Success	
		2 = Error	
		4 = Severe error	
8	400	State:	
		0 = Port online	Online
		1 = Port offline	Offline
10	2000	Current port	Current
13	20000	An online access path to this port exists	Accpath
14	40000	Port or the device it services has context	Context

Example

```
>                             ! Invoke Indirect.
AT.>                          ! Enable substitution mode.
AT.>                          ! Type CON command line.
>CON ESTAT DB1:               ! System displays command line.
AT.>                          ! Requests display of <EXSTAT>.
>;126401                      ! System displays current <EXSTAT> contents.
AT.>                          ! Exit Indirect
AT.> ^Z
>@ <EOF>
>
```

Presents a session where the user has used Indirect interactively. The CON command line displays the current status of DB1, which is stored in the Indirect special symbol <EXSTAT> .

The number 126401 has the following meaning:

100000	Device's driver loaded.
20000	Device has online access path.
6000	Device is both multiport and MASSBUS.
400	Device is off line.
1	Exit status is Success.

4.5.5 HELP

The HELP command displays the help text for CON.

Format

```
CON>HE[LP]
```

Example

```
CON>

CON -- Configuration Control Command Task
Implemented commands:

ONLINE <device_list>
ONLINE MEMORY <memory_box>
ONLINE ALL
OFFLINE <device_list>
OFFLINE MEMORY <memory_box>
OFFLINE ALL
SET <controller_sp>
```

Shows the form in which the HELP command displays information about the system configuration.

4.5.6 IDENT

The IDENT command displays the current version numbers of the CON and HRC tasks. The date and time the tasks were task built are also displayed.

Format

```
CON>IDENT
```

Example

```
CON>

CON -- CONFIGURE Version 4.00
Task built: 1-MAR-87 12:52:43.5

HRC -- HRC Version 004.000
Task built: 12-MAR-87 23:59:10.9
```

In this example, Version 4.0 of CONFIGURE was task built on March 1, 1987; Version 4.0 of HRC was task built on March 12, 1987.

4.5.7 LINK

The LINK command logically connects a port of a switched bus run to a processor and thus permits access to the devices on the switched segment of the bus run.

To logically connect a switched bus run to the processor's UNIBUS, the devices on the switched segment must all be off line or marked for on line. You cannot dynamically link switched bus runs.

The LINK command is supported by multiprocessor systems only.

Format

```
CON>LINK UBx TO CPx
```

Parameters

UBx

Specifies the target-switched bus run. Switched bus runs are named using the designation UB followed by a letter from the DIGITAL hardware alphabet as follows: UBE, UBF, UBH, UBJ, UBK, UBL, UBM, UBN, UBP, UBR, UBS, and UBT.

CPx

Specifies the target processor (A, B, C, or D).

Example

```
CON>
```

Logically connects the first switched bus run (UBE) to processor A (CPA).

4.5.8 LIST

The LIST command displays the current command sequence or writes the sequence to a file. You can use this command to keep a list of the commands that will duplicate the current configuration. If the buffer in which the current command sequence is stored is empty (see Section 4.5.1 for the description of the BUILD command), CON returns the following message:

```
CON -- Command list is empty. Nothing to print
```

If you specify the file type CMD, you can execute the commands from the indirect command file and then return to CON. For example, to produce the list of commands that will duplicate the current configuration, enter the following commands:

```
CON>
CON>
```

Then, when you want to use the file to duplicate your configuration, execute it by typing the following:

```
CON>
CON>
```

Format

```
CON>LIST [filespec]
```

Parameter

filespec

Specifies the name and location of the file to contain the command listing. The valid format for the file specification is as follows:

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

(For more information on file specifications, see the *RSX-11M-PLUS MCR Operations Manual.*)

Example

```
CON>
CON>
ONLINE  RHA
ONLINE  RHB
ONLINE  RHC
OFFLINE DMA
OFFLINE YLA
ONLINE  YZA
ONLINE  DKA
ONLINE  DLA
ONLINE  LPA
ONLINE  DB0:
ONLINE  DB1:
ONLINE  DR0:
ONLINE  DR1:
ONLINE  MM0:
ONLINE  MM1:
ONLINE  CO0:
ONLINE  TT0:
ONLINE  TT1:
ONLINE  TT2:
    .
    .
    .
ONLINE  TT10:
ONLINE  DK0:
ONLINE  DK1:
ONLINE  DL0:
ONLINE  LP0:
ONLINE  NL0:
ONLINE  DM0:
ONLINE  DM1:
```

Creates a sequence of commands that duplicates the current system configuration. Next, the LIST command displays the sequence of commands you would use to reestablish the current system configuration (that is, the sequence of commands created by BUILD).

4.5.9 OFFLINE

The OFFLINE command changes the state of an inactive device to offline, which removes the device from the active set of resources in the current system configuration. This prevents further device access. Before attempting to take a device off line, be sure there is no I/O activity on the device (see Section 4.2.3). HRC must be able, in 1000_{10} tries or less, to verify that there is no I/O activity in progress.

The transition to the offline state for a specific device controller is completed only if all associated devices are also off line. If an associated device is not off line, HRC rejects the OFFLINE command because placing the device off line could interfere with active I/O (the device has context).

Before placing a device off line, be sure there are no conditions present that constitute device context. To remove device context, enter the appropriate DCL or MCR commands (for example, STOP a print processor, LOGOUT a terminal, or DISMOUNT the volume).

If a task has attached the device unit, the task must be detached (or request to be aborted) before the unit can be placed off line. For example, if you issue an OFFLINE command for the line printer controller LPB and the connected unit LP1: has context, CON rejects the command and issues the following error message:

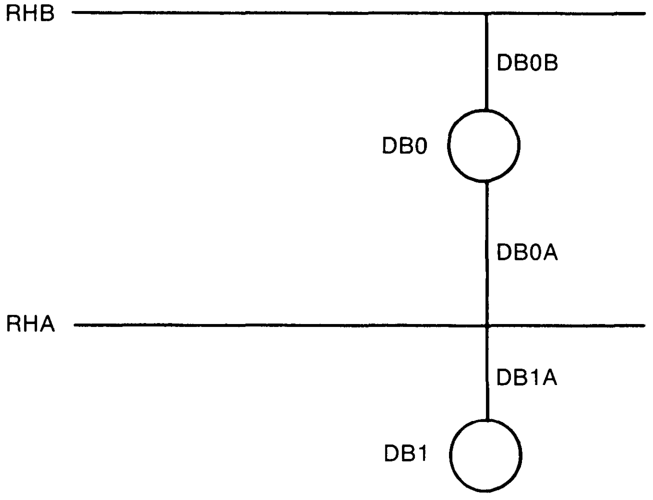
```
CON -- Unable to take unit with context offline
```

If the device is in the marked for online state (for example, a device unit with its controller off line), the command changes the device state to offline. Subsequently, when the controller is brought on line, the system does not activate the device.

If the device is marked for offline, the command changes the device state to offline and thus enables a subsequent online attempt.

Individual ports of dual-port devices can be taken off line separately. You may want to do this if the port does not work correctly (for example, you cannot access the device through the port), or if you want to increase system throughput. Figure 4–1 is an example of how to increase throughput.

Figure 4–1: Sample Configuration for Increasing System Throughput



ZK-575-81

With this configuration, the system uses the DB0A port. Because activity on DB1 must use the DB1A port, it has to wait until DB0A is free. If you place only ports DB0B and DB1A on line, each DB disk has its own controller and does not have to wait for another port to be free.

Formats

```
CON>OFF[LINE] device-spec1[,device-spec2...,device-specn]
```

```
CON>OFF[LINE] ALL
```

Parameters

device-spec

Represents the target device specification. Note that device-spec also includes processors (CPx).

ALL

Places all devices off line except the system disk and the terminal that issued the command. Before you enter the OFFLINE ALL command, dismount all volumes (tapes and disks) except the system disk and log out all terminals except the issuing terminal.

The OFFLINE ALL command is useful prior to saving a system image with the SAVE command.

OFFLINE

Requirements

Before you enter the OFFLINE command, be sure the following conditions are met:

- Only a privileged user can modify the system configuration.
- No tasks are attached to the device.
- No volumes are mounted on the device.
- If the device is a terminal, no users are logged in on it.

Example

```
CON>
CON>
```

Places the devices logically off line, which allows the bus run UBE to be placed off line. This example assumes that no terminals are logged in to the system.

4.5.10 OFFLINE MEMORY

The OFFLINE MEMORY command removes a memory box from the system configuration. Because memory must be contiguous, the target box must be the memory box in the system configuration with the highest addresses (the last memory box added to the system). If the target box is not the last box in the memory configuration, CON rejects the command and issues the following message:

```
HRC... Can't take box offline. Not last box in memory
```

If a partition maps to a memory box targeted for offline status, CON rejects the command and issues the following message:

```
HRC... Can't take box offline. Partition overmaps box
```

Therefore, you must reduce the partition structure before removing the memory box. The DCL command SET PARTITION/TOP (MCR command SET /TOP) can adjust the partition structure up or down (see the *RSX-11M-PLUS Command Language Manual* or the *RSX-11M-PLUS MCR Operations Manual*), but it is your responsibility to ensure that no one is using the memory in the memory box. The system is protected so that you cannot bring the memory box off line if it is in use.

The OFFLINE MEMORY command is supported by multiprocessor systems only.

Format

```
CON>OFF[LINE] MEM[ORY] memory-box
```

Example

```
CON>
```

Removes memory box MK2 from the system configuration.

4.5.11 ONLINE

The ONLINE command changes the state of a device to on line and enables device access.

The transition to the online state is completed only if an access path exists between the device and the processor, and if the device driver can successfully initialize the device. If the device lacks an access path, the ONLINE command changes the device state to the marked for online state; the device is activated when an online access path becomes available (when the controller is brought on line).

Note that the database values for interrupt vector and control and status register (CSR) assignments have no effect on the system until the device is brought on line.

The ONLINE command activates a device unit only if the associated controller is on line. The command first tests the unit to determine whether it actually exists on the controller and then performs drive sizing to determine device unit type. If the unit exists, it is now ready for user access. If the associated controller is off line, the command marks the unit for online status; it is presence-tested and sized when the access path is completed and, if present, activated.

To activate a device controller, the ONLINE command tests the CSR address to determine whether the controller is physically present. If the controller is present, then all of the associated units that are marked for online status are presence-tested, sized, and brought on line. Offline units remain in the relative offline state regardless of any ONLINE command performed on the associated controller.

To activate a processor on multiprocessor systems, the ONLINE command issues a bootstrap request to the target processor. If the target processor ignores the bootstrap request or cannot boot successfully, CON issues the following message (after a timeout of 5 seconds):

```
**Command execution error. Device offline
```

Formats

```
CON>ONL[INE] device-spec1[,device-spec2...,device-specn]
```

```
CON>ONL[INE] ALL
```

Parameters

device-spec

Represents the target device specification. Note that device-spec also includes processors (CPx).

ALL

Places every device controller and device unit known to the system logically on line.

Note

Placing all devices on line takes several seconds on a large system. Also, the command does not generate any error messages to indicate that a device may not be on line.

ONLINE

Restrictions

- Only a privileged user can modify the system configuration.
- The specified device must be physically present in the target system and included in the system at system generation.
- If the specified device is serviced by a loadable driver, the driver and device database must be loaded.

Notes

1. Individual ports of dual-port devices can be placed on line separately. You may want to do this if the port does not work correctly (for example, you cannot access the device through the port) or if you want to increase system throughput.
2. For dual-port terminals, you need a Status Control Block (SCB) for each terminal to bring the ports on line individually and to switch them individually. To bring the ports on line, the ONLINE command places the controller on line. However, you must switch the terminals manually by using the CS11 switch on each terminal.
3. If you have only one SCB for each controller, all of the dual-port terminals have to be switched over to the same controller.

Examples

```
CON>
CON>
CON>
CON>
CON>
CON>
```

Places the controller LPA and associated printer LP0: on line. Then, places the terminal interfaces YZA and YZB on line. Last, places eight terminals on line for each controller (a total of 16).

```
CON>
```

Places every controller and device unit known to the system, except central processing units (CPUs), logically on line (that is, all controllers and devices included at system generation). If a device unit or device controller lacks an access path, the command marks these for online status.

4.5.12 ONLINE MEMORY

The ONLINE MEMORY command adds a memory box to the system configuration. To place the additional memory on line, you must first place the appropriate memory box port controllers and the memory box itself logically on line. The box you specify will have the highest memory addresses.

To use the additional memory, expand the partition structure with the DCL command SET PARTITION /TOP (MCR command SET /TOP). For a description of these commands, see the *RSX-11M-PLUS Command Language Manual* or the *RSX-11M-PLUS MCR Operations Manual.*

The ONLINE MEMORY command is supported by multiprocessor systems only.

Format

```
CON>ONLINE MEMORY memory-box
```

Example

```
CON>
```

Adds the memory box MK2 to the system configuration.

4.5.13 SET

The SET command changes the value of the CSR or the interrupt vector address for an offline device. The values are stored in the device driver's database.

SET does not verify the CSR or interrupt vector address that you specify until you attempt to place the controller on line. Therefore, if you specify an incorrect CSR address, the system cannot access the device and the ONLINE command fails. Likewise, if you specify an incorrect interrupt vector address and then place the device logically on line, the system sends interrupts to nonexistent interrupt vectors; the device is not accessed. Also, if you specify an interrupt vector that is in use and then attempt to place the controller on line, CON returns an error message. (See the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver* for descriptions of CSR and vector assignment errors.)

On multiprocessor systems, the SET command also activates the hardware sanity timer and enables an alarm if the timer expires.

Format

```
CON>SET controller option=value
```

Parameters

controller

Specifies the device controller (for example, RHA or DMA).

option

Specifies one of the following options:

CSR — Alters the value of the controller's CSR address stored in the driver database.

VEC — Alters the value of the controller's interrupt vector address stored in the driver database.

TIMER — Activates or inhibits the sanity timer for the CPx controller. (The TIMER option is supported by multiprocessor systems only.)

ALARM — Activates or inhibits an alarm upon sanity timer timeout for the CPx controller. (The ALARM option is supported by multiprocessor systems only.)

value

Depends upon the option you select, as follows:

- If you select the CSR option, specify a value greater than or equal to 160000_8 and less than or equal to 177777_8. If you specify a CSR address of 160000_8 or less, the device is not known to the system and cannot be placed on line. By modifying the CSR addresses, you prevent the system from knowing about devices that should not be brought on line. This is particularly useful if your current hardware configuration differs from the generated system.

  For example, if your target system has a DMC11 controller with a CSR address of 160100_8 and the generated system has a DZ11 controller with the same address, the system will fail when the CON ONLINE ALL command attempts to bring the DMC11 on line as a DZ11. However, if you set the DMC11 controller's CSR address to 160000_8

or less, you prevent the CON ONLINE ALL command from bringing that controller on line, which also prevents a system failure.

- If you select the VEC option, the value must be less than 774_8.
- If you select the TIMER or ALARM option, the value is either ON (to activate) or OFF (to inhibit).

  Note that these options apply to multiprocessor systems only.

Requirement

The device must be off line and the driver must be loaded.

Examples

```
CON>
```

Changes the interrupt vector address for the first line printer controller to 160_8.

```
CON>
```

Changes the CSR address for the first line printer controller to 177600_8.

4.5.14 SWITCH

The SWITCH command modifies the processor association of a switched bus run by performing an unlink operation followed by a link operation. The command uses a special hardware feature of the DT07 to guarantee that the switched bus is linked to the target processor after the unlink operation. Thus, the SWITCH command ensures that the switched bus will be logically connected to the desired processor.

For example, in a three-processor system, a properly configured switched bus can be connected to any one of three processors. An UNLINK command permits any processor to then link the switched bus. However, the SWITCH command specifies that only a specific processor can link the switched bus run following the unlink operation. If CPA and CPB are running as a dual-processor system and CPC is a single-processor system, a switch from CPA to CPB occurs even if CPC is trying to gain control of the bus.

CON rejects the command if a device attached to the switched bus has context (that is, the device is processing or allowing processing of I/O).

The SWITCH command is supported by multiprocessor systems only.

Format

```
CON>SWITCH UBx TO CPx
```

Parameters

UBx

Specifies the target-switched bus run. Switched bus runs are named by using the designation UB followed by a letter from the DIGITAL hardware alphabet (E, F, H, J, K, L, M, N, P, R, S, and T).

CPx

Specifies the target processor.

Example

```
CON>
CON>
CON>
CON>
CON>
CON>
CON>
CON>
CON>
```

Logically disconnects the first switched bus run (UBE) from its respective processor and then logically connects UBE to processor B.

4.5.15 UNLINK

The UNLINK command disconnects a port of a switched bus run from its respective processor. This command reverses the effect of the LINK command.

CON rejects the command if any device attached to the switched bus has *context* (that is, the device is processing or allowing processing of I/O).

Format

```
CON>UNLINK UBx
```

Parameter

UBx

Specifies the switched bus.

Example

```
CON>                    ! Terminals assumed not logged in
```

Disconnects the switched bus port of UBE from its respective processor.

4.6 Error Messages

The CON and HRC error messages consist of one or more lines of information in the following format:

```
CON -- error message
(detail of error message)
```

The first line is the general error message. The second and any succeeding lines define the error message.

In addition to the description, the error messages may include Directive Status Word (DSW) error codes and I/O error codes. For explanations of these codes, refer to the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual* or the *Micro/RSX User's Guide, Volume 1*.

A list of the CON and HRC error messages and a brief explanation of each follows.

4.6.1 CON Error Messages

The following messages are returned by the CON interface:

CON—Bad indirect command file specification

Explanation: You specified the name of an indirect command file (@filename) with incorrect syntax.

User Action: For information on the correct syntax for indirect command files, see the *RSX-11M-PLUS Indirect Command Processor Manual*. Then, reenter the command line.

CON—Command execution error

Explanation: CON cannot process the specified command line. This message is followed by more specific information.

User Action: Refer to the information that accompanies this message.

CON—Command intent inconsistent with device state

Explanation: You attempted to place a device on line that is in the marked for offline state.

User Action: Place the device offline, using an explicit OFFLINE command; then, reenter the command.

CON—Command list is empty. Nothing to print

Explanation: The CON command LIST returns this message when the internal buffer for CON is empty.

User Action: No user action is required. This is an informational message.

CON—Command syntax error

Explanation: The command line does not conform to command syntax rules. The command is displayed with a question mark (?) following the last recognized keyword or parameter that you typed.

User Action: Check the command description for correct format and reenter the command line.

CON—CONFIGURE Version nnn

Explanation: CON returns this message when you enter the CON command IDENT.

User Action: No user action is required. This is an informational message.

CON—CON is unable to do an extend task operation to obtain pool for a read of the system configuration. Please ensure that a checkpoint file has been allocated and reissue the CON command

Explanation: CON requires additional memory space.

User Action: Ensure that a checkpoint file has been allocated and reenter the CON command.

CON—File open error/FILE=filename/No such file

Explanation: You specified an invalid file specification with the LIST command.

User Action: Check the device name and User File Directory (UFD); then, reenter the command line.

CON—HRC has timed out during initialization

Explanation: This message indicates an error in the I/O data structure.

User Action: Verify that your device data structures are correct.

CON—HRC version nnn

Explanation: This is an informational message only. CON returns the message when you enter the CON command IDENT.

User Action: No user action is required.

CON—Illegal command

Explanation: You used an invalid CON command, or you used a valid CON command incorrectly.

User Action: Refer to the appropriate command description; then, reenter the command line with valid command syntax.

CON—Illegal keyword

Explanation: You used an invalid keyword with the CON command, or you used a valid keyword incorrectly.

User Action: Refer to the appropriate command description; then, reenter the command line with valid command syntax.

CON—Initialization error

Explanation: CON is unable to initialize itself or its links with the RD: driver and the HRC... task.

User Action: Verify that the driver is loaded and that HRC... is installed.

CON—Initialization request of HRC... has failed. Please ensure that HRC... is installed.

Explanation: CON is unable to run HRC... because HRC... is not installed.

User Action: Use the DCL command SHOW TASKS/INSTALLED or the MCR command TAS to see if HRC... is installed. If HRC... is not installed, install it before you reenter the command.

CON—Input error on input command file
FILE=filename

Explanation: The file you have specified exists, but CON is unable to read it. The file may be corrupt or the device drive may be damaged.

This message is usually followed by more specific information.

User Action: Check the file to be sure it is readable. Also, refer to the information that accompanies this message.

CON—Internal consistency error

Explanation: An error internal to either CON or HRC has occurred. This is most often an internal addressing trap or a malfunctioning driver.

User Action: If the problem seems to be with DIGITAL-standard software, submit a Software Performance Report (SPR).

CON—Internal core pool exhausted

Explanation: CON is out of internal pool space or was invoked before checkpoint space was allocated.

User Action: If your system is out of pool space, see Chapter 8. You may need to reboot the system.

CON—Maximum indirect command file depth exceeded

Explanation: You specified an indirect command file that exceeds the maximum nesting depth of three.

User Action: No user action is required. This cannot be done.

CON—No device name matches select string

Explanation: When you entered the DISPLAY command, you specified a logical device name that does not exist in the system.

User Action: No user action is required. This is an informational message.

CON—No HRC version information available

Explanation: When HRC is not installed, the CON command IDENT returns this message.

User Action: No user action is required. This is an informational message.

CON—No reconfiguration driver is available
Output to HRC is suppressed

Explanation: The reconfiguration driver (RD:) is not loaded in the system.

User Action: Load RD: and then reenter the command.

CON—Open error on input command file
FILE=filename

Explanation: You specified an input file that does not exist.

User Action: Reenter the command with a valid file name.

CON—Unable to assign command LUN

Explanation: CON is unable to assign a logical unit number (LUN) to your terminal (TI) for its command input. This may indicate a low pool condition.

User Action: Wait for pool to be restored; then, reenter the command.

CON—Unknown command error

Explanation: For undetermined reasons, CON cannot read the command.

User Action: Submit an SPR to DIGITAL.

CON—Unknown device or invalid device specification

Explanation: The device specified for a command other than the DISPLAY command is not known to the system.

User Action: Consult the system configuration for the correct name of the target device.

CON—You must be privileged to modify the system configuration

Explanation: You attempted to modify the system configuration from a nonprivileged terminal.

User Action: Enter the CON command from a privileged terminal.

4.6.2 HRC Error Messages

The following messages are related to HRC but also come from CON:

Attempt to quiet unit for controller failed

Explanation: When a controller with units on line is taken off line, the online units are set to the marked for offline state. HRC attempts to access a unit only when the unit lacks context and can be taken off line. HRC makes 1000_{10} attempts to ensure that the device does not have context. After 1000_{10} unsuccessful attempts, CON displays this message.

User Action: Stop or abort the task that is performing I/O, or wait for the task to exit. Then, reenter the command.

Attribute format error

Explanation: HRC received a read or write attributes QIO in which the desired attributes were either incorrectly specified or inappropriate for the specified device.

User Action: Correct the task that issued the QIO. If the error is due to CON, submit an SPR.

CSR for controller not present in I/O page

Explanation: You attempted to bring a controller on line that does not exist in the current hardware configuration.

User Action: Be sure you specified the correct device name. If necessary, use the Virtual Monitor Console Routine (VMR) to include the device in the system image before attempting to bring the device on line.

Device already linked

Explanation: (This message appears only on multiprocessor systems.) You entered a LINK command for a bus run that is already linked to a CPU.

User Action: No user action is required. A bus run can be linked to one CPU only.

Device not linked

Explanation: (This message appears only on multiprocessor systems.) An UNLINK command was issued for a bus run that is not currently linked to a CPU.

User Action: No user action is required. The bus run is already unlinked.

Device is unknown in this configuration

Explanation: You attempted to bring on line a device that is unknown because it was previously assigned a control and status register (CSR) address of 160000_8 or less.

User Action: No user action is required. This is an informational message.

Executive or driver status change error

Explanation: During a controller online or offline operation, either the Executive controller status change routines ($KRBSC) or the driver's controller status change routine (xxKRB) were unable to change the state of the controller.

This error indicates a problem with your hardware.

User Action: If possible, reenter the command with a different device name.

HRC... can't take box offline. Not last box in memory

Explanation: (This message appears only on multiprocessor systems.) You attempted to take off line a memory box that is not at the end of the current physical memory configuration.

User Action: No user action is required. This cannot be done.

HRC... can't take box offline. Partition overmaps box

Explanation: An attempt was made to take a memory box off line, but the memory is currently in use for a partition.

User Action: Remove the partition with the DCL command SET NOPARTITION (MCR command SET /NOPAR) or reduce it in size with the DCL command SET PARTITION /TOP (MCR command SET /TOP). For a description of these commands, see the *RSX-11M-PLUS Command Language Manual* or the *RSX-11M-PLUS MCR Operations Manual.*

HRC... detected I/O database consistency error

Explanation: While taking a device off line, HRC detected an error in the structure of the system device database. Your system image may be corrupt.

User Action: Use the Crash Dump Analyzer (CDA) to determine the source of the problem or submit an SPR. (For more information on CDA, see the *RSX-11M-PLUS and Micro/RSX Crash Dump Analyzer Reference Manual.*)

HRC... internal addressing error

Explanation: An odd address trap (or other software trap) has occurred within HRC. This may be due to a problem in HRC or an inconsistency in the system I/O database.

User Action: Submit an SPR to DIGITAL.

HRC... internal tables insufficient for this system

Explanation: HRC creates a list of units on the stack when taking a controller off line. This message is produced if HRC encounters a controller with too many units to create a list. The maximum number of units for each controller is defined by the symbol .MXCTL in the module HRPRE.MAC and is set to 256_{10}, by default.

User Action: Modify the symbol, re-create the system image with VMR, or disconnect the units from the controller.

HRC... invalid device specification

Explanation: (This message appears only on multiprocessor systems.) HRC received a request to bring memory on line, but the specified device was not a memory box.

User Action: Reenter the command with a valid memory box.

HRC... request format error

Explanation: HRC received a QIO with an incorrect parameter format. If the issuing task for the request was CON, this error implies a software fault.

User Action: Submit an SPR to DIGITAL.

HRC... unable to access device to size drive

Explanation: When certain types of device units are brought on line, HRC must "size" the drive (for example, to determine if a DB device is an RP05 or an RP06). This message is issued if the sizing process fails for any reason (for example, the drive is not turned on when you issue the CON command ONL).

User Action: Be sure the drive is turned on; then, reenter the command.

HRC... unknown error from online/offline call

Explanation: HRC received an unsuccessful status from the driver online/offline call. The status was not one of the expected return codes.

User Action: Submit an SPR to DIGITAL.

Invalid device descriptor

Explanation: HRC received a request to take a port off line, but the port description was invalid.

User Action: Modify the task so that it uses a valid port description.

Memory box parameter error

Explanation: You attempted to bring on line a set of memory boxes whose current physical configuration does not allow HRC to execute the specified command.

User Action: Reset the memory boxes to a valid configuration.

Offline controller failure

Explanation: HRC attempted to take a controller off line (implicitly; for example, during an UNLINK operation), but the controller did not change state correctly.

User Action: Enter explicit OFFLINE commands to place the controllers off line individually. When you attempt to take the controller that resulted in this error off line, the OFFLINE command will fail with a more specific error message.

Offline unit failure

Explanation: HRC attempted to take a unit off line (implicitly; for example, during an UNLINK operation), but the unit did not change state correctly.

User Action: Enter explicit OFFLINE commands to place the units off line individually. When you attempt to take the unit that resulted in this error off line, the OFFLINE command will fail with a more specific error message.

Parameter error

Explanation: HRC received a request to link a bus run, but the device specified was not a bus switch. If the requesting task was CON, this probably implies a software fault.

User Action: Submit an SPR to DIGITAL.

Syntax error

Explanation: (This message appears only on multiprocessor systems.) HRC received a command to link a bus run to a device that is not a CPU.

User Action: Reenter the command with a valid CPU specification.

Timeout on unit quieting operation

Explanation: HRC was unsuccessful in attempting to "catch" a unit in its quiet state to take it off line after 1000_{10} attempts.

User Action: Stop or abort the task that is performing I/O, or wait for the task to exit. Then, reenter the command.

Unable to access bus run

Explanation: HRC was unable to create or remove the vectors for a device because the device's bus run was inaccessible. Because HRC checks bus run status and uses the marked for online state for devices on offline bus runs, this message probably implies a software fault.

User Action: Submit an SPR to DIGITAL.

Unable to switch unit from the current controller

Explanation: This error is displayed for two reasons:

1. HRC attempted to take off line the controller for a port to which the bus switch is currently switched.
2. HRC attempted to take off line a controller (with units) whose access path cannot be changed.

User Action: No user action is required. HRC cannot take controllers off line under these circumstances.

Unable to take unit with context offline

Explanation: You attempted to take one of the following units off line:

- A mounted Files–11 device
- An attached device
- A logged-in terminal
- A bus switch whose bus run is currently linked
- A memory box unit whose memory is currently on line

User Action: No user action is required. This cannot be done.

Chapter 5

Virtual Monitor Console Routine (RSX-11M-PLUS Systems Only)

This chapter describes the Virtual Monitor Console Routine (VMR), a privileged system task that allows you to configure your system image file. The chapter begins with a brief introduction to VMR and continues with information on running VMR and using VMR commands.

If you are familiar with VMR, you may want to skip directly to Section 5.5, which is called VMR Command Descriptions. Section 5.5 contains command definitions, a list of parameters and keywords you can use for each command, and examples of the commands.

A list of VMR error messages and suggested user actions appears at the end of the chapter.

5.1 Introduction to VMR

VMR commands are a subset of Monitor Console Routine (MCR) commands. They differ from MCR commands in that VMR commands are directed to the disk image of a system rather than to the current running system. You use VMR to make the same changes to the system image file on disk that you would make to the running system with MCR. The system image file that you configure by using VMR commands can later be bootstrapped.

There are three types of VMR commands: initialization, informational, and task control. Table 5-1 contains a list of VMR commands grouped according to the function they perform.

See Section 5.5 for a description of each VMR command.

Table 5-1: Functional List of VMR Commands

Type of Command	Command Name
Initialization	ASSIGN CONFIGURATION INSTALL LOAD REDIRECT SAVE SET TIME UNLOAD
Task control	ALTER CANCEL FIX-IN-MEMORY REASSIGN REMOVE RUN UNFIX
Informational	DEVICES LUNS PARTITIONS TASKLIST

5.2 Getting Started

Before invoking VMR, use the DIGITAL Command Language (DCL) command ASSIGN (MCR command ASN) to assign the logical device names SY: and LB: to the device that holds the system image file you want to modify. Also, set your default directory and User Identification Code (UIC) to the system directory (SYSUIC) by using the DCL command SET DEFAULT /NONAMED (MCR command SET /NONAMED followed by the command SET /UIC). By default, the system directory is [1,54].

The symbol definition file that corresponds to the system image file must also meet the following requirements:

- It must have the same file name as the system image file.
- It must have a file type of STB.
- It must reside on the same device and in the same directory as the system image file (that is, on device LB: and in the system directory).

When you have corresponding system image and symbol definition files, you can run VMR. The subsections that follow describe the methods for running VMR.

5.2.1 Running VMR

Invoke the VMR task image with the RUN command or the task name (VMR, by default).

Formats

```
>

>
```

If you precede the task image file name (VMR.TSK) with a dollar sign ($), the system searches for the file in the System Task Directory (STD) on the LB: device. The library directory is searched first and then the system directory. (RSX-11M-PLUS and Micro/RSX systems are supplied with the system directory set to [1,54] and the library directory set to [3,54].) If you do not include the dollar sign, the system searches for the task image file in the directory to which your terminal is currently set.

VMR responds by requesting the name of the system image file you want to modify. Enter the name of the system image file at the Enter filename: prompt. For example:

```
Enter filename:
```

After you enter the name of the system image file, VMR opens and reads the symbol definition file. Then, VMR opens the system image file and verifies that the file is actually a system image. If it is, VMR displays the explicit VMR prompt (VMR>). At the prompt, enter any VMR command, or exit from VMR by pressing CTRL/Z.

If the system image file you specify has Executive data space support and you are running VMR on your system for the first time, VMR loads the data space and displays the following messages before issuing the VMR prompt:

```
Loading Executive data space
Data space loading completed
```

5.2.2 Running VMR with Indirect Command Files

To run VMR with an indirect command file, type VMR followed by a space. Then, enter an at sign (@) followed by the name of the indirect command file. For example, to run VMR with the command file TASKS.CMD, enter the following command line:

```
>
```

After the system loads VMR, VMR executes the indirect command file. When VMR reaches the end of the file, it exits and returns control to the system. Note that the first line of the command file must contain the name of the system image file you want to modify.

Once you have started VMR, you can execute an indirect command file anytime VMR prompts for input. Indirect command files are useful when you want to repeat a series of commands for several different systems (INSTALL commands, for example).

To execute an indirect command file interactively, enter an at sign (@) followed by the name of the indirect command file at the VMR> prompt. For example:

```
VMR>
```

VMR executes the indirect command file, extracting command input from the specified file. When it reaches the end of the file, it redisplays the VMR> prompt and waits for further input.

VMR supports two levels of indirect command files; you can invoke a second indirect command file from within the first one. However, if you try to invoke a third level (that is, if you invoke another indirect command file from within the second file), VMR returns an error message.

5.3 VMR Command Syntax

The following sections describe how to format VMR commands and how to enter comments in VMR command lines.

5.3.1 VMR Command Format

VMR command lines consist of a command name, parameters, one or more keywords, and keyword values. The components of a VMR command line are described as follows:

Command name

Specifies the name of the VMR command.

You can abbreviate all VMR command names to three characters. VMR accepts three characters and then searches for a space or tab followed by the command parameter or a RETURN character (if there are no parameters). Note that VMR does not permit embedded spaces or tabs in command names.

Parameter

Specifies the parameter for the command.

Certain VMR commands require parameters. VMR parameters can be file specifications, keywords, devices, or task names. Always precede a parameter with a space or tab.

Keyword

Specifies one or more keywords for the command.

VMR keywords are command specific. Most VMR keywords are preceded by a slash (/). Some VMR commands permit more than one keyword per command line.

VMR keywords can be placed in any order in a command line and still have the same meaning. For example, the following two command lines have the same meaning:

```
VMR>
```

```
VMR>
```

Keyword value

Specifies the value for the keyword.

Always precede a keyword value with an equal sign (=) or colon (:). In the following command line, =SUPER is the keyword value assigned to the keyword /TASK.

```
VMR>
```

5.3.2 Including Comments in VMR Commands

VMR interprets an entire line of text as a comment if the first character in the line is a semicolon (;). For example:

```
VMR>;
```

To insert a comment within a command line, use an exclamation point (!) at the start of the comment, and use an exclamation point or RETURN at the end of the comment. VMR ignores all text between the two exclamation points or between the exclamation point and the RETURN character. For example:

```
VMR>
```

Comments are especially useful to clarify command lines in VMR indirect command files.

5.3.3 Extending a VMR Command Line

You can extend a command line by using a hyphen (-) immediately before the line terminator. Then, after you press the RETURN key, VMR returns the prompt (VMR>) on the next display line. At the prompt, enter the remainder of your command line.

Extra command lines are called *continuation command lines*. You can enter any number of these continuation lines, but the total command line cannot exceed 132_{10} characters. Command line continuation is useful when selected keywords or options cause a command line to exceed 80 characters.

For example, an RSX-11M-PLUS system would respond to the following command line (typed over three lines) in the same way as if the entire command had been typed on one line:

```
VMR>
VMR>
VMR>
```

Regardless of how many display lines you use to type a command, if the total number of characters in the command line exceeds 132_{10}, VMR returns an error message and your command is not processed. VMR does not display the error message until after you press the RETURN key to end the final line of the command.

5.4 VMR File Specifications

VMR accepts the following standard file specification format:

```
ddnn:[directory]filename.type;ver[/IM]
```

The components of VMR file specifications are described as follows:

ddnn: Specifies the name of the device on which the file is located. A device name consists of two American Standard Code for Information Interchange (ASCII) characters (dd), which are followed by an optional 1- to 3-digit octal unit number (nn). You must include a colon (:) at the end of the device name. For example, DM1: is a valid device name.

The default device in VMR is SY0:.

[directory]

Specifies the directory in which the file is located. There are two types of directories in VMR, *numeric* and *named*.

A *numeric directory* contains two octal numbers, from 1 to 377_8, separated by a comma and enclosed in square brackets ([ ]). For example, [1,54] is a valid numeric directory.

A *named directory* consists of one to nine alphanumeric characters, enclosed in square brackets. For example, [IGGY] is a valid named directory.

If you do not specify a directory, the system defaults to the directory under which VMR is currently running.

filename

Specifies the name of the file. In VMR, a file name can consist of up to nine alphanumeric characters. For example, RSX11M is a valid file name.

There is no default file name in VMR.

type

Specifies the type of file. VMR file types can consist of one to three alphanumeric characters. You must precede a file type with a period. For example, .SYS is a valid file type.

File types are optional. However, omitting a file type may cause VMR to assign one by default. VMR assigns default file types to the following files:

File	Default File Type
System image file	SYS
Task image file	TSK
Indirect command file	CMD

ver

Specifies the version number of the file. A version number helps differentiate files that would otherwise have identical file specifications. Always precede a version number with a semicolon (;).

If you select support for decimal version numbers during system generation, the version number is between 1 and $32,767_{10}$.

If you do not specify a version number for a file, the system defaults to the highest version number of that file in the directory.

/IM

Specifies a switch that, when added to the specification for the system image file, overrides the VMR requirement for the symbol definition file. If you use the /IM switch when you specify a system image file, the only command VMR accepts is the SAVE command.

Use the /IM switch to write non-RSX–11M–PLUS system images to secondary media in bootable format. The /IM switch is also useful when you use an RSX–11M–PLUS system as a host system for developing small, memory-resident systems that must be loaded from a serial medium, such as magnetic tape.

5.5 VMR Command Descriptions

The following pages include descriptions of the VMR commands in alphabetical order. Each command description contains the following information:

Command abbreviation
Indicates the minimum number of letters that you can specify when using the command name. The command abbreviation is located at the outside margin of each page in capital letters.

Command name and function
Includes the full command name in capital letters. The command name is followed by text describing the command's function.

Command format
Describes the format of the command, including defaults, acceptable values for parameters, and the effects of each keyword on command function.

Notes
Include warnings about side effects, counteracting commands, or restrictions.

Examples
Illustrate the use of the command.

5.5.1 ALTER

Use the ALTER (ALT) command to change the static priority of an installed task.

Format

```
ALT[ER] taskname /keyword
```

Parameter

taskname

Specifies the name of the task whose priority you want to alter.

Keyword

/PRI=priority

Specifies the task's static priority. You can assign a priority from 1 to 250_{10}, where the highest priority is 250_{10} and the lowest priority is 1. For example, a task with a priority of 200_{10} takes precedence over a task with a priority of 199.

The system assumes that the priority value you specify is an octal number, unless you place a period after the value.

Example

```
VMR>
```

Alters the static priority of task TEST to 248_{10}.

```
VMR>
```

Alters the static priority of task TEST to 50_8.

5.5.2 ASSIGN

Use the ASSIGN (ASN) command to define, delete, or display logical device assignments. Logical device assignments are a way to associate logical names with physical, pseudo, or logical devices. When you assign a logical name to a pseudo or logical device, the system resolves the assignment to the associated physical device.

There are several types of logical device assignments. VMR supports the manipulation of global assignments only. Global assignments apply to all tasks in the running system.

A logical device name has the same syntax as that of a physical device unit. It consists of a 2-character ASCII name (alphabetic) and an optional 1- or 2-digit octal unit number, followed by a colon (:). The 2-character name can be either equivalent to a standard device name (for example, DK:) or formed by two letters picked at random (for example, ZZ:). If a logical device name is identical to a physical device name, the logical name is the one used by the system.

Note

> You cannot use the VMR command ASSIGN to manipulate logical names of any form other than llnn:. This restriction applies even if your system includes extended logical name support.

The ASSIGN command has three formats. Note that a keyword (/SYSTEM or /GBL) is part of the required syntax for each format.

Format 1: Defining Global Assignments

```
ASN ppnn:=ll[nn]:/keyword
```

Parameters

ppnn:

Specifies a physical, logical, or pseudo device name.

ll

Specifies a logical device name.

nn

Specifies an optional 1- or 2-digit unit number.

Keywords

/GBL
/SYSTEM

Define systemwide (global) logical device assignments. /GBL and /SYSTEM are synonyms.

Format 2: Displaying Global Assignments

```
ASN /keyword
```

ASN

Keywords

/GBL
/SYSTEM

Display systemwide (global) logical device assignments. /GBL and /SYSTEM are synonyms.

Format 3: Deleting Global Assignments

```
ASN =[llnn:]/keyword
```

llnn

Specifies a logical device name, where nn is an optional 1- or 2-digit unit number. If you omit llnn:, ASSIGN deletes all global assignments.

Keywords

/GBL
/SYSTEM

Deletes systemwide (global) logical device assignments. /GBL and /SYSTEM are synonyms.

Examples

```
VMR>
```

Defines the global logical device name GB0: and assigns it to the device DM.

```
VMR>
GB    DMO:    Global  (Terminal)
```

Displays all global assignments for the system.

```
VMR>
```

Deletes all global assignments.

5.5.3 CANCEL

Use the CANCEL (CAN) command to cancel time-based initiation requests for a task. These requests result from the Executive directive RUN (RUN$) or any of the time-synchronized variations of the VMR command RUN.

The CANCEL command does not affect the execution of a task active in the saved system image. The CANCEL command removes only the time-based schedule requests still in the queue.

Format

```
CAN[CEL] taskname
```

Parameter

taskname

Specifies the name of the task whose time-based initiation requests are to be canceled.

Example

```
VMR>
```

Cancels all periodic rescheduling and time-based initiation requests for task XKE.

5.5.4 CONFIGURATION

Use the CONFIGURATION command (CON) to set the control and status register (CSR) or vector address of a device or to display system configuration information.

The CON command has two formats.

Format 1: Setting the CSR and Vector Address

```
CON[FIGURATION] SET controller option=value
```

Parameters

controller

Specifies the name of the controller for which the CSR or vector address is to be changed (for example, RHA or DMA).

option

Specifies one of the following options:

CSR=value	Sets the new CSR address for the controller.
VEC=value	Sets the new vector address for the controller.

Format 2: Displaying the System Configuration

```
CON[FIGURATION] DIS[PLAY] [keyword][FOR string]
```

Keywords

CONTROLLERS
UNITS
CPU
FULL

Parameter

FOR string

Specifies an optional parameter that allows the selective display of a particular device or device group. When you use this parameter, VMR displays the requested information for the devices matching the specified string.

Keyword Descriptions

CON[TROLLERS]

Displays every device controller in the current configuration. The CONTROLLERS display contains the following information:

- Controller's name
- Controller's status (whether it is on line or off line)
- UNIBUS Run Mask (URM)
- Controller's CSR address

- Controller's starting vector
- Controller's priority

UNI[TS]

Displays every device unit in the current system. The UNITS display contains the following information:

- Logical device name.
- Controller name and physical unit number of the device on that controller. For dual-access devices, this information is supplied for each port.
- Device status (whether the device is on line or off line).
- A field indicating that the device driver is loaded.

CPU

Displays all the central processing units (CPUs) in the current system.

FUL[L]

Displays the CPUs, the controllers, and the device units in the current configuration.

If you do not specify a keyword, the default is a FULL display.

Examples

```
VMR>
```

Sets the CSR address for the third DH11 controller to 160100.

```
VMR>
```

Sets the vector address for the third DH11 controller to 340.

```
VMR>
RHA    OFL     CPA     CSR=176700       VEC=254 PRI=5
RHB    OFL     CPA     CSR=176300       VEC=150 PRI=5
RHC    OFL     CPA     CSR=172040       VEC=204 PRI=5
RHD    OFL     CPA     CSR=172440       VEC=224 PRI=5
   .
   .
   .
YMA    OFL     CPA     CSR=170500       VEC=300 PRI=5
YZA    OFL     CPA     CSR=160100       VEC=340 PRI=5
VMR>
```

Displays information about every device controller in the current configuration for a single-processor system.

```
VMR>
DB0:    RHA0:           OFL     DRIVER
DB1:    RHA1:           OFL     DRIVER
DB2:    RHA2:           OFL     DRIVER
DB3:    RHA3:           OFL     DRIVER
    .
    .
    .
NL0:                    OFL     DRIVER
VT0:                    OFL     DRIVER
RD0:                    ONL     DRIVER
VMR>
```

Displays information about every device unit in the current configuration for a single-processor system.

```
VMR>
CPA
VMR>
```

Displays the CPU for a single-processor system.

```
VMR>
CPA
RHA     OFL     CPA     CSR=176700      VEC=254 PRI=5
RHB     OFL     CPA     CSR=176300      VEC=150 PRI=5
RHC     OFL     CPA     CSR=172040      VEC=204 PRI=5
RHD     OFL     CPA     CSR=172440      VEC=224 PRI=5
    .
    .
    .
DB0:    RHA0:           OFL     DRIVER
DB1:    RHA1:           OFL     DRIVER
DB2:    RHA2:           OFL     DRIVER
DB3:    RHA3:           OFL     DRIVER
    .
    .
    .
NL0:                    OFL     DRIVER
VT0:                    OFL     DRIVER
RD0:                    ONL     DRIVER
VMR>
```

Displays the CPU, the controllers, and the device units in the current configuration for a single-processor system.

```
VMR>
RHB     OFL     CPA     CSR=176300      VEC=150 PRI=5
RHC     OFL     CPA     CSR=176700      VEC=254 PRI=5
RHD     OFL     CPA     CSR=172440      VEC=224 PRI=5
RHH     OFL     CPB     CSR=176700      VEC=254 PRI=5
RHM     OFL     CPC     CSR=176700      VEC=254 PRI=5
RHN     OFL     CPC     CSR=172440      VEC=224 PRI=5
```

```
RHR    OFL     CPD     CSR=176300       VEC=150 PRI=5
RHS    OFL     CPD     CSR=176700       VEC=254 PRI=5
VMR>
```

Displays information about all the RH-type controllers in the current configuration for a multiprocessor system.

```
VMR>
CPA
CPB
CPC
CPD
VMR>
```

Displays all the CPUs in the current configuration for a multiprocessor system.

```
VMR>
CPA
CPB
CPC
CPD
   .
   .
   .
BS0:  BSA0:  BSB0:  OFL  DRIVER
BS1:  BSC0:  BSD0:  BSE0:  BSF0:  OFL  DRIVER
   .
   .
   .
DB0:  RHH0:  RHM0:  OFL  DRIVER
   .
   .
   .
VMR>
```

Displays all the CPUs, controllers, and device units in the current configuration for a multiprocessor system.

```
VMR>
YHA    OFL     CPA     CSR=160020       VEC=310 PRI=5
YHB    OFL     CPA     CSR=160040       VEC=320 PRI=5
YHC    OFL     CPA     CSR=160060       VEC=330 PRI=5
VMR>
```

Displays information about all the YH-type controllers in the current configuration for a multiprocessor system.

5.5.5 DEVICES

Use the DEVICES (DEV) command to display the symbolic names of all device units in the system image of a particular device type. If you do not specify a device type, VMR displays the symbolic names of all device units.

The device names appear in a single vertical column. A second column, to the right, contains additional information about each device (see Notes).

Format

```
DEV[ICES] [dd:]
```

Parameter

dd:

Specifies the device type.

Notes

1. The pseudo device SY: is not redirected to the bootstrapped device until the system is bootstrapped.
2. The following list describes the terms that can appear in the second column of the device list when you use the DEVICES command. (More than one term can appear on the same line.)
   - LOADED indicates that the driver for the device is loadable and currently loaded.
   - UNLOADED indicates that the driver for the device is loadable but not loaded.
   - The absence of either LOADED or UNLOADED indicates that the driver is permanently resident. This means the driver is part of the Executive.
   - OFFLINE indicates that a device has yet to be brought on line by the reconfiguration task. All physical devices in the device list have this notation. (For more information about device states, see Chapter 4.)
   - A device name in the second column is the device to which the corresponding device in the first column has been redirected.

Example

```
VMR>.
LP0:  Loaded
DB0:  Loaded
DB1:  Loaded
DB2:  Loaded
DD0:  Loaded
DD1:  Loaded
DK0:  Loaded
DK1:  Loaded
    .
    .
    .
DR0:  Loaded
DR1:  Loaded
DS0:  Offline  Loaded
DS1:  Offline  Loaded
DT0:  Loaded
DT1:  Loaded
DX0:  Loaded
DX1:  Loaded
EM0:  Unloaded
MM0:  Loaded
MM1:  Loaded
CO0:  TT0:
TT0:  Loaded
TT1:  Loaded
TT2:  Loaded
    .
    .
    .
NL0:
TI0:
CL0:  LP0:
LB0:  DR0:
SY0:  DR0:
VMR>
```

Displays the symbolic names of all device units in the system image. The terms "Loaded" and "Unloaded" refer to the status of the driver for the device. "Offline" refers to the current device state.

5.5.6 FIX-IN-MEMORY

Use the FIX-IN-MEMORY (FIX) command to load a task into memory and to load a region into its partition in the system image. VMR can fix a task in the system image only if the partition in which it is to be fixed is available.

Fixed tasks remain in memory even after they exit. Therefore, the system can run a fixed task faster because the task resides in memory and does not need to be reloaded. However, reexecuting fixed tasks is not always practical. If you expect to reexecute a fixed task, you should write it in such a way that it initializes all impure data and logical unit number (LUN) assignments at run time.

Use either the UNFIX or REMOVE command to remove a fixed task from memory or to remove a region from its partition.

Format

```
FIX taskname[/keyword]
```

Keywords

/DIR
/NSF
/REG
/RON

Parameter

taskname

Specifies the task or region that you want to fix in memory. If you do not specify a keyword, VMR assumes that the task is a single-user task.

Keyword Descriptions

/DIR

Specifies that the task to be fixed is a directive common. Directive commons are used to hold some of the Executive directives instead of keeping them in Executive address space. Because directive commons contain Executive code, they must be installed and fixed in the system image before the system can be bootstrapped. Directive commons are installed and fixed in the system image by commands in the SYSVMR.CMD file as part of the system generation procedure.

Individual directive commons are installed and fixed contiguously in memory to form one large directive common called DIR11M. DIR11M must be fixed in memory before any other directive commons can be fixed. Other directive commons must be fixed contiguously to DIR11M.

Directive commons cannot be shuffled, which means that the Shuffler cannot move them around in memory after they have been fixed. For more information about the Shuffler, see Chapter 8.

To eliminate a directive common, use the REMOVE command with the /REG keyword.

/NSF

Specifies that a fixed task cannot be shuffled. This means that the Shuffler cannot move the task around in memory after the task has been fixed. For information about the Shuffler, see Chapter 8.

/REG

Specifies that the task to be fixed is a common region.

/RON

Specifies that the task to be fixed is a multiuser task and that the read-only portion is to be fixed in memory.

When you use the /RON keyword, specify the entire task name.

Restrictions

The following restrictions apply when you fix tasks:

- You must install a task before you fix it.
- Active tasks cannot be fixed.
- Do not fix a checkpointable task because fixing a checkpointable task makes the task noncheckpointable.
- You cannot fix tasks whose names are in the forms ...xxx and xxx$$$ because that copy of the task is never executed. (The task is a prototype task.)

Examples

```
VMR> [illegible]
```

Fixes the task XKE in its partition.

```
VMR> [illegible]
```

Fixes the resident common TTCOM in its partition and specifies that the common cannot be shuffled.

```
VMR> [illegible]
```

Indicates that the EDI editor is built as a multiuser task and fixes the task's read-only portion in memory.

5.5.7 INSTALL

The INSTALL (INS) command makes a task or common recognizable to the system. An installed task is dormant (not active) until the Executive receives a request to run the task. INSTALL accepts optional keywords that provide parameters not specified when the task was built or that override previously specified parameters.

When you install a task, the system creates a Task Control Block (TCB) for the task in a memory-resident table called the System Task Directory (STD). The task parameters contained in the TCB enable the system to run the task when requested to do so by an MCR command or by a system directive. You cannot run a task until it has been installed.

Note

> The MCR command RUN has an option that installs a task, runs the task, and then removes the task. For more information, see the *RSX-11M-PLUS MCR Operations Manual.*

When you install a task in an unsaved system, the system stores the file identification (file ID) for the task image file in the task's header. In addition, it stores the logical block number (LBN) of the task image in the Task Control Block (TCB). However, in the saved system, the LBN of the installed task is not retained in the TCB. Instead, when you save the system, the SAVE task uses the file ID located in the task's header to replace the LBN contained in the TCB. Then, when you reboot the system, the system opens the task file and replaces the file ID (stored in the TCB) with the current LBN. (If you delete a task before rebooting the system, the system cannot open the task file. As a result, the system considers the task uninstalled; so, it automatically removes the task's TCB from the STD.)

By replacing the LBN contained in the TCB with the file ID located in the task's header, the MCR command SAVE makes it possible to move task files to different physical locations without losing their file identifications. For example, the Disk Save and Compress Utility (DSC; supplied with RSX-11M-PLUS systems only) and the Backup and Restore Utility (BRU) compress and copy a disk that contains a saved system. These procedures move the task files to different physical locations. However, the TCB entries contain task file identifications instead of LBNs, so the file identifications are preserved. Consequently, the system can function normally after it is rebooted. (For more information on DSC and BRU, see the *RSX-11M-PLUS Utilities Manual.*)

Format

```
INS[TALL] filespec[/keyword(s)]
```

Keywords

/AFF=[URM(s)]
/CKP=option
/CLI=option
/DFB=option
/FMAP=option
/INC=size
/IOP=option
/PAR=pname
/PMD=option
/PRI=number
/PRO=[system,owner,group,world]
/RON=option
/ROPAR=pname
/SEC=option
/SLV=option
/SYNC=option
/TASK=taskname
/UIC=[group,member]
/WB=option
/XHR=option

Parameters

filespec

Specifies a task image file specification in the following format:

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

The task image file you specify must be on the device LB:. The default for the file type is TSK.

Keyword Descriptions

/AFF=[URM(s)]

Specifies the UNIBUS Run Mask (URM) affinity of a task running on a multiprocessor system. When you specify the /AFF keyword, the installed task runs on the processor to which the specified URMs are connected or the task does not run at all (if the tasks are on different CPUs).

You must specify at least one URM. Enclose the URM or URMs in square brackets. If you specify more than one URM, separate them with commas. For example:

```
/AFF=[UBE,UBF,UBH]
```

The option [URM(s)] represents the 3-letter name of the UNIBUS Run Mask (URM) for the central processing unit (CPU) or switched UNIBUS run. The following names are valid URM specifications:

CPA	UBE	UBK	UBP
CPB	UBF	UBL	UBR
CPC	UBH	UBM	UBS
CPD	UBJ	UBN	UBT

/CKP=option

Enables or disables checkpointing for a task.

The options are YES and NO. If you specify /CKP=YES, checkpointing is allowed. If you specify /CKP=NO, checkpointing is disabled for the task. The /CKP keyword overrides the /CP switch specified at task build.

The task image file need not contain allocated checkpoint space. The /CKP=YES keyword instructs the system to allocate checkpoint space for the task, when required, from a checkpoint file. (For more information on allocating checkpoint space, see Chapter 1.)

The default is /CKP=YES for tasks that are eligible for checkpointing.

/CLI=option

Indicates that the task being installed is or is not a command line interpreter (CLI). (For information on command line interpreters, see Chapters 17 and 18.)

The options are YES and NO. /CLI=YES indicates that the task is a command line interpreter. /CLI=NO indicates that the task is not a command line interpreter.

To indicate that a task is a CLI, install it with the /CLI=YES keyword or task build it with the Task Builder (TKB) switch /CL. After you install the CLI, you can use the MCR command CLI to set your terminal to it. (See the *RSX-11M-PLUS and Micro/RSX Task Builder Manual* for more information on the TKB switch /CL. For information on the CLI command, see the *RSX-11M-PLUS MCR Operations Manual.*)

The default is specified at task-build time.

/DFB=option

Specifies whether or not the task's header is bound to Executive data structures when the task is installed. More specifically, /DFB specifies whether or not the Unit Control Block (UCB) pointers for the task's Logical Unit Table (LUT) and the Partition Control Block (PCB) pointers for the task's address window are bound to the Executive logical unit number (LUN) assignments.

The options are YES and NO. /DFB=YES specifies that binding to the Executive will be deferred until the task is loaded into memory. If you specify /DFB=YES, you can avoid problems that occur when a task is installed in more than one system (running or disk resident) at the same time. If you specify /DFB=NO, binding occurs at the time the task is installed.

The default is /DFB=YES.

/FMAP=option

Informs the system that the task uses the fast-mapping feature of the Executive. You can use the /FMAP keyword only with tasks that have external headers.

The options are YES and NO. /FMAP=YES indicates that the task uses the fast-mapping feature of the Executive. /FMAP=NO indicates that the task does not use the fast-mapping feature of the Executive.

See the description of the /XHR keyword for more information about external headers. See the *RSX-11M-PLUS and Micro/RSX Executive Reference Manual* for more information about fast mapping.

The default is specified at task-build time.

/INC=size

Indicates the extension size of a task. The extension size refers to the number of additional words of address space you want allocated to a task. The extension size is usually expressed in octal but may also be expressed in decimal by following the size with a period (.).

Tasks such as the MACRO-11 assembler (MAC), TKB, the Peripheral Interchange Program (PIP), the File Structure Verification Utility (VFY), and the Indirect Command Processor (AT.), use additional address space as dynamic memory in which to allocate symbol tables and buffers. The INSTALL command allocates the dynamic region between the end of the task image and the end of the task's address space. The /INC keyword defines the number of words in the region.

The /INC keyword overrides the TKB options EXTTSK and PAR.

When you install a common region with an increment, it no longer fits in its original task image file. Therefore, you cannot specify both /INC=size and /WB=YES when you install a common.

The default size is the size specified at task-build time or zero.

/IOP=option

Informs the system that a privileged task needs access to the I/O page. The options are YES and NO. If you specify /IOP=YES, INSTALL determines that the task uses the I/O page (APR 7) and, if the task is larger than 8K words, INSTALL issues a warning message. If you specify /IOP=NO, INSTALL assumes that the task does not need to use the I/O page, and it does not issue a warning message.

/IOP complements the TKB switch /IP. Specifying /IP informs INSTALL that the task is intentionally mapping to the I/O page.

The default is specified at task-build time.

/PAR=pname

Indicates a partition that overrides the partition name specified at task-build time. If the partition you specify does not exist, VMR issues a warning message and attempts to install the task in the GEN partition.

The default is specified at task-build time.

/PMD=option

Informs the system whether a Postmortem Dump (PMD) is required.

The options are YES and NO. If you specify /PMD=YES, the system performs a PMD of the task when the task is aborted because of a synchronous system trap (SST) error condition. If you specify /PMD=NO, the system does not perform a PMD when the task is aborted.

To generate a PMD, install the Postmortem Dump task PMD... in the system. /PMD overrides the TKB switch /PM.

The default is specified at task-build time.

/PRI=number

Specifies the priority for the task. The value of number can be between 1 and 250_{10}, where 250 is the highest priority. The standard numbering conventions apply; that is, the default is octal, and the value is a decimal number if it is followed by a period.

The default is specified at task build time or it is equal to 50_{10}.

/PRO=[system,owner,group,world]

Sets the protection mask for a common region. The protection specification has the following format:

```
[RWED,RWED,RWED,RWED]
```

The user classes (system, owner, group, world) are positional; that is, the location of the word in the string defines the user class to whom the code applies. The square brackets around the protection code are required syntax.

For each user class, the system recognizes the following types of protection access codes:

Code	Type of Access
R	Read access
W	Write access
E	Extend access
D	Delete access

To assign an access type to a particular group, place the access type or types in the same position as the group name and separate the groups with commas. For example:

```
/PRO=[RWED,RWED,ED,R]
```

To specify the default protection for a user class, enter an asterisk (*) instead of access codes in that classes' place in the string. To prevent a group from having any type of access, insert a comma in the group's position. For example:

```
/PRO=[RWED,RWED,,RWED]
```

In this example, a task running under the same UIC group as the file's owner has no access.

You cannot specify protection options for a read-only common region or the read-only portion of a multiuser task. A task linked to a common region with an improper protection mask cannot be installed.

The default is /PRO=[RWED,RWED,RWED,RWED].

/RON=option

Informs the system that a common region is or is not a read-only region. This keyword overrides the effect of the /PRO keyword.

The options are YES and NO. If you specify /RON=YES, the region is installed as read-only. If you specify /RON=NO, the region is installed with read/write access.

The default is /RON=NO.

/ROPAR=pname

Indicates the read-only partition pname into which the read-only portion of a multiuser task is to be installed.

If the partition you specify does not exist, VMR issues a warning message and attempts to install the read-only portion of the task into the partition containing the read/write portion of the task. The /ROPAR keyword overrides the TKB option ROPAR.

The default is specified at task-build time.

/SEC=option

Places the Task Control Block (TCB) for the task being installed in secondary pool. (For information on secondary pool, see Chapter 8.)

The options are YES and NO. If you specify /SEC=NO, the TCB for the task is placed in primary pool instead of secondary pool. The /SEC keyword is valid only for tasks whose names are in the form ...xxx or xxx$$$.

The default is /SEC=YES.

/SLV=option

Informs the system of the task's slave status.

The options are YES and NO. If you specify /SLV=YES, data sent to the installed task is marked with the TI: and UIC of the sending task. When the installed task receives the data, the system sets the task's TI: and UIC to the TI: and UIC associated with the data. The /SLV=YES keyword overrides the TKB switch /SL. If you specify /SLV=NO, the TI: of the task does not change when the task receives data.

The /SLV keyword cannot be used with tasks whose names are in the form ...xxx or xxx$$$. These tasks are prototype tasks and cannot be installed as slaved tasks.

The default is specified at task-build time.

/SYNC=option

Synchronizes task use of a common region by preventing tasks that share the common from running at the same time (on different CPUs). This prevents the second task from reading from or writing to the common while the first task is executing.

This keyword applies to multiprocessor systems only.

The options are YES and NO. You can specify /SYNC=YES for two or more tasks, but only one of the tasks can execute at one time. If one of the tasks becomes blocked, the others compete to execute.

The default is /SYNC=NO.

/TASK=taskname

Specifies the name by which the system refers to the task.

The default taskname is the name specified at task-build time or the first six characters of the file name.

/UIC=[group,member]

Specifies the UIC for the task. The task's UIC determines the file-protection class (system, owner, group, world) to which a task belongs. The square brackets are required syntax.

The default is the UIC specified at task-build time. The UIC you specify with the INSTALL /UIC command applies only to tasks started in one of the following ways:

- Issuing the Executive directive Run Task (RUN$)
- Entering a RUN command that specifies a time delay before run time

Tasks run under the protection UIC of the issuing terminal (TI) if they are started in one of the following ways:

- Typing the name as a command (for example, PIP for the Peripheral Interchange Program or TKB for the Task Builder)
- Entering a RUN command that does not specify a time delay before run time

/WB=option

Specifies the file to which a read/write common will be checkpointed. The options are YES and NO. By default, a common is checkpointed to a system checkpoint file.

If you specify /WB=YES, the common is checkpointed to its original task image file; also, when the common is removed, it is written back to its original task image file. However, if a common is installed with an increment, it will no longer fit into its original task image file. Therefore, when installing a common, you cannot specify /WB=YES with /INC=size.

If you specify /WB=NO, the common is checkpointed to a system checkpoint file, and it is not written back to the original task image file when it is removed.

If a common is a read-only region, there is no need to checkpoint it. Thus, INSTALL ignores the /WB switch if a common is installed with /RON=YES.

The default is /WB=NO.

/XHR=option

Specifies that the Executive's copy of the task's header will be in an area external to the system's pool.

The options are YES and NO. If you specify /XHR=YES, the task is installed with an external header. If you specify /XHR=NO, the task is installed with a resident header.

When you specify /XHR=YES, the Executive's copy of the task's header is placed in a reserved area immediately before the task image. When the task executes, the Executive uses the copy of the header instead of the actual header. If the task is checkpointed, the system writes the entire task image and the copy of the header into the checkpoint file. Thus, the header in the task image is left unchanged.

/XHR corresponds to the TKB option /XH. The default is specified at task-build time. (For a detailed description of the TKB option /XH, refer to the *RSX-11M-PLUS and Micro/RSX Task Builder Manual.*)

Notes

1. A TCB contains both the address of the load-device Unit Control Block (UCB) and the LBN of the task image file. Because task image files are contiguous, the Executive can load the task image in one transfer when an MCR command or system directive requests an installed task. However, if the task is greater than 32K words, loading requires more than one transfer. For example, if a task is built with user instruction and data space, the Executive loads the data space first, and then it loads the instruction space.
2. If you use the INSTALL command to install a resident library, INSTALL makes an entry for the resident library in the Common Block Directory (CBD). When a task maps to the library, the system loads the library into memory. You can remove the library only when tasks are no longer mapped to it.

3. Changes made in the memory image of a common can be preserved when the common is removed by installing the common with the /WB=YES option.

   If you install the common with the /WB=YES option, the REMOVE command writes the common, with its changes, into the original task image file. Thus, the original image is replaced. When the common is reinstalled, the new version of the common (as it existed in memory when it was removed) is placed in memory.

4. You cannot use the INSTALL command to install device commons that have been created with the SET PARTITION (SET /PAR) command. (A device common is a special type of common that occupies physical addresses in the I/O page.)

   When you use the SET command to create a device partition, the following operations occur:

   - If the device partition being created is the first device partition in the system, the system automatically creates a main partition called IO PAR. The partition being created then becomes a subpartition of IO PAR.
   - The system creates an entry for the new device partition in the Common Block Directory (CBD) and inserts the partition into the directory. The device partition is now a device common. Because the system creates a common block for the device partition automatically, there is no need to install it as a common. If you attempt to do so, you will receive an error message.

Examples

```
VMR> [illegible]
```

Installs task SCAN with the default priority and directory.

```
VMR> [illegible]
```

Informs the system that the task SCAN is a command line interpreter (CLI).

```
VMR> [illegible]
```

Informs the system that the task SCAN will use the fast-mapping feature of the Executive.

```
VMR> [illegible]
```

Informs the system that the privileged task, SCAN, needs access to the I/O page.

```
VMR> [illegible]
```

Installs task MAC in the partition SYSCTL and increases the task size by 4096_{10} words.

```
VMR> [illegible]
```

Installs the fourth version of the task SCAN (file type TSK) from device LB:, group number 1, member number 111, with a priority of 103_8.

```
VMR>
```

Requests the system to install the common region SCANCM with a protection mask that permits tasks running under the same User Identification Code (UIC) as the file's owner to have read, write, extend, and delete access. Tasks with other UIC's are denied access to the common.

```
VMR>
```

Informs the system that other tasks can share a common with the task SCAN, but only one of the tasks may execute at one time.

```
VMR>
```

Specifies that the read/write common TEST will be checkpointed to its original task image file.

5.5.8 LOAD

The LOAD (LOA) command loads a nonresident (loadable) device driver into the system image. LOAD either links a driver to an already resident database or loads a loadable database with the driver. This command eliminates the need to link infrequently used device drivers permanently into the Executive. The result is the potential for an increase in the amount of system dynamic memory (pool).

You can load drivers that are up to 8K words in length. However, note that the loadable device driver itself must map to locations above 4K.

For more information on loadable device drivers, see the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual.*

Format

```
LOA[D] dd:[/keyword(s)]
```

Keywords

/CTB=cc[a,b, . . .]
/PAR=pname
/SIZE=psize
/VEC

Parameter

dd:

Specifies a 2-character American Standard Code for Information Interchange (ASCII) loadable device driver name.

Keyword Descriptions

/CTB=cc[a,b, . . .]

Specifies that the multidriver controller Controller Table Block (CTB) in the resident database be modified instead of the symbol $ddCTB in the loadable database. Note that the keyword is necessary only if the driver contains a loadable database; to load most drivers, this keyword is not required. (See the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver* for more information.)

Specify the following arguments:

cc

Specifies the name of the multidriver controller (CTB) to be used in the loadable database.

[a,b, . . .]

Specifies the slots in the CTB's Controller Request Block (KRB) table that should point at the KRBs in the loadable database. The slots are named A,B,C,D,E,F,H,J,K,L,M,N,P,R,S, and T.

/PAR=pname

Specifies the partition into which the driver is to be loaded. The pname value is the name of the partition.

The /PAR keyword overrides the partition specified at task-build time.

/SIZE=psize

Specifies the minimum size of the partition (in octal words) into which the driver is to be loaded.

VMR rounds the partition size up to the next largest unit of 100_8. The partition size must be less than $77,741_8$.

/VEC

Specifies that the driver to be loaded is *vectored.*

Vectoring is a technique for building a portable device driver. You can load a vectored driver into any RSX–11M–PLUS or Micro/RSX operating system that has the same version number as the system on which the driver was built. However, you must specify the /VEC keyword to suppress the validation of the Executive symbol table files used in the driver. /VEC avoids the comparison of the driver symbol table file and the Executive symbol table file.

For more information on vectored device drivers, see the *RSX–11M–PLUS and Micro/RSX Guide to Writing an I/O Driver.*

Notes

1. The driver for a device, for example ZY:, can indicate association with a partition by allocating one word in the driver and by labeling it with the global symbol $ZYCOM. If $ZYCOM is defined in a driver, you cannot load the driver until you create ZYCOM, because the LOAD command writes the Partition Control Block (PCB) address of ZYCOM into $ZYCOM.
2. Guidelines for building user-written, loadable device-driver tasks are defined in the *RSX–11M–PLUS and Micro/RSX Guide to Writing an I/O Driver.*
3. The LOAD command requires that a driver's TSK and STB files reside in the same directory and on the same disk as the system image file you are working on. For example, the following command requires that the files DTDRV.TSK and DTDRV.STB reside on the device SY:, under the current UIC:

   ```
   LOAD DT:
   ```

4. The LOAD command requires the following pool space:
   - If the driver database is being loaded, the database needs contiguous pool space equal to its size.
   - Each interrupt entry in the driver needs pool space for an Interrupt Control Block (ICB).
   - The PCB needs pool space.

Examples

```
VMR>
```

Loads the TU16 driver (MMDRV).

```
VMR>
```

Loads the RK05 driver (DKDRV) into DRVPAR.

```
VMR>
```

Loads the RM03 driver (DRDRV), using the RHB multidriver controller.

LUN

5.5.9 LOGICAL UNIT NUMBERS

Use the LOGICAL UNIT NUMBERS (LUN) command to display the static logical unit number (LUN) assignments for a specified task. Static assignments are those recorded in the task's disk image file (for example, assignments specified at task-build time). The display consists of a list of physical device units in one column and their corresponding LUNs in an adjoining column.

Format

```
LUN[S] taskname
```

Parameter

taskname

Specifies the name of the task whose LUNs you want to display.

Example

```
VMR>
SY0: 1
SY0: 2
CL0: 3
TT3: 4
TT3: 5
DK0: 6
TT3: 7
```

The display shows static assignments for LUNs 1 to 7 as recorded in the task image file header. No other LUNs are assigned statically to the task XKE.

5.5.10 PARTITION DEFINITIONS

Use the PARTITION DEFINITIONS (PAR) command to display a description of each memory partition in the system image.

The display consists of six columns that specify the following information:

- Partition name
- Address of Partition Control Block (PCB)
- Starting address of partition (in octal)
- Size of partition (in octal)
- One of the following partition types:

Display	Partition Type
MAIN	Main partition
TASK	Partition contains a task
DRIVER	Partition contains a device driver
RO COM	Partition is a read-only common
RW COM	Partition is a read/write common
DEVICE	Partition is a common and is mapped to the I/O page
SEC POOL	Partition is in secondary pool
CPU	Partition is a central processing unit (CPU) partition in a multiprocessor system

DEVICE partitions are used to allow tasks to communicate with specific device registers, such as the UDC and ICS/ICR-11 industrial control subsystem.

For a description of subpartitions and dynamic regions, see Chapter 1 of this manual or refer to the *RSX-11M-PLUS and Micro/RSX Executive Reference Manual.* For more information on loadable drivers, see the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver.*

- Description of partition occupant in one of the following forms:

Form	Meaning
[taskname]	An inactive memory-resident task
< taskname>	An active task
(dd:)	The specific device driver
+ xxxx +	The name of the first installed task that is associated with an unnamed common, usually the read-only segment of a multiuser task
! xxxx !	A common region

Format

```
PAR[TITIONS]
```

PAR

Example

```
VMR>PAR RET
SYSPAR 117734 00174400 00010000 MAIN
       065230 00174400 00010000 TASK <TKTN >
DRVPAR 117624 00216400 00140000 MAIN
       117340 00216400 00003100 DRIVER (DB:)
   .
   .
   .
       116634 00243300 00035300 RW COM !TTCOM !
       116550 00300600 00015600 DRIVER (TT:)
       116504 00316400 00001300 DRIVER (RD:)
       116440 00317700 00001300 DRIVER (VT:)
       116374 00321200 00000100 DRIVER (NL:)
       036504 00322600 00003200 DRIVER (DR:)
LDRPAR 117560 00356400 00002600 MAIN
       116054 00356400 00002600 TASK   <...LDR>
TSTPAR 117514 00361200 00040000 MAIN
SECPOL 117450 00421200 00022000 SEC POOL
GEN    117404 00443200 04734600 MAIN
       115454 00443200 00070600 TASK   <F11ACP>
       073400 00534000 00006000 TASK   <TT36  >
       052274 00542600 00003500 RO COM +...EDI+
       060774 00571400 00020100 TASK   <QMG...>
       037210 00620100 00036100 TASK   <HRC...>
       045104 00656200 00026400 TASK   <RMDEMO>
   .
   .
   .
       062254 01170400 00040000 TASK   <SYSLOG>
       061340 01230400 00012100 TASK   <LPP0  >
       061674 01242500 00012100 TASK   <LPP1  >
       061774 01254600 00037100 TASK   <BAP2  >
       063050 01313700 00037100 TASK   <BAP1  >
       071114 01437000 00035400 TASK   <AT.V3 >
       070100 01401400 00035400 TASK   <AT.T36>
       037474 01474400 00074500 TASK   <EDTT3 >
       072244 01571100 00026500 TASK   <PARV3 >
       042100 01631100 00064500 TASK   <TT47  >
       037540 01715600 00102700 TASK   <EDIT25>
       064534 02050500 00033500 TASK   <TTF6  >
       037724 02143000 00134300 RW COM !RMSRES!
```

Displays a description of the following memory partitions in a saved system:

Partition	Description
SYSPAR	System partition, used by the Task Termination Notification Program (TKTN) and other system tasks
DRVPAR	Contains the loadable device drivers
LDRPAR	Contains the Loader task, which is always fixed in memory

Partition	Description
TSTPAR	Used for running tests ("test" partition)
SECPOL	Secondary pool
GEN	General partition, the default for most tasks in memory

5.5.11 REASSIGN

Use the REASSIGN (REA) command to reassign a task's logical unit numbers (LUNs) from one physical device unit to another. REASSIGN has no effect on the assignments for a task that is fixed or loaded in memory. The reassignments affect only the static assignments recorded in the task's disk image file.

Format

```
REA[SSIGN] taskname LUN new:
```

Parameters

taskname

Specifies the name of the task for which the LUN is to be reassigned.

LUN

Specifies the logical unit number.

new:

Specifies the new device unit, which can be a physical, logical, or pseudo device.

Examples

```
VMR>
```

Reassigns LUN 3 of task LODN to physical device TT0.

```
VMR>
```

Reassigns LUN 3 of task TEST to logical device XX.

5.5.12 REDIRECT

Use the REDIRECT (RED) command to redirect all I/O requests from one physical device unit to another physical device unit.

You cannot redirect the following:

- The pseudo device TI:
- A device to the pseudo device TI:
- A pseudo device to itself
- An attached device
- A terminal to the null device.

You can redirect a logged-in terminal to a device other than a terminal (except for the null device); however, it is not generally advisable to do so because terminal I/O would be written to the device. If you do redirect a terminal to another type of device, you should redirect the terminal back to itself before you log out.

Format

```
RED[IRECT] new:=old:
```

Parameters

new:

Specifies the new device unit to which requests will be redirected.

old:

Specifies the old device unit from which requests will be redirected.

Examples

```
VMR>
```

Redirects all I/O requests for device TT6 to device TT3.

```
VMR>
```

Redirects all I/O requests for device LP0 to device TT0.

5.5.13 REMOVE

Use the REMOVE (REM) command to delete a task name or a region name from the System Task Directory (STD). Removing a task from the system image makes the task unrecognizable to the system. The action resulting from this command is the complement of INSTALL.

The REMOVE command cannot remove an active task from a saved system. An example of such a task is SAV; an attempt to remove SAV from a saved system results in an error message.

Format

```
REM[OVE] name[/keyword]
```

Parameter

name

Specifies the name of the task, region, or directive common to be removed.

Keyword

/REG

Specifies that the name parameter you specify is the name of a region.

You must specify this keyword when removing the directive commons. To remove the commons, use the following command line:

```
VMR>
```

Note that when you remove DIR11M, you remove all of the directive commons in the system.

Notes

1. If a task or region that is the object of a REMOVE command is fixed, the command automatically unfixes the task from the system image.
2. REMOVE automatically cancels all time-based schedule requests for the specified task.
3. In general, use MCR on the running system to remove and install command line interpreter (CLI) tasks. Then, to update the system image file, use the MCR command SAVE.

   Although you can use the VMR command REMOVE to remove a CLI from the task image file, you cannot remove the CLI until you use the MCR command CLI /ELIM to eliminate the CLI Parser Block (CPB). Likewise, you can use the VMR command INSTALL to install a CLI, but you must use the MCR command CLI /INIT to initialize the CLI. Thus, for most CLI tasks, MCR is a more direct method for eliminating and removing old CLIs or installing and initializing new ones.

   An exception to this note is the primary MCR task (MCR...). The CPB for this task can never be eliminated, and the task itself can be removed only with VMR.
4. A region cannot be removed if there are tasks installed in the system that reference that region.
5. Changes made in the memory image of a common can be preserved when the common is removed by installing the common with the /WB=YES option.

If the common is installed with the /WB=YES option, the REMOVE command writes the common, with its changes, into its original task image file, replacing the original image. When the common is reinstalled, the new version of the common is placed in memory.

Example

```
VMR>
```

Removes the task named SCAN from the STD. The task image file remains unaffected.

5.5.14 RUN

Use the RUN command to initiate the execution of an installed task at any of the following time options:

- At a time increment from the system image time
- At a time increment from clock unit synchronization
- At an absolute time of day
- At one clock tick after the system bootstrap

All of these options are available with or without rescheduling, and each option has a separate format.

You cannot run a task whose name is in the form ...xxx or xxx$$$. These tasks are prototype tasks, which are never executed.

Note

> The correct value to be inserted in the clock queue for Format 1 to Format 3 is derived from the current value of the system time stored in the system image. Therefore, the system image time must be set before any RUN requests are made, or they may not occur at the correct time after the system is bootstrapped.

Format 1: Running at a Time Increment from System Image Time

```
RUN taskname time [/keyword(s)]
```

Keywords

/RSI=time
/UIC=[uic]

Parameters

taskname

Specifies the name of the task. A task name consists of one to six characters.

time

Specifies the number of time units added to the current system image time to determine the time at which the task will run. When you specify this parameter with the /RSI keyword, it specifies the number of time units added to the time of the task's last completed run to determine the time at which the task will be run again.

The time parameter has the following format:

```
magu
```

mag

Specifies the magnitude or number of time units to be clocked. The maximum number of time units you can specify depends on the type of time units, as follows:

Hours	Specifies a positive number from 0 to 24_{10}.
Minutes	Specifies a positive number from 0 to 1440_{10}.
Seconds	Specifies a positive number from 0 to $32{,}767_{10}$.
Clock ticks	Specifies a positive number from 0 to $32{,}767_{10}$.

If you are specifying a reschedule interval with the /RSI keyword and you specify a magnitude of 0, the magnitude defaults to 1. For example, specifying /RSI=0H is the same as specifying /RSI=1H.

The total time cannot exceed 24 hours. The magnitude is decimal by default.

u

Specifies the type of time units to be clocked. Use single letters to specify the following time units:

Unit	Reschedule Interval
T	Clock ticks
S	Seconds
M	Minutes
H	Hours

A *tick* is a clock interrupt. The rate at which interrupts occur depends on the type of clock installed in the system. For a line frequency clock, the tick rate is either 50 or 60 ticks per second, corresponding to the line frequency. For a programmable clock, a maximum of 1000 ticks per second is available. (You can select the frequency during system generation. For more information, see the *RSX-11M-PLUS System Generation and Installation Guide.*)

Keyword Descriptions

/RSI=time

Specifies the reschedule interval. The reschedule interval indicates to the system when the task is to be run again. Each time the interval of time you specify lapses, the system issues an initiation request for the task. The format for time is as follows:

```
magu
```

For information on specifying magu, see the preceding description of the time parameter.

The default is no rescheduling.

/UIC=[uic]

Specifies the User Identification Code (UIC) for the task. The task's UIC determines the file protection class (system,owner,group,world) to which the task belongs. The brackets are part of the required syntax.

The default is the UIC established for pseudo device CO:.

Format 2: Running at a Time Increment from Clock Synchronization

```
RUN taskname sync [time][/keyword(s)]
```

Keywords

/RSI=time
/UIC=[uic]

See Format 1 for a description of the keywords.

Parameters

taskname

Specifies the name of the task. A task name consists of one to six characters.

sync

Indicates the synchronization unit. You can specify the following synchronization values:

Unit	Synchronization
T	Synchronizes on the next tick.
S	Synchronizes on the next second.
M	Synchronizes on the next minute.
H	Synchronizes on the next hour.

time

Specifies that the time increment value, if present, is added to the synchronization value to produce the time the task will be run. The format for specifying time is described in Format 1.

When you specify this format, the system determines the task run time by waiting for the specified time unit (sync) and then by waiting for the specified time increment (time) to elapse.

Format 3: Running at an Absolute Time of Day

```
RUN taskname hh:mm:ss [/keyword(s)]
```

Keywords

/RSI=time
/UIC=[uic]

These keywords are described under Format 1.

Parameters

taskname

Specifies the name of the task. A task name consists of one to six characters.

hh:mm:ss

Specifies the absolute time of day at which the task will be run. Replace the letters in the format with decimal numbers and indicate the following time units:

hh	Specifies hours.
mm	Specifies minutes.
ss	Specifies seconds.

Format 4: Running at a Clock Tick after System Bootstrap

```
RUN taskname [/keyword(s)]
```

Keywords

/RSI=time
/UIC=[uic]

These keywords are described under Format 1.

Parameter

taskname

Specifies the name of the task. A task name consists of one to six characters.

Examples

```
VMR>
```

Runs task XKE 15 minutes from the current system image time.

```
VMR>
```

Runs task XKE 15 minutes from the current system image time, rescheduling it every 90 seconds. The task will run under UIC [3,1].

```
VMR>
```

Synchronizes with the system image time on the next hour, runs the task XKE 5 minutes after synchronization, and reschedules the task to run every hour after the last completed run.

```
VMR>
```

Runs task XKE at 5 minutes before midnight.

```
VMR>
```

Runs task XKE one clock tick after the system bootstrap and reschedules the task to run every 2 minutes after that.

5.5.15 SAVE

Use the SAVE (SAV) command to write the image of a system onto a sequential medium in bootable format. You can then transport the magnetic tape or diskette to another computer and bootstrap it by using a hardware bootstrap. (Refer to the description of the /IM switch in Section 5.4.)

On paper tape, system images are created in absolute loader format and must be bootstrapped with the absolute loader.

Format

```
SAV[E] ddnn:[label][/keyword(s)]
```

Keywords

/BOOT=type
/DENS=1600

Parameters

ddnn:

Specifies the device unit onto which the system image is written. The SAVE command accepts the following devices:

Mnemonic	Device
CT	Tape cassette
DD	TU58 DECtape II
DT	DECtape
DU	RX33/RX50 diskette, RA60 disk, and RC25 disk (removable)
DX	RX01 diskette
DY	RX02 diskette
MM	TE16/TU16/TU45/TU77 magnetic tape
MS	TS11/TSV05/TU80 magnetic tape
MT	TE10/TU10/TS03 magnetic tape
MU	TK50 cartridge tape and TU81 magnetic tape
PP	Paper tape

label

Specifies the file label name that must be specified for all devices except paper tape. The label name can be 1 to 12 alphanumeric characters.

Keyword Descriptions

/BOOT=type:

Specifies the type of boot block being used when the system image is saved. The options for type are DX and DY (for RX01s and RX02s, respectively).

The /BOOT keyword allows you to use an RX01 drive to bootstrap a system that was saved using an RX02 drive in single-density mode. Similarly, /BOOT allows you to use an RX02 drive to bootstrap a system that was saved using an RX01 drive in single-density mode.

If you do not use /BOOT, the boot block for the type of device being used is written to the device.

/DENS=1600

Specifies the bits per inch (bpi) density for MM-type magnetic tapes (such as TU77).

Notes

1. When you bootstrap systems on media other than paper tape, regardless of the size of the system image, the last 1000_8 bytes of physical memory are not copied to accommodate the bootstrap.
2. When you save a system on paper tape, contiguous blocks that contain only zeros and begin on a 32-word address are compressed into two words that contain the zero count. This compression significantly reduces both the amount of paper tape required and the time expended to create the tape and read it into memory. After the entire image has been loaded into memory, a short routine writes zeros into the appropriate blocks of memory before passing control to the bootstrapped system.
3. Systems saved on a 9-track magnetic tape drive can be bootstrapped from a drive capable of bootstrapping that tape density.
4. When you specify the /IM switch with the system file specification, the SAVE command always uses a bootstrap that does not enable memory management.
5. Unless you specify /DENS=1600, when you save a system image to a magnetic tape drive that supports multiple densities, the drive is forced to 6250 bpi (for TU81 drives) or 800 bpi (for all other drives) before the system starts the SAVE operation.

Example

```
VMR>
```

Writes the system image onto the MT device in bootable format. System changes made by other VMR commands are also saved with the system image.

5.5.16 SET

The SET command allows you to do the following:

- Set and display device characteristics for devices.
- Set and display default UICs for terminals.
- Set and display system UIC.
- Set and display terminal status.
- Create and eliminate partitions.
- Add space to the system dynamic memory.
- Enable write-checking and seek optimization on devices that support it.

When you use the SET command, a situation may arise in which the system fails to accept privileged user input. To prevent this from happening, be sure the system always has at least one privileged terminal (for entering privileged commands) and one nonslaved terminal (for entering unsolicited input to MCR).

VMR permits only one keyword per command line. You can prefix most keywords by NO to negate or disable the function of the keyword. VMR returns a syntax error message when NO precedes a keyword for which it is invalid.

Examples of the SET command follow the keyword descriptions.

Format

```
SET /keyword[=value]
```

Table 5–2 briefly summarizes the SET command keywords. They are grouped according to the functions they perform, as follows:

- Setting device characteristics
- Establishing directories
- Modifying memory allocation
- Setting pool limits
- Ensuring system protection
- Tuning the system

Table 5-2: SET Keyword Summary

Keyword	Description
Setting Device Characteristics	
/[NO]ABAUD[=ttnn:]	Enables or disables autobaud detection for a terminal.
/[NO]ANSI[=ttnn:]	Specifies support for American National Standards Institute (ANSI) standard escape sequences.
/[NO]AVO[=ttnn:]	Specifies the advanced video option for a terminal.
/[NO]BLKMOD[=ttnn:]	Enables or disables local editing and block-mode transmission.
/[NO]BRO[=ttnn:]	Enables or disables the broadcast option.
/BUF=ddnn:[size]	Sets or displays the default buffer size.
/CHAR_LENGTH=ttnn:[size]	Sets or displays the length of characters transmitted and received.
/[NO]CRT[=ttnn:]	Enables or disables the backwards deletion option.
/[NO]DEC[=ttnn:]	Indicates compatibility with VT100-series terminals.
/[NO]EBC[=ttnn:]	Enables or disables the 8-bit character option.
/[NO]ECHO[=ttnn:]	Displays or inhibits the display of input characters on a terminal.
/[NO]EDIT[=ttnn:]	Specifies performance of ANSI-defined advanced editing functions.
/[NO]ESCSEQ[=ttnn:]	Enables recognition of escape sequences.
/[NO]FDX[=ttnn:]	Enables or disables simultaneous processing of input and output (the full-duplex option).
/[NO]FORMFEED[=ttnn:]	Indicates whether or not a terminal has a hardware form feed.
/HFILL=ttnn:[value]	Specifies the number of fill characters placed after a carriage return.
/[NO]HHT[=ttnn:]	Indicates whether or not a terminal has a hardware horizontal tab.
/[NO]HSYNC[=ttnn:]	Enables or disables sychronization between the host system and the specified terminal.
/LINES=ttnn:[value]	Establishes the number of lines per page for a terminal.
/[NO]LOWER[=ttnn:]	Enables or disables the conversion of lowercase input characters to uppercase.
/[NO]OPT=[ddnn:opttyp]	Enables or disables I/O queue optimization.

Table 5-2 (Cont.): SET Keyword Summary

Keyword	Description
Setting Device Characteristics	
/[NO]PARITY[=ttnn:[option]]	Enables or disables parity generation and checking for a terminal.
/[NO]PASTHRU[=ttnn:]	Enables or disables standard operating system responses to special characters (such as CTRL/C or CTRL/O).
/[NO]PRINTER_PORT[=ttnn:]	Indicates that a terminal does or does not have a printer port.
/[NO]PRIV[=ttnn:]	Establishes a terminal as privileged or nonprivileged.
/[NO]PUB[=ddnn:]	Establishes a device as public or nonpublic.
/[NO]REGIS[=ttnn:]	Indicates that a terminal does or does not support the ReGIS graphics character set.
/[NO]REMOTE[=ttnn::[speed]]	Declares that at terminal line is remote (connected to a modem) or local.
/[NO]RPA[=ttnn:]	Enables or disables the read-pass-all option.
/[NO]SERIAL[=ttnn:]	Specifies serial or parallel processing of unsolicited input.
/[NO]SLAVE[=ttnn:]	Establishes or removes slave status from a terminal.
/[NO]SOFT[=ttnn:]	Specifies whether or not a terminal accepts software-defined character sets.
/SPEED=ttnn:[recv:xmit]	Establishes the receive and transmit baud rate.
/TERM=ttnn:[type]	Establishes the terminal type.
/[NO]TTSYNC[=ttnn:]	Enables or disables the terminal synchronization option.
/[NO]TYPEAHEAD[=ttnn:[size]]	Specifies whether or not a terminal driver stores input characters in a buffer to prevent their loss.
/UIC[=[uic][:ttnn]]	Establishes the default UIC.
/[NO]VFILL[=ttnn:]	Enables or disables the vertical fill characters option.
/[NO]WCHK[=ddnn:]	Enables or disables write-checking.
/[NO]WRAP[=ttnn:]	Enables or disables the wraparound option.

Table 5–2 (Cont.): SET Keyword Summary

Keyword	Description
Establishing Directories	
/LIBUIC[=[uic]]	Establishes the UIC for nonprivileged system task library files.
/NETUIC[=[uic]]	Specifies the UIC in which all DECnet-related tasks are stored.
/SYSUIC[=[uic]]	Establishes the UIC for the system and all system tasks.
Modifying Memory Allocation	
/AFF=[UNIBUS-run-mask, . . .]	Specifies which I/O page contains the device partition. This keyword is supported on multiprocessor systems only.
/BOT=pname:value	Directs VMR to move the bottom boundary of the specified partition (pname) up or down.
/MAXEXT[=size]	Establishes the maximum size to which a task can extend itself.
/PAR=pname[:base:size[:type]]	Establishes or displays a partition.
/NOPAR=pname	Eliminates a partition from the system.
/POOL[=top]	Specifies the top of pool.
/POOLSIZE[=value]	Increases the total size of pool.
/SECPOL	Displays secondary pool use on the system.
/TOP=pname:value	Modifies the top boundary of a partition.
Setting Pool Limits	
/PLCTL[=[high][:[low][:[frz][:[basep]]]]	Sets the pool limit parameters used by the Pool Monitor Task (PMT).
Ensuring System Protection	
/[NO]LOGON	Enables or disables logging in on the system.

Table 5–2 (Cont.): SET Keyword Summary

Keyword	Description
	Tuning the System
/RNDC[=nn]	Defines the length of the Executive round-robin scheduling in ticks.
/RNDH[=nn]	Defines the highest priority considered for Executive round-robin scheduling.
/RNDL[=nn]	Defines the lowest priority considered for Executive round-robin scheduling.
/SWPC[=nn]	Defines the number of clock ticks for a single Executive swapping interval.
/SWPR[=nn]	Defines a priority range for Executive swapping.

Notes on Memory Allocation Keywords

1. If a command attempts to eliminate a partition in which tasks are installed or drivers loaded, VMR rejects the command until the tasks have been removed.
2. When you define a partition, the name must not already be defined as a partition. In addition, a partition cannot overlap any other partition.
3. The address specified with the /POOL keyword must be lower than the base address of the first partition in the system.
4. Once you have allocated space to the pool, you cannot recover the space for use in partitions.
5. When the highest memory addresses of a system are allocated, the size of the partition is modified automatically to reflect the amount of memory present when the system is bootstrapped.
6. The top address of the last partition is modified by the SAVE command at boot time to reflect the actual top of physical memory of the hardware in use.

Keyword Descriptions

/ABAUD[=ttnn:]

Enables autobaud detection for the remote dial-up line for the specified terminal. The terminal driver samples the line's first input character, tries to determine the incoming speed, and sets the interface speed accordingly.

When you omit the terminal specification, VMR displays all the terminals with autobaud detection.

/NOABAUD[=ttnn:]

Disables autobaud detection for the specified terminal. The terminal driver does not attempt to determine the incoming speed for the remote dial-up line.

When you omit the terminal specification, VMR displays all the terminals that do not have autobaud detection.

/AFF=[UNIBUS-run-mask, . . .]

Specifies which I/O page will contain the device partition. /AFF is required for creating device partitions on the system.

This keyword is supported on RSX–11M–PLUS multiprocessor systems only.

[UNIBUS-run-mask, . . .]

Specifies the 3-letter name of the UNIBUS Run Mask (URM) with which the device partition is associated. Use one of the following names:

Valid UNIBUS Run Mask (URM) Names

CPA	UBE	UBK	UBP
CPB	UBF	UBL	UBR
CPC	UBH	UBM	UBS
CPD	UBJ	UBN	UBT

For example, to create a device partition that is mapped to the I/O page for CPA, enter the following command line:

Specify at least one URM. If you specify more than one URM, separate them with commas. The brackets are part of the required syntax.

/ANSI[=ttnn:]

Informs the system that the specified terminal supports ANSI escape sequences. When you omit the terminal specification, VMR displays all the terminals that have the ANSI character set.

/NOANSI[=ttnn:]

Informs the system that the specified terminal does not support ANSI escape sequences. When you omit the terminal specification, VMR displays all terminals that do not have the ANSI character set.

/AVO[=ttnn:]

Informs the system that the specified VT100-series terminal has the advanced video option. This option gives the terminal the capability to blink, perform bolding, and flash parts of the screen, and to set the screen at 132_{10} columns. If the word SET-UP blinks in set-up mode, the terminal has the advanced video option.

When you omit the terminal specification, VMR displays all the terminals that have the advanced video option.

/NOAVO[=ttnn:]

Informs the system that the specified terminal does not have the advanced video option. When you omit the terminal specification, VMR displays all terminals that do not have the advanced video option.

/BLKMOD[=ttnn:]

Enables local editing and block-mode transmission for the specified terminal. When you omit the terminal specification, VMR displays all the terminals that have the block-mode option enabled.

/NOBLKMOD[=ttnn:]

Disables local editing and block-mode transmission for the specified terminal. When you omit the terminal specification, VMR displays all the terminals that have the block-mode option disabled.

/BOT=pname:value

Directs VMR to move the bottom boundary of the specified partition (pname) up or down the amount indicated by the value parameter. The /BOT keyword modifies the bottom boundary of a partition even though it has tasks installed in it. If you modify the partition so that it becomes too small to hold the tasks installed in it, VMR displays a warning message but modifies the partition anyway.

Specify values for pname and value, as follows:

pname

Specifies the 1- to 6-character alphanumeric partition name.

value

Specifies the number of 64-byte blocks.

The value parameter can be supplied in five formats: +value, –value, value, +* or *, and –*, as follows:

Format	Action
+value	Moves the partition's bottom boundary up the amount indicated by the specified value. Moving the bottom boundary up makes the partition smaller. You must supply the value in the same units as those of the base and size parameters for the SET keyword /PAR.
–value	Moves the partition's bottom boundary down the amount indicated by the specified value. Moving the bottom boundary down makes the partition larger.
value	Moves the partition's bottom boundary until the total partition size is equal to the specified value.

Format	Action
+* or *	Moves the partition's bottom boundary up as far as possible. Note that using this format will cause the bottom boundary to move up to the bottom of the first fixed task or fixed driver loaded into the partition at the time the /BOT keyword is executed. If there is nothing loaded into the partition, it assumes a size of one 64-byte block.
-*	Moves the partition's bottom boundary down as far as possible. Using this format will cause the bottom boundary of the partition to move down to the top boundary of the partition below it, or to the top of the Executive pool if the partition is the lowest partition in memory.

/BRO[=ttnn:]

Enables the broadcast option for the specified terminal. The terminal can receive messages sent from other terminals with the MCR command BROADCAST.

When you omit the terminal specification, VMR displays all the terminals that can receive messages.

/NOBRO[=ttnn:]

Disables the broadcast option for the specified terminal. The terminal cannot receive messages sent from other terminals.

When you omit the terminal specification, VMR displays all the terminals that cannot receive messages.

/BUF=ddnn:[size]

Sets the default buffer size of the specified device. The size you specify must be from 0 to 255_{10}. (For a terminal, the size you specify must be greater than or equal to 15_{10}). The /BUF keyword is particularly useful for defining line printer width (for example, 80 or 132 columns).

When you omit the size specification, VMR displays the current buffer size of the device.

/CHAR_LENGTH=ttnn:[size]

Establishes the length of characters transmitted and received by terminals attached to the system through variable-speed multiplexers. However, note that the command is not valid for serial-line units.

You can specify the character length as either 7 or 8 bits. If you do not specify a length, VMR displays the current setting. The default value for transmission is 8-bit characters.

/CRT[=ttnn:]

Informs the system that the specified terminal is a cathode-ray tube (CRT) terminal and enables the backwards deletion option.

When you enable the backwards deletion option, pressing the RUBOUT or DELETE key moves the printing position one space to the left and erases any character displayed in that position.

When you omit the terminal specification, VMR displays all the terminals that are defined as CRTs.

/NOCRT[=ttnn:]

Informs the system that the specified terminal is not a CRT and disables the backwards deletion option. When you omit the terminal specification, VMR displays all the terminals that are not defined as CRTs.

/DEC[=ttnn:]

Indicates that the specified terminal is upward compatible with the VT100-series of terminals. When you omit the terminal specification, VMR displays all the terminals that are upward compatible.

/NODEC[=ttnn:]

Informs the system that the specified terminal is not upward compatible with the VT100-series of terminals. When you omit the terminal specification, VMR displays all the terminals that are not upward compatible.

/EBC[=ttnn:]

Enables the 8-bit character option. The terminal driver passes all 8 bits of the characters to the specified terminal. When you omit the terminal specification, VMR displays all the terminals that have the 8-bit character option enabled.

/NOEBC[=ttnn:]

Disables the 8-bit character option. The terminal driver clears the eighth bit of the characters before passing them to the device. When you omit the terminal specification, VMR displays all the terminals that do not have the 8-bit character option enabled.

/ECHO[=ttnn:]

Enables the echo option. The terminal driver displays each character that is typed on the specified terminal. When you omit the terminal specification, VMR displays all the terminals that have the echo option enabled.

/NOECHO[=ttnn:]

Disables the echo option. The terminal driver does not display the characters that are typed on the specified terminal. When you omit the terminal specification, VMR displays all the terminals that have the echo option disabled.

/EDIT[=ttnn:]

Informs the system that the specified terminal can perform ANSI-defined advanced editing functions. The VT102 is an example of such a terminal. When you omit the terminal specification, VMR displays all the terminals that can perform the advanced editing functions.

/NOEDIT[=ttnn:]

Informs the system that the specified terminal cannot perform ANSI-defined advanced editing functions. When you omit the terminal specification, VMR displays all the terminals that cannot perform the advanced editing functions.

/ESCSEQ[=ttnn:]

Enables the recognition of escape sequences from the specified terminal. The terminal driver treats the ESC character as the beginning of an escape sequence rather than as a line terminator. (See the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual* for more information on escape sequences.)

When you omit the terminal specification, VMR displays all the terminals that recognize escape sequences.

/NOESCSEQ[=ttnn:]

Disables the recognition of escape sequences at the specified terminal. When you omit the terminal specification, VMR displays all the terminals that do not support the recognition of escape sequences.

/FDX[=ttnn:]

Enables the full-duplex option. The terminal driver accepts input from the specified terminal while simultaneously outputting to it. When you omit the terminal specification, VMR displays all the terminals that have the full-duplex option enabled.

/NOFDX[=ttnn:]

Disables the full-duplex option. The terminal driver will not process input and output from the specified terminal simultaneously. When you omit the terminal specification, VMR displays all the terminals that do not have the full-duplex option enabled.

/FORMFEED[=ttnn:]

Informs the system that the specified terminal supports a hardware form feed. When you omit the terminal specification, VMR displays all the terminals that support hardware form feeds.

/NOFORMFEED[=ttnn:]

Informs the system that the specified terminal does not support a hardware form feed. When you omit the terminal specification, VMR displays all the terminals on the system that do not support hardware form feeds.

/HFILL=ttnn:[value]

Specifies the number of fill characters (value) that the terminal driver is to place after a carriage return when sending output to the specified terminal. The value is a number from 0 to 7.

When you omit the value parameter, VMR displays the number of fill characters currently being output by the terminal driver to the specified terminal.

/HHT[=ttnn:]

Indicates that the specified terminal supports a hardware horizontal tab. When you omit the terminal specification, VMR displays all the terminals on the system that support a hardware horizontal tab.

/NOHHT[=ttnn:]

Indicates that the specified terminal does not support a hardware horizontal tab. When you omit the terminal specification, VMR displays all the terminals that do not support a hardware horizontal tab.

/HSYNC[=ttnn:]

Enables host-terminal synchronization. *Host-terminal synchronization* controls the type-ahead buffer to prevent it from overflowing. When the type-ahead buffer is nearly full, the system temporarily locks the terminal keyboard by generating a CTRL/S sequence. CTRL/S prevents you from entering more information. After the terminal transmits all of the information in the type-ahead buffer to the operating system, the system unlocks the keyboard by generating a CTRL/Q, which allows the terminal to receive input.

The /HSYNC keyword is recommended for users with terminals that transmit information in blocks, for example, the DIGITAL VT131 and VT132 terminals. It is also recommended for users with terminals that emulate these models. If you do not have one of these terminals it is unlikely that you will fill the type-ahead buffer. To change the size of your type-ahead buffer, use the /TYPEAHEAD keyword.

/HSYNC is the default for the VT100-series of terminals. When you omit the terminal specification, VMR displays all the terminals that have host-terminal synchronization.

For more information on type-ahead buffers, see the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual.*

/NOHSYNC[=ttnn:]

Disables host-terminal synchronization for the specified terminal. Disabling synchronization means that when the type-ahead buffer for the terminal is nearly full, the system does not lock the keyboard, and subsequent input is lost.

/NOHSYNC is the default for VT52 terminals. When you omit the terminal specification, VMR displays all the terminals that do not have host-terminal synchronization.

For more information on type-ahead buffers, see the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual.*

/LIBUIC[=[uic]]

Directs the MCR command INSTALL to search the specified User Identification Code (UIC) in response to a RUN $file or INSTALL $file command.

The UIC is in the format [g,m], where g and m are octal numbers from 1 to 377_8 that represent a group and member number, respectively. The brackets are part of the required syntax.

When you omit the uic specification, VMR displays the current library UIC. The default library UIC is [3,54].

/LINES=ttnn:[value]

Establishes the number of lines per page on the specified terminal. The value is a number from 1 to 255_{10}. If you specify a number that is larger than the number of lines per page supported by the specified terminal type, VMR defaults to the largest possible value.

When you omit the value parameter, VMR displays the lines per page for the specified terminal.

/LOGON

Allows users to log in to the system. /LOGON and /NOLOGON clear and set a flag checked by the HELLO command when a user attempts to log in on a terminal.

/NOLOGON

Prevents users from logging in to the system. /NOLOGON provides a means of preventing users from accessing the system during such activities as system maintenance or account file maintenance.

/LOWER[=ttnn:]

Specifies that lowercase characters are not converted to uppercase when received from the specified terminal. When you omit the terminal specification, VMR displays all the terminals for which lowercase characters are not converted.

/NOLOWER[=ttnn:]

Resets the terminal characteristics of the specified terminal so that lowercase characters received on input are automatically converted to uppercase and echoed as uppercase. When you omit the terminal specification, VMR displays all the terminals for which conversion to uppercase is specified.

/MAXEXT[=size]

Establishes the maximum size to which a task can extend itself beyond its mapped array area by using the Executive directive Extend Task (EXTK$). (See the *RSX-11M-PLUS and Micro/RSX Executive Reference Manual* for a description of the directive.) The size is given in units of 64_{10}-byte blocks. The maximum size allowed is $177,777_8$ blocks.

Specify the size in one of the following formats:

Specified Size	Resulting Size
n	(n_8*100_8)
n.	$(n_{10}*64_{10})$
nK	(n_8*4000_8)
n.K	$(n_{10}*2048_{10})$
*	177777_8

When you omit the size specification, VMR displays the current maximum extension size in 64_{10}-byte units.

/NETUIC[=[uic]]

Specifies the UIC in which all DECnet-related tasks are stored. The /NETUIC keyword applies only to systems that select the external communication products option during system generation.

The UIC is in the format [g,m], where g and m are octal numbers from 1 to 377_8 that represent a group and member number, respectively. The square brackets are part of the required syntax.

When you omit the uic specification, VMR displays the current network UIC.

/OPT[=ddnn:opttyp]

Enables disk I/O queue optimization for the specified device. For more information on optimization and on the algorithm parameter for this keyword (opttype), see Chapter 14.

Specify the device and optimization algorithm, as follows:

ddnn:

Specifies the device that will use I/O queue optimization. Valid devices are DB, DL, DM, and DR devices. If you do not specify a device, VMR displays all the devices that use optimization.

opttyp

Specifies the algorithm for optimization. The algorithms are as follows:

NEAREST
ELEVATOR
CSCAN

The default algorithm set at system generation is NEAREST.

When you omit ddnn: and opttyp, VMR displays all the devices that have I/O queue optimization enabled.

/NOOPT[=ddnn:]

Disables disk I/O queue optimization for the specified device. When you omit the device specification, VMR displays all the devices that do not use I/O queue optimization.

/PAR=pname[:base:size[:type]]

Establishes a partition in memory.

pname

Specifies the 1- to 6-character alphanumeric partition name.

base

Specifies the physical base address of the partition specified as a number of 64-byte blocks or as a wildcard (*).

size

Specifies the size of the partition in 64-byte blocks or as a wildcard (*).

The size can range from 1 to $170{,}000_8$ (1920K-words), minus the size of the Executive and other partitions in the system.

type

Specifies one of the following partition types:

Type	Function
SYS	Establishes a main partition, which is allocated for user tasks and loadable drivers.
SECPOL	Establishes a secondary pool partition, which is allocated for secondary pool.
DEV	Establishes a common partition that maps the device registers. Device partitions are allocated for device commons that tasks use to access the device registers on the I/O page. Note that creating a device partition automatically creates a main partition named IO PAR, which spans the entire I/O page. The partition you specify with the /PAR keyword then becomes a subpartition of IO PAR. On multiprocessor systems, use the /AFF keyword to specify which I/O page contains the device partition.
CPU	Establishes a central processing unit (CPU) partition, which is allocated on multiprocessor systems to store vectors for all the devices associated with the specified processor.

The partition type is optional. However, if you do not specify a partition type, SYS is the default partition type assigned.

You can enter the base and size arguments as wildcards (*) or in one of the following formats:

Format	Calculated Value
nnnn	($nnnn_8$*100_8)
nnnn.	($nnnn_{10}$*64_{10})
nnnK	(nnn_8*4000_8)
nnn.K	(nnn_{10}*2048_{10})

For SYS partitions, the size can range from 0 to 1920K or any valid representation of these limits. For example, the following values for size allocate a 2048_{10}-byte partition:

If you specify the base address of a partition as a wildcard (*), VMR creates a partition at the lowest address possible for a partition of the size you specify. If you specify the size of a partition with a wildcard, VMR creates the largest possible partition starting at the base address you specify. If you specify wildcards for both the base and size, VMR creates a partition that completely fills the first gap in the existing partition layout.

If you specify a wildcard for a CPU partition, VMR creates a partition of size (n-1)*4K, where n is the number of processors specified during system generation. Note that when you establish the base and size for a CPU partition, the entire CPU partition must be below 124K.

If you do not specify base, size, or type, VMR displays the values of these parameters for the named partition. If the named partition is a main partition, VMR also displays the parameters of all its subpartitions.

/NOPAR=pname

Eliminates a named partition from the system. The partition you specify is eliminated from the list of partitions and the system. If any tasks or commons are installed in or attached to the partition, the partition is not eliminated, and VMR generates an error message.

/PARITY[=ttnn:[option]]

Enables parity generation and checking. The option you specify selects the state of the parity bit. If you do not enable parity checking, parity bits are not transmitted.

Parity checking is used to verify the transmission of data between a terminal and the operating system. Verification occurs by sending an extra bit (called a "parity bit") with each character, which serves as a check that the transmitted character was received properly.

The options for /PARITY are EVEN and ODD. The state of the parity bit (0 or 1) is determined by the option you specify, as follows:

/PARITY=ttnn:EVEN

Adds either a 0 or 1 to make the total number of parity bits even. This is the default.

/PARITY=ttnn:ODD

Adds either a 0 or 1 to make the total number of parity bits odd.

The system checks parity on input from a terminal. The terminal checks parity on output from the system. The parity check does not return an error under the following conditions:

- If you specify an EVEN parity and the system counts an even number of bits
- If you specify an ODD parity and the system counts an odd number of bits

Note that the terminal hardware must be set to the same mode of parity generation and checking as the system for correct data transmission to occur. On the VT200-series of terminals, this is done through the terminal's set-up mode.

Note

> The /PARITY keyword is not related to the /EBC keyword. The 8-bit character option clears the eighth data bit only.

When you omit the terminal specification, VMR displays all the terminals that have parity checking enabled.

/NOPARITY[=ttnn:]

Disables parity generation and checking for the specified terminal. Data transmitted between the terminal and the system is not verified.

When you omit the terminal specification, VMR displays all the terminals that do not have parity generation and checking enabled.

/PASTHRU[=ttnn:]

Requests the system to ignore the original function of special characters typed at the specified terminal and, instead, to pass them to a program in their binary form. If your terminal does not seem to be working properly, it may have been set to /PASTHRU.

/PASTHRU allows you to suppress standard operating system responses to special characters (for example, CTRL/C and CTRL/O), thereby controlling output to the terminal screen. While /PASTHRU is enabled, the terminal does not recognize any control characters, except CTRL/S and CTRL/Q if it is set to /TTSYNC, nor does it respond to carriage-return or line-feed characters.

When you omit the terminal specification, VMR displays all the terminals that have been set to /PASTHRU.

/NOPASTHRU[=ttnn:]

Requests the system to recognize the original function of special characters typed at the specified terminal. When you omit the terminal specification, VMR displays all the terminals that have not been set to /PASTHRU.

/NOPASTHRU is the default.

/PLCTL[=[high][:[low][:[frsiz][:basep]]]]

Sets the pool limit parameters used by the Pool Monitor Task (PMT). The following list defines the parameters:

high
Specifies the high pool limit in bytes.

low
Specifies the low pool limit in bytes.

frsiz
Specifies the minimum byte size of the largest free pool block required for avoiding low pool actions by PMT.

basep
Specifies the base task priority, which is the lowest priority a nonprivileged task can have and still be eligible for memory contention during times of low pool; this priority remains in effect until pool conditions improve. (Note that only those tasks requested to run while the system is in a low pool state are affected).

The following list defines the limits of the parameters and shows the default values:

Parameter	Minimum Limit	Maximum Limit	Default
high	Low parameter	Total size of system pool	1600_{10} bytes
low	84_{10} bytes	High parameter	600_{10} bytes
frsiz	84_{10} bytes	High parameter	200_{10} bytes
basep	0_{10} bytes	250_{10} bytes	51_{10} bytes

See Chapter 8 for more information on determining values for these parameters.

When you omit the parameter specifications, VMR displays the current parameters for the system.

/POOL[=top]

Increases the size of pool.

top

Specifies the first location in memory used for partition allocation. A wildcard (*) directs VMR to supply the maximum amount of pool possible.

Specify the wildcard as follows:

```
/POOL=*
```

If you request more pool than can be supplied, VMR prints a warning message and supplies the maximum amount of pool possible.

If you do not specify a value for top, VMR displays the virtual address of the top of the Executive, the size of the longest block of pool space in words, the total number of words in the pool, and the lowest physical address at which a partition can start. The format for the display is as follows:

```
POOL=top:max:total[:par]
```

This format displays top in units of 64-byte blocks, max and total in decimal words, and par as the lowest physical address (32-word blocks).

Note

You can use the address when calculating partition layouts for systems supporting kernel data space. If your system is not a kernel data space system, this value is always equal to the virtual address of the top of pool.

/POOLSIZE[=value]

Increases the size of pool. The value you specify indicates the total amount of pool to be allocated.

If you do not specify a value, VMR displays the virtual address of the top of the Executive, the size of the longest block of pool space in words, the total number of words in the pool, and the lowest physical address at which a partition can start in 32-word blocks.

The /POOL and /POOLSIZE keywords perform the same function. /POOL allows you to specify the top of pool. /POOLSIZE allows you to specify the total size of pool.

/PRINTER_PORT[=ttnn:]

Informs the system that the specified terminal has a printer port. The VT200-series of terminals, for example, has printer ports. When you omit the terminal specification, VMR displays all the terminals that have a printer port.

/NOPRINTER_PORT[=ttnn:]

Informs the system that the specified terminal does not have a printer port. When you omit the terminal specification, VMR displays all the terminals that do not have a printer port.

/PRIV[=ttnn:]

Sets the specified terminal to privileged status. When you omit the terminal specification, VMR displays all the privileged terminals in the system.

/NOPRIV[=ttnn:]

Sets the specified terminal to nonprivileged status. When you omit the terminal specification, VMR displays all the nonprivileged terminals in the system.

/PUB[=ddnn:]

Establishes the specified device as a public device. When you omit the device specification, VMR displays all the public devices in the system.

/NOPUB[=ddnn:]

Causes the specified device to lose its public status. When you omit the device specification, VMR displays all the nonpublic devices in the system.

/REGIS[=ttnn:]

Informs the system that the specified terminal supports the ReGIS graphics set. When you omit the terminal specification, VMR displays all the terminals that support the ReGIS graphics set.

The VT125 and the VT240 terminals both support the ReGIS graphics set.

/NOREGIS[=ttnn:]

Informs the system that the specified terminal does not support the ReGIS graphics set. When you omit the terminal specification, VMR displays all the terminals that do not support the ReGIS graphics set.

/REMOTE[=ttnn:[speed]]

Informs the system that the specified terminal is connected to a modem and can be connected to the system by means of a dial-up network. The terminal has a line to a multiplexer that supports remote lines (such as a DHU11).

Specifying speed establishes the initial speed (answer speed) of the remote dial-up line for the specified terminal. You cannot specify an answer speed for a DL11 or DLV11 multiplexer.

Table 5–3 lists the valid baud rates for various multiplexers.[1]

[1] The DHQ11, CXA16, and CXB16 multiplexers do not support remote lines.

Table 5–3: Valid Baud Rates for Variable-Speed Multiplexers

Multiplexer	Valid Baud Rates			
DH11	0	134.5	600	4800
	50	150	1200	9600
	75	200	1800	EXTA (user-specified A)
	110	300	2400	EXTB (user-specified B)
DHU11/DHV11/DHQ11	75	300	2000	19200
	110	600	2400	
	134.5	1200	4800	
	150	1800	9600	
DZ11/DZQ11/DZV11	50	150	1800	4800
	75	300	2000	7200
	110	600	2400	9600
	134.5	1200	3600	
CXY08/CXA16/CXB16	75	300	2000	19200
	110	600	2400	
	134.5	1200	4800	
	150	1800	9600	

When you omit the terminal specification, VMR displays all the remote terminals. When you omit the speed specification, the current setting for the terminal remains in effect.

/NOREMOTE[=ttnn:]

Establishes the terminal's line as a local line that is not connected to a modem. When you omit the terminal specification, VMR displays all the local terminals.

/RNDC[=nn]

Defines the length of the Executive round-robin scheduling interval in ticks. (For more information on scheduling, see Chapter 1.)

The length value nn is assumed to be octal, unless you place a period after the value. The minimum value for nn is 0. When you omit the length specification, VMR displays the current value for the interval.

/RNDH[=nn]

Defines the highest priority that is considered for Executive round-robin scheduling. For more information on scheduling, see Chapter 1.

Specify a value for nn between 1 and 250_{10}, where 250_{10} is the highest priority and 1 is the lowest priority. For example, a priority of 200_{10} takes precedence over a priority of 199_{10}.

The priority you enter is assumed to be octal unless you place a period after the number. Also, it must be higher than the one specified with the /RNDL keyword. If you do not specify a priority, VMR displays the current value for the priority.

/RNDL[=nn]

Defines the lowest priority, from 1 to 250, that will be considered for Executive round-robin scheduling. 250_{10} is the highest priority and 1 is the lowest priority you can assign. Therefore, a priority of 200_{10} takes precedence over a priority of 199_{10}.

The priority you enter is assumed to be octal unless you place a period after the number.

The priority class must be lower than the one specified with the /RNDH keyword. (For more information, see Chapter 1.)

When you omit the priority class, VMR displays the current value for the priority.

/RPA[=ttnn:]

Enables the read-pass-all option. The terminal driver passes all characters input at the keyboard to the terminal's input buffer. When you omit the terminal specification, VMR displays all the terminals on the system that have the read-pass-all option enabled.

/NORPA[=ttnn:]

Disables the read-pass-all option. The terminal driver does not pass special function characters (for example, CTRL/C) input at the keyboard to the terminal's input buffer. When you omit the terminal specification, VMR displays all the terminals on the system that have the read-pass-all option disabled.

/SECPOL

Displays secondary pool use in the system. The display is in the following format:

```
SECPOL=secfr:secsiz:pctfr
```

secfr — Specifies the number of free blocks in secondary pool in units of 32_{10}-word blocks.

secsiz — Specifies the size of secondary pool in units of 32_{10}-word blocks.

pctfr — Specifies the percentage of free blocks in secondary pool.

/SERIAL[=ttnn:]

Causes unsolicited input from the specified terminal to be processed serially, rather than in parallel.

Usually, if you enter a second command line before the first command line has completed executing, the system processes the second command line in parallel with the first. When a terminal is set to serial processing, however, the system waits until the first command line is finished processing before starting the second.

When you omit the terminal specification, VMR displays all the terminals that process serially.

The default is /NOSERIAL.

/NOSERIAL[=ttnn:]

Causes unsolicited input from the specified terminal to be processed in parallel rather than serially. Parallel processing means that one or more command lines are processed at the same time. This is the default.

When you omit the terminal specification, VMR displays all the terminals that process in parallel.

/SLAVE[=ttnn:]

Establishes the specified terminal as one that can enter data only if it is solicited by a task. Therefore, the specified terminal always rejects unsolicited input (other than CTRL/O, CTRL/Q, and CTRL/S).

When you omit the terminal specification, VMR displays all the terminals currently classified as slaved.

/NOSLAVE[=ttnn:]

Sets the specified terminal to nonslaved status. When you omit the terminal specification, VMR displays all the terminals currently classified as nonslaved.

/SOFT[=ttnn:]

Informs the system that the specified terminal can accept software-defined character sets. The VT200-series of terminals can accept these character sets. When you omit the terminal specification, VMR displays all the terminals that accept software-defined character sets.

/NOSOFT[=ttnn:]

Informs the system that the specified terminal cannot accept software-defined character sets. When you omit the terminal specification, VMR displays all the terminals that do not accept software-defined character sets.

/SPEED=ttnn:[recv:xmit]

Establishes the receive and transmit speed for terminals attached to the system through a variable-speed multiplexer. The values are as follows:

recv

Specifies the speed at which characters are input to the computer from the terminal.

xmit

Specifies the speed at which characters are output to the terminal from the computer.

You must specify both the recv and xmit values when setting the speed. If you do not specify either value, VMR displays the current settings.

For a list of valid baud rates, see Table 5–3. If you have a multiplexer that does not support split speeds, the receive and transmit speeds must be the same.

/SWPC[=nn]

Defines the number of clock ticks for a single Executive swapping interval. (For a description of swapping, see Chapter 1.)

The number of clock ticks nn is in the range 0 to $45{,}568_{10}$. The number of clock ticks is assumed to be octal unless you place a period after the number.

When you omit the clock ticks specification, VMR displays the current value for the interval.

/SWPR[=nn]

Defines a priority range for Executive swapping. (For a description of swapping, see Chapter 1.)

The priority number nn is in the range 0 to 127_{10}, where 127_{10} is the highest priority and 0 is the lowest priority. For example, a priority of 100_{10} takes precedence over a priority of 99. The priority you enter is assumed to be octal unless you place a period after the priority number.

The value for the /SWPR keyword affects the installed priority of all tasks. Each task is initiated at its installed priority, plus the value for nn as assigned in /SWPR=nn (priority + nn). While the task is executing, its priority is decremented until it becomes the installed priority minus the value for nn (priority - nn). When the priority reaches this lower limit, the task is swapped out so that another task with a higher priority can execute. The priority for the new task changes in the same way.

When you omit the range specification, VMR displays the current value for the priority range.

/SYSUIC[=[uic]]

Establishes the UIC for the system and all system tasks. The UIC has the format [g,m], where g and m are octal numbers that represent a group and member number, respectively.

The UIC specified with the /SYSUIC keyword also becomes the default UIC used by the LOAD command and the install-run-remove option of the MCR command RUN.

When you omit the UIC specification, VMR displays the current system UIC (by default, [1,54]).

/TERM=ttnn:[type]

Establishes the terminal type of the specified terminal. When VMR sets the terminal type, it automatically sets the HFILL, VFILL, CRT, FORMFEED, and HHT characteristics for the terminal. The standard terminal types are as follows:

Standard Terminal Types

ASR33	LA30P	LA50	LA2xx	VT50	VT101	VT132
ASR35	LA30S	LA75	LA210	VT52	VT102	VT2xx
DTC01	LA34	LA100	LQP01/02/03	VT55	VT105	LN03
KSR33	LA36	LA120	PC3xx	VT61	VT125	
LA12	LA38	LA180S	VT05B	VT100	VT131	

If you specify a terminal type that is unknown to VMR, VMR does not set any device characteristics for the terminal. However, VMR stores the value you have specified in the Unit Control Block (UCB). When you omit the type parameter, VMR displays the terminal type of the specified terminal.

VMR also accepts an alternate form of the command. You can specify the terminal type and equate it to the desired terminal. This automatically associates certain characteristics of the particular terminal type with the specified terminal. (See the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual* for a list of the implicit characteristics for each terminal type.) The command is in the following format:

```
SET /term-type=ttnn:
```

The term-type parameter is one of the standard terminal types. Note that the terminal types are mutually exclusive.

/TOP=pname:value

Directs VMR to move the top boundary of the specified partition up or down the amount indicated by the value parameter.

The /TOP keyword modifies the top boundary of a partition even if the partition has tasks installed in it. If you modify the partition so that it becomes too small to hold the tasks installed in it, VMR displays a warning message but modifies the partition anyway.

pname

Specifies the 1- to 6-character alphanumeric partition name.

value

Specifies the number of 64-byte blocks.

The value can be supplied in five formats: +value, −value, value, +* or *, and −*, as follows:

Format	Action
+value	Moves the partition's top boundary up by the specified amount. Moving the partition's top boundary up makes the partition larger. Note that you must supply the value in the same units as the base and size parameters for the /PAR keyword.
-value	Moves the partition's top boundary down by the specified amount. Moving the partition's top boundary down makes the partition smaller.
value	Moves the partition's top boundary until the partition's total size is equal to the specified value.
+* or *	Moves the partition's top boundary up as far as possible. The most the top boundary can move up is to the bottom boundary of the partition above it, or, to the top of memory, if the partition is the last partition in memory.
-*	Moves the partition's top boundary down as far as possible. The most the top boundary can move down is to the top of the first fixed task, common, or driver in the partition at the time the /TOP keyword is executed. If the partition is available (unoccupied), it assumes a size of one 64-byte block.

/TTSYNC[=ttnn:]

Enables the terminal synchronization option for the specified terminal. This means that the system responds to the CTRL/S and CTRL/Q characters used to control the flow of information on the terminal screen, when you type them on the terminal. /TTSYNC is the default.

When you omit the terminal specification, VMR displays all the terminals that have terminal synchronization.

/NOTTSYNC[=ttnn:]

Disables the terminal synchronization option for the specified terminal. The system will not respond to the CTRL/S and CTRL/Q characters when you type them on the terminal.

If you set your terminal to /NOTTSYNC, the NO SCROLL key on VT100-series terminals and the HOLD SCREEN key on VT200-series terminals will no longer work.

When you omit the terminal specification, VMR displays all the terminals that do not have terminal synchronization.

/TYPEAHEAD[=ttnn:[size]]

Enables the type-ahead option for the specified terminal. The terminal driver stores input characters in the type-ahead buffer before passing them to the task. When the type-ahead option is enabled, characters input during periods between requests are not lost.

The size parameter applies only to operating systems that support kernel data space. It specifies the size of the type-ahead buffer, which can be between 0 and 255_{10}. However, if you specify 0 or 1, the effect is the same as /NOTYPEAHEAD; the terminal does not have the type-ahead option enabled. The default size is 66 on I- and D-space systems. On systems that do not include support for I- and D-space, the size is fixed at 58.

When you omit the terminal specification, VMR displays all the terminals that have the type-ahead option enabled.

For more information on type-ahead buffers, see the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual.*

/NOTYPEAHEAD[=ttnn:]

Disables the type-ahead option for the specified terminal. The terminal driver does not store characters to prevent their loss because the size of the buffer is forced to 1.

When you omit the terminal specification, VMR displays all the terminals that do not have the type-ahead option enabled.

For more information on type-ahead buffers, see the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual.*

/UIC[=[uic:]ttnn:]

Establishes the specified User Identification code (UIC) as the default UIC for the specified terminal (the default is the issuing terminal, TI:). A privileged user can override the terminal UIC with the /UIC keyword of the RUN command.

External MCR function tasks are requested with the /UIC keyword. These tasks are the MOUNT, INSTALL, DMO, and UFD commands (which execute as tasks), and all system-supplied software.

When you specify only the terminal, VMR displays the UIC for that terminal.

/VFILL[=ttnn:]

Enables the vertical fill characters option for the specified terminal. The terminal driver adds four fill characters following each line feed. When you omit the terminal specification, VMR displays all the terminals on the system that have the VFILL option enabled.

/NOVFILL[=ttnn:]

Disables the vertical fill characters option for the specified terminal. The terminal driver does not add any vertical fill characters following line feeds. When you omit the terminal specification, VMR displays all the terminals on the system that do not have the VFILL option enabled.

/WCHK[=ddnn:]

Specifies that all write operations to Files-11 devices are to be followed by a write-check. The checks assure the reliability of data transfers to the specified disk. When you omit the device specification, VMR displays all Files-11 disk drives with write-checking enabled.

/NOWCHK[=ddnn:]

Specifies that write-checking is to be disabled for the specified device. When you omit the device specification, VMR displays all disks that support write-checking but have the option disabled.

/WRAP[=ttnn:]

Enables the wraparound option for the specified terminal. The terminal driver automatically generates a carriage-return/line-feed combination when the number of input characters exceeds the buffer size of the specified terminal. Both the carriage return and line feed are transparent and do not appear in the input buffer.

When you omit the terminal specification, VMR displays all the terminals on the system that have the wraparound option enabled.

/NOWRAP[=ttnn:]

Disables the wraparound option for the specified terminal. The terminal driver does not accept input characters beyond the input buffer size for the terminal.

When you omit the terminal specification, VMR displays all the terminals on the system that have the wraparound option disabled.

Examples

```
VMR>
```

Enables the autobaud detection option for terminal TT4.

```
VMR>
ABAUD=TT2:
ABAUD=TT4:
ABAUD=TT5:
```

Displays all the terminals that have the autobaud detection option enabled.

```
VMR>
```

Informs the system that the VT100 terminal TT22 has the advanced video option.

```
VMR>
```

Enables local editing and block-mode transmission for the terminal TT3.

```
VMR>
```

Enables the broadcast option for terminal TT5.

```
VMR>
BRO=TT2:
BRO=TT4:
BRO=TT5:
```

Displays all the terminals that have the broadcast option enabled.

```
VMR>
BUF=LP0:132.
```

Displays the current buffer size of the hardcopy terminal LP0.

```
VMR>
```

Sets the buffer size of terminal TT1 to 40_{10}.

```
VMR>
```

Enables the backwards deletion option on the display terminal whose number is TT3.

```
VMR>
CRT=TT3:
CRT=TT7:
CRT=TT12:
```

Displays all the terminals that have the backwards deletion option enabled.

```
VMR>
```

Informs the system that terminal TT10 is not upward compatible with the VT100-series of terminals.

```
VMR>
```

Informs the system that terminal TT25 can perform ANSI-defined advanced editing functions.

```
VMR>
```

Establishes terminal TT3 as a terminal that can send and receive escape sequences.

```
VMR>
ESCSEQ=TT3:
ESCSEQ=TT7:
ESCSEQ=TT16:
```

Displays all the terminals that can send and receive escape sequences.

```
VMR>
```

Disables support of escape sequences for terminal TT3.

```
VMR>
```

Enables host-terminal synchronization for terminal TT22.

```
VMR>
VMR>
LA30S=TT0:
LA30S=TT2:
```

Sets TT2 as an LA30S terminal and then displays all the LA30S terminals.

```
VMR>
NETUIC=[20,3]
```

Displays the current network UIC.

```
VMR>
```

Sets the network UIC to [25,6].

```
VMR>
```

Enables I/O queue optimization for the device DM1, using the NEAR algorithm.

```
VMR>
OPT=DB0:NEAR
OPT=DB1:NEAR
OPT=DM0:NEAR
OPT=DM1:NEAR
```

Displays all the disk devices that have I/O queue optimization enabled.

```
VMR>
```

Enables parity generation and checking for terminal TT15 and informs the system that if an even number of bits are counted during data transmission, the transmission of data does not return an error.

```
VMR>
```

Informs the system that it should ignore special characters typed at terminal TT22 and that it should pass the characters to a program in their binary form.

```
VMR>
PLCTL=1600.:600.:200.:51.
```

Displays the current pool limit parameters used by the Pool Monitor Task (PMT).

```
VMR>
```

Informs the system that terminal TT22 has a printer port.

```
VMR>
```

Sets TT0 to be a privileged terminal.

```
VMR>
PRIV=TT0:
PRIV=TT1:
```

Displays all privileged terminals.

```
VMR>
```

Informs the system that the terminal TT15 supports the ReGIS graphic character set.

```
VMR>
```

Sets the terminal line corresponding to TT21 as a remote dial-up line.

```
VMR>
REMOTE=TT1:
REMOTE=TT2:
REMOTE=TT3:
REMOTE=TT4:
REMOTE=TT21:
```

Displays all the terminals that are currently set as remote dial-up lines.

```
VMR>
RNDC=6.
```

Displays the current length of the round-robin scheduling interval.

```
VMR>
```

Sets the length of the round-robin scheduling interval to 10_{10} ticks.

```
VMR>
SECPOL=285.:640:44%
```

Displays the amount of secondary pool currently being used.

```
VMR>
```

Sets TT3 to be a slaved terminal.

```
VMR>
```

Informs the system that terminal TT22 accepts software-defined character sets.

```
VMR>
SWPR=5.
```

Displays the current value for the Executive swapping priority range.

```
VMR>
```

Sets the Executive swapping priority range to 10_{10}.

```
VMR>
Term=TT25:  VT100
```

Displays the terminal type of terminal TT25 as VT100.

```
VMR>
```

Requests the system to give terminal TT25 the same capabilities as a VT200-series terminal.

```
VMR>
```

Enables terminal synchronization for terminal TT22.

```
VMR>
VT05B=TT4:
VT05B=TT5:
VT05B=TT6:
```

Displays all VT05B terminals.

```
VMR>
```

Enables write-checking on device DK1.

```
VMR>
```

Defines a partition called SYSPAR whose base address is 42000_8 and whose length is $14,000_8$ bytes.

```
VMR>
```

Eliminates the main partition SYSPAR.

```
VMR>
```

Creates a partition named GEN, setting the base address as low as possible and making the partition as large as possible.

```
VMR>
```

Establishes the top of pool.

```
VMR>
POOL=1200:11470.:11738.:2003
```

Shows that the last virtual address in the Executive is 120000, the longest free block is $11,470_{10}$ words, and the total of all pool space is $11,738_{10}$ words. The lowest physical address at which a partition can start is 2003 (32-word blocks).

5.5.17 TASKLIST

Use the TASKLIST (TAS) command to display a description of each installed task in the system. The display contains several columns that list, from left to right, the following information:

- Task name
- Task version identification
- Task Control Block (TCB) address
- Partition name
- Task priority
- Size of task in bytes (in octal)
- Load device identification
- Disk address logical block number (in octal) on unsaved systems, or the image or task file identification on saved systems
- Task memory state (saved systems only)

The display lists the tasks installed in primary pool and then the tasks installed in secondary pool. The tasks in secondary pool are indicated by a plus sign (+).

Format

```
TAS[KLIST][taskname]
```

Parameter

taskname

Specifies the name of a single installed task, whose description you want displayed.

Examples

```
VMR>

TKTN   03.14    035300 GEN      248. 00010000 LB0:-00631420
...MCR 01.01    035174 GEN      160. 00040000 LB0:-00630300
MCR... 3.3      035070 GEN      160. 00010000 LB0:-00627721
...INS 4        035614 GEN      100. 00040000 LB0:-00627333
...LOA 03.3     035510 GEN       50. 00040000 LB0:-00627643
...UNL 03.4     035404 GEN       50. 00040000 LB0:-00070512
VMR>
```

Illustrates an unsaved system.

```
VMR>
...LDR 08.03      034210 SYSPAR      248. 00003000 LB0:- FILE ID:(1111,12) FIXED
TKTN   03.14      115424 GEN         248. 00010000 LB0:- FILE ID:(31540,27)
MTAACP 0010       115214 GEN         200. 00013500 LB0:- FILE ID:(31461,24)
MCR... 3.3        116464 SYSPAR      160. 00010000 LB0:- FILE ID:(31454,12)
...DCL 0113A      115634 GEN         160. 00040000 LB0:- FILE ID:(31344,53)
DCL... 0113A      115530 GEN         160. 00010000 LB0:- FILE ID:(31345,5)
...MCR 01.01      036770 GEN         160. 00040000 LB0:- FILE ID:(30546,34)
...MOU 23.00      036250 GEN         160. 00040000 LB0:- FILE ID:(5424,15)
F11ACP M0320      117674 GEN         149. 00065700 LB0:- FILE ID:(30101,47)
DB00F1 M0320      117570 GEN         149. 00065700 LB0:- FILE ID:(31370,6)
COT... X00.01     117360 GEN         145. 00015400 LB0:- FILE ID:(31336,120)
...DMO 23.00      117254 GEN         140. 00040000 LB0:- FILE ID:(31346,23)
...INI 20.01      116674 GEN         140. 00040000 LB0:- FILE ID:(31424,55)
...UFD V0410      116044 GEN         140. 00040000 LB0:- FILE ID:(31541,34)
PMD... 3.1        114630 GEN         140. 00023600 LB0:- FILE ID:(31464,70)
HRC... 01.01      036560 GEN         140. 00050400 LB0:- FILE ID:(31240,21)
SHF... 03.01      115320 SYSPAR      105. 00010000 LB0:- FILE ID:(31521,40)
FXR... 01         117104 GEN         100. 00003100 LB0:- FILE ID:(31371,113)
...INS 4          116570 GEN         100. 00027700 LB0:- FILE ID:(31435,12)
...SAV 04.35      116254 TSTPAR      100. 00040000 LB0:- FILE ID:(31475,5)
SAVT0  04.35      036130 TSTPAR      100. 00040000 LB0:- FILE ID:(31475,5)
...PRV 4.30       036664 GEN         100. 00040000 LB0:- FILE ID:(3433,163)
...AT. 04.25      037254 GEN          65. 00057700 LB0:- FILE ID:(31421,131)
...BOO 04.08      117464 GEN          50. 00040000 LB0:- FILE ID:(31323,6)
...ACS 01.1       115740 GEN          50. 00040000 LB0:- FILE ID:(31321,73)
...HEL 01.25      037360 GEN          50. 00044100 LB0:- FILE ID:(31416,15)
VMR>
```

Illustrates a saved system. The display for a saved system differs from the display for an unsaved system in that the file ID of the task file, rather than the logical block number, is displayed. FIXED indicates that the task is fixed in memory.

```
VMR>
   .
   .
   .
BAP0   02         113440  GEN       80. 00045700 LB0:- FILE ID:(4332,27)
QMG... 1.7        112450  GEN       75. 00031400 LB0:- FILE ID:(4366,2)
LP0    1.9        112734  GEN       70. 00015400 LB0:- FILE ID:(4354,135)
LP1    1.9        112604  GEN       70. 00015400 LB0:- FILE ID:(4354,135)
SHC... 01.01      110600  GEN       60. 00051700 LB0:- FILE ID:(4360,2)
SHUTUP 02         110320  GEN       50. 00011200 LB0:- FILE ID:(4320,65)
...DMO 23.20      006025+ GEN      160. 00015500 LB0:- FILE ID:(4167,35)
...MCR 2.4        006037+ GEN      160. 00032000 LB0:- FILE ID:(2132,227)
...MOU 2502       006041+ GEN      160. 00040000 LB0:- FILE ID:(4205,13)
...DCL 00         006130+ GEN      160. 00035600 LB0:- FILE ID:(2535,1525)
   .
   .
   .
VMR>
```

This example is from an RSX-11M-PLUS saved system that has tasks installed in secondary pool.

5.5.18 TIME

Use the TIME (TIM) command to set the date and time or to display the date and time. The TIME command has two formats.

Formats

```
TIM[E] [hrs:mins[:secs]] [m1/day/year]
```

```
TIM[E] [hrs:mins[:secs]] [day-m2-year]
```

Parameters

hrs
Specifies hours (range 0 to 23).

mins
Specifies minutes (range 0 to 59).

secs
Specifies seconds (range 0 to 59). This parameter is optional; the default is zero.

m1
Specifies the numeric representation of month (range 1 to 12).

m2
Specifies the 3-letter abbreviation for month.

day
Specifies the day (range 1 to 31).

year
Specifies the year (range 0 to 99). The year is relative to 1900; therefore, 1987 is entered as 87.

Notes

1. You can specify the time and date in any order.
2. If you specify the time and the date, VMR sets the system image clock and calendar.
3. If you specify the time but not the date, VMR only sets the clock.
4. If you specify the date but not the time, VMR only sets the calendar.
5. If you don't specify a time or date, VMR displays the current system image time and date.
6. The form used to specify the date does not affect the display format. The date is always displayed as day-month-year.
7. All numeric values are decimal. No terminating period is required.

Examples

```
VMR>
10:23:31 21-JUN-87
```

Displays the current system image time and date.

```
VMR>
```

Sets the time to 14:30:00 and the date to 24-JUN-87.

5.5.19 UNFIX

Use the UNFIX (UNF) command to free a fixed task from the virtual memory of the system image. The UNFIX command is the complement of the FIX command.

If a fixed task exits or aborts, it still occupies the physical memory in the partition.

Format

```
UNF[IX] taskname[/keyword]
```

Keywords

/REG
/RON

Parameter

taskname

Specifies the task that you want to unfix from memory.

Keyword Descriptions

/REG

Specifies that the task to be unfixed is a common region.

/RON

Specifies that the read-only segment of a task is to be unfixed from memory.

Example

```
VMR>
```

Unfixes task XKE, which frees the partition in which it resides.

5.5.20 UNLOAD

Use the UNLOAD (UNL) command to remove a loadable device driver from the system image. If a device is attached, however, its driver cannot be unloaded.

To unload a device driver from the system image, the driver's symbol definition file must reside on LB: under the same directory as the directory of the system image file. By default, the system directory is [1,54]; the library directory is [3,54].

The UNLOAD command cannot remove a loadable database from the system image, even if the database was loaded by means of the LOAD command.

Format

```
UNL[OAD] dd:[/VEC]
```

Parameters

dd:

Specifies a 2-character ASCII device name. This is a required parameter.

/VEC

Specifies that the driver to be unloaded is *vectored.*

Vectoring is a technique for building a portable device driver. You can load a vectored driver into any RSX-11M-PLUS or Micro/RSX operating system that has the same version number as the system on which the driver was built. However, you must specify the /VEC keyword to suppress the validation of the Executive symbol table files used in the driver. /VEC avoids the comparison of the driver symbol table file and the Executive symbol table file.

For more information on vectored device drivers, see the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual.*

Example

```
VMR>
```

Unloads the line printer driver (LPDRV).

5.6 VMR Error Messages

This section contains explanations for VMR error messages. Error messages can appear in one of the following formats:

Format 1: Diagnostic Error Messages

```
VMR -- *DIAG* message
```

A diagnostic error message usually indicates that something unexpected happened, but that the error did not interfere with VMR's operation.

Format 2: Unmarked Error Messages

```
VMR -- message
```

An unmarked error message usually indicates that VMR rejected the command line. If the command line that caused the error is in an indirect command file, VMR ignores the rejected line and attempts to execute the remaining command lines.

Format 3: Fatal Error Messages

```
VMR -- *FATAL* message
```

A fatal error message indicates that VMR rejected the command line and that VMR has either exited or reset itself to the beginning (that is, VMR redisplays the Enter filename: prompt). If the command line that caused the error is in an indirect command file, VMR exits without attempting to execute the remaining command lines.

The following explanations for error messages contain the text of the error message, but not the format. They are in alphabetical order.

VMR—Access to common block denied

Explanation: You attempted to install a task that has specified illegal access to a shared region.

User Action: Enter the command from a User Identification Code (UIC) that has the necessary access privileges.

VMR—Addressing extensions not supported

Explanation: You attempted to install a task that was built, using the virtual section (VSECT) directive, into a system image that does not support the VSECT feature.

User Action: No user action is required. This cannot be done.

VMR—Alignment error

Explanation: The base address or size of the partition being created with the SET command conflicts with existing partitions or physical memory.

User Action: Create the partition with a different base address.

VMR—Base address must be on a 4K boundary

Explanation: The virtual base address of the task being installed is not aligned on a 4K boundary.

User Action: Rebuild the task for the correct starting address; then, reenter the command.

VMR—Base mismatch common block <commonname>

Explanation: The base address of the partition does not match the address of the common block. The task image is probably built against a different version of the common than the common currently installed in the system.

User Action: Rebuild the task against the current version of the common. Then, reenter the command.

VMR—Bus switch not loaded

Explanation: You attempted to load a device that requires a bus run before you loaded the bus switch driver (BS:).

User Action: Load the bus switch driver (BS:); then, reenter the command.

VMR—Cannot fix an I/D space task

Explanation: You attempted to fix a task that is built with instruction and data space.

User Action: No user action is required. These tasks cannot be fixed.

VMR—Cannot have multiple pool or CPU partitions

Explanation: You attempted to create the secondary pool partition SECPOL in a system that already has a SECPOL partition.

User Action: No user action is required. There can be only one SECPOL partition in a system. (However, you can load additional secondary pool space into memory by creating secondary pool subpartitions in the GEN partition. For more information, see Chapter 8.)

VMR—Cannot install shared region with increment

Explanation: You specified both the /INC=size and the /WB=YES keywords when installing a common. When a common is installed with an increment, it will no longer fit in its original task image file; therefore, you cannot have the common written back into its task image file (/WB=YES).

User Action: Install the common with either the /INC=size keyword or the /WB=YES keyword.

VMR—Cannot install tasks or commons from other than LB:

Explanation: SAVE does not allow saving of a system that has tasks and/or commons that were not installed from the library device LB:.

User Action: No user action is required. A task or common cannot be installed from a device other than LB:.

VMR—Cannot load/unload a pseudo device

Explanation: The device you attempted to load is a pseudo device. Pseudo devices do not have device drivers.

User Action: When you load or unload a device driver, specify the physical name of the driver.

VMR—Cannot unfix a mapped common region

Explanation: You attempted to unfix a common region that is currently mapped by a task.

User Action: No user action is required. This cannot be done.

VMR—Cannot unfix the directive common

Explanation: Because directive commons are merged with DIR11M as they are fixed, the size of a particular common is unknown.

User Action: No user action is required. Directive commons can be removed but not unfixed.

VMR—Checkpoint area too small

Explanation: The area allocated for checkpointing in the task image file is smaller than the partition into which the task is being installed. The system image does not support the dynamic allocation of checkpoint space.

User Action: Allocate additional checkpoint space by rebuilding the task with the Task Builder (TKB) switch /AL. (For more information, see the *RSX-11M-PLUS and Micro/RSX Task Builder Manual.*)

VMR—Checkpoint space too small, using checkpoint file

Explanation: The checkpoint space allocated in the task image file is too small to hold the task (usually because of the /INC keyword). The system image supports the dynamic allocation of checkpoint space. It will use the checkpoint file to store the task when it rolls it out to disk.

User Action: No user action is required. This is a warning message.

VMR—Circular redirect error

Explanation: The attempt to redirect a device failed because it would result in a circular device list.

User Action: Redirect the device to a different device.

VMR—Command I/O error

Explanation: The system detected an I/O error while the command line was being read. This error may indicate that pool has been depleted.

User Action: Reenter the command. If necessary, wait until pool is restored; then, reenter the command.

VMR—Command line too long

Explanation: The command line is longer than 132_{10} characters.

User Action: Reenter the command line with fewer characters.

VMR—Command syntax error <command-line>

Explanation: The syntax of the command line is incorrect. The incorrect part of the command line is enclosed in brackets.

User Action: Reenter the command line with the correct command line format.

VMR—Common block currently installed

Explanation: You attempted to install a common block that has already been installed.

User Action: Be sure you specified the correct name for the common block. If so, no further action is necessary.

VMR—Common block is task partition <commonname>

Explanation: A task's request for access to a common block has been rejected because the requested partition is a task partition. This error usually occurs when there is an error in the task source code or when two tasks use the same name.

User Action: Modify the code to correct the error or to specify a different task name.

VMR—Common block not loaded <commonname>

Explanation: The common block specified by VMR in the error message has been linked to the task, but the common block has not been installed yet.

User Action: Install the common block, and then install the task.

VMR—Common block parameter mismatch <commonname>

Explanation: Parameters of a common block did not match those in the task's label block. This error occurs when you attempt to run a task with either an older or a more recent version of a common.

User Action: Rebuild the task with the common; then, reenter the RUN command.

VMR—CTB <name> does not exist

Explanation: The Controller Table Block (CTB) name that you specified with the LOAD /CTB keyword does not exist in the resident database.

User Action: Reenter the command with the correct CTB name.

VMR—CTB name <name> is a duplicate

Explanation: The loadable database contains a CTB with a name that is the same as a CTB name in the resident database. CTB names must be unique.

User Action: Rename the CTB in the loadable database; then, reload the database.

VMR—CTB <name> not supported by driver—not loaded

Explanation: This is a warning message. VMR displays this message for either of the following reasons:

- The Driver Dispatch Table (DDT) in the driver does not have all the CTB mnemonics that the rest of the database (that is, Data Control Blocks (DCBs) and CTBs) implies that it should have.
- The CTB is defective. A defective CTB may contain a wrong name or it may point at the wrong DCB.

User Action: Reenter the command with the correct CTB name.

VMR—Data space file exceeds available space in system image

Explanation: The RSX11M.SYS file is not large enough to contain the Executive data-space image file DSP11M.TSK.

This message applies only to systems with separate Executive instruction and data space.

User Action: Create a larger system image file with the PIP switch /BL or the COPY command qualifier /BLOCK_SIZE, as follows:

```
MCR>

DCL>
```

VMR—DCB table for CTB <name> is full

Explanation: LOAD attempted to write the address of the Device Control Block (DCB) of the loadable database into the DCB table of the Controller Table for a multicontroller device. There were no null entries in the DCB table.

User Action: For more information on DCB tables, see the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver.*

VMR—Deferred binding not supported

Explanation: You specified the /DFB=YES keyword with the INSTALL command, but the system does not support deferred binding.

User Action: Reenter the command without the /DFB keyword.

VMR—Device <ddnn:> is attached

Explanation: You attempted to unload a driver that has one or more of its device units attached. The first attached unit is ddnn. You cannot unload a driver that has device units attached.

User Action: Specify a different device or wait for the task to complete; then, reenter the command.

VMR—Device <ddnn:> is not in system

Explanation: VMR displays this message for any of the following reasons:

- You attempted to save an image on a device that does not exist on the host system.
- You attempted to redirect I/O requests to a device that does not exist on the host system.
- You specified a device in the LOAD command line for which there is no database. This is caused by one of the following two conditions:
  - The device does not exist in the system device tables.
  - You did not declare the device to be loadable; therefore, LOAD cannot find a loadable database for it.
- You specified a device in the command line that has not been defined in the system image.
- You specified a device in the DEVICES command line that does not exist in the system image.

User Action: Reenter the command with either the name of a device that is known to the system or the correct control and status register (CSR) address for the device.

VMR—Device not mounted

Explanation: You failed to mount the device that contains the system image file.

User Action: Mount the system device and reenter the command.

VMR—Device not redirectable

Explanation: You attempted to redirect a device that is not redirectable (according to the Unit Control Block (UCB) for the device).

User Action: Reenter the command with a valid device name.

VMR—Device not terminal

Explanation: You attempted to set terminal characteristics for a device that is not a terminal.

User Action: Use the MCR command DEV to obtain a list of valid terminal names; then, reenter the command with one of the names listed.

VMR—Device not variable speed multiplexer

Explanation: You attempted to set the speed for a terminal that is not attached to a variable-speed multiplexer (such as DHV11 or DZV11).

User Action: No user action is required. You cannot set the baud rate for a terminal that is not attached to variable-speed multiplexer.

VMR—Directive common not contiguous to DIR11M

Explanation: The directive commons were not installed contiguous to DIR11M.

User Action: Install all the directive commons sequentially (that is, install DIR11M first, then DR211M, then DR311M, and so on). Do not install other common regions between the directive commons.

For an example of how to load the directive commons, see the SYSVMR.CMD file in directory [1,54] on the pseudo device LB:.

VMR—Directive common(s) not installed

Explanation: Not all of the directive commons were installed in the system image.

User Action: Install the commons. Then, fix them in memory.

VMR—Driver already resident

Explanation: You attempted to load an already loaded device driver or one that is permanently resident in the system image.

User Action: No user action is required. This is an informational message.

VMR—Driver built with wrong executive STB file

Explanation: The symbol definition file for the driver you attempted to load or unload is not compatible with the Executive symbol definition file for the current system image. This means that the driver has been built for another system and must be rebuilt before you can load it into the current system.

User Action: Task build the driver again with the correct RSX11M.STB file.

VMR—Driver cannot be unloaded

Explanation: You attempted to unload a permanently resident driver (that is, a driver linked permanently to the Executive).

User Action: No user action is required. Only loadable drivers can be removed.

VMR—Driver dispatch table is inconsistent

Explanation: You attempted to load a driver with an illegally formatted DDT.

User Action: For information on formatting the DDT, refer to the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver.*

VMR—Driver not loaded

Explanation: In the UNLOAD command line, you specified a driver that is not resident in memory.

User Action: Before you initialize or mount the device, load the device driver.

VMR—Driver requires running system for load/unload

Explanation: The driver you requested has the symbols $ddLOA or $ddUNL in its source. Support of driver load/unload calls can be provided only by the MCR commands LOAD and UNLOAD.

User Action: For information on the MCR commands LOAD and UNLOAD, see the *RSX-11M-PLUS MCR Operations Manual.*

VMR—Executive too large

Explanation: The Executive is too large to be accommodated in the system image. This message indicates that the Executive is larger than 20K.

User Action: No user action is required. The system is unusable.

VMR—External headers not supported

Explanation: You attempted to install a task built with external headers in a system that was not generated with external header support.

User Action: Include external header support during system generation or by modifying the command file INSBLD.CMD. If you cannot include external header support, then enter the INSTALL command without specifying /XHR=YES.

VMR—Fast map not supported

Explanation: You attempted to install a task with the /FMAP=YES keyword, without having first selected support for fast mapping during system generation.

User Action: Install the task with the /FMAP=NO keyword.

VMR—Feature not supported in system image

Explanation: The command you entered requires a feature that is not currently supported by the system.

User Action: Select support for the desired feature during system generation.

VMR—File <name> has illegal STB format

Explanation: The driver's symbol definition file contains illegal object code or data record format.

User Action: Check and rebuild the driver; then, reenter the command.

VMR—File <name> has inconsistent data

Explanation: This error can occur for the following reasons:

- You attempted to fix a partition or a task that is logically beyond the end of the system image file.
- The partition for loadable drivers is logically beyond the end of the system image file. The system image is probably corrupt.
- The partition of the common library you are attempting to load is logically beyond the end of the system image file.

These problems indicate that either the system image file is too small or the organization of the partitions within the system image file is incorrect.

User Action: Use the PAR command to examine the partitions. Also, check the size of the system image file RSX11M.SYS. If necessary, create a larger system image file with the PIP switch /BL or the COPY command qualifier /BLOCK_SIZE, as follows:

```
MCR>

DCL>
```

VMR—File <name> not a valid driver task image

Explanation: The driver's task image is invalid for one of the following reasons:

- The driver's task image has been overlaid.
- The driver has a header.
- The driver has referenced a resident library.

User Action: Rebuild the driver. To ensure that the task image is valid, follow the guidelines in the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver.*

VMR—File not contiguous

Explanation: You attempted to install a task or to load a device driver from a noncontiguous file.

User Action: To produce a contiguous file, use the DCL command COPY /CONTIGUOUS or the Peripheral Interchange Program (PIP) switch /CO. Then, reenter the command. (For information on the COPY command, see the *RSX-11M-PLUS Command Language Manual.* For information on PIP, see the *RSX-11M-PLUS Utilities Manual.*)

VMR—File not found

Explanation: You requested a file not located in the directory of the volume you specified.

User Action: Use the PIP switch /LI to obtain a directory listing. Reenter the command with the correct file specification.

VMR—File not task image

Explanation: The data in the label block of the task you tried to install is incorrect, which indicates that the file is not a task image.

User Action: Reenter the command with the correct name for the task image file. If the task image is corrupt, modify and rebuild the task, or use another version of the task image.

VMR—Illegal device/volume

Explanation: You attempted to save your system image on an invalid device.

User Action: Reenter the command with a valid device name. (For a list of valid devices, see Section 5.5.15.)

VMR—Illegal driver task APR usage

Explanation: The device driver being loaded must be built for Active Page Register (APR) 5 and must be less than or equal to 4K words.

User Action: Rebuild the driver correctly.

VMR—Illegal error severity code <code>

Explanation: This message indicates an internal failure in VMR.

User Action: If this error persists, submit a DIGITAL Software Performance Report (SPR).

VMR—Illegal file specification <filespec>

Explanation: The file specification <filespec> is incorrect or contains wildcards.

User Action: Reenter the command with a valid file specification.

VMR—Illegal first APR

Explanation: The task you tried to install was built using an incorrect base APR.

User Action: Rebuild the task and observe the following rules:

- A nonprivileged task must be built with APR 0 as its base APR.
- A privileged task that does not map into the Executive must be built with APR 0 as its base APR.
- If the Executive is 16K words or less, a privileged task that is to map to the Executive routines and data structures must be built with APR 4 as its base APR.
- If the Executive is 20K words, (or less on RSX-11M-PLUS systems), a privileged task that is to map to the Executive routines and data structures must be built with APR 5 as its base APR.

VMR—Illegal function

Explanation: You entered a command that VMR does not recognize, or the command line refers to a feature that is not supported in the system. You may have entered the command line incorrectly.

User Action: Reenter the command with the correct command syntax.

VMR—Illegal Get Command Line <error code>

Explanation: This message indicates a system failure.

User Action: If the error persists, submit an SPR.

VMR—Illegal keyword value

Explanation: You entered a value for a keyword that is out of range.

User Action: Reenter the command line with a valid keyword.

VMR—Illegal priority

Explanation: You specified an invalid priority for the task.

User Action: Specify a priority value between 1 and 250, where 250 is the highest priority and 1 is the lowest priority. A priority is assumed to be octal unless you put a period after the priority number.

VMR—Illegal slave attribute

Explanation: You attempted to install a task with the /SLV keyword, but the task's name is in the form ...xxx.

User Action: No user action is required. The task is a prototype task and cannot be installed as a slaved task.

VMR—Illegal switch <switch>

Explanation: The switch following the error message is incorrect in the context of the command line you have entered.

User Action: Reenter the command line with the correct switch.

VMR—Illegal use of <pname> partition or common

Explanation: You attempted to install a task in a central processing unit (CPU) partition (CPUPAR) or secondary pool partition (SECPOL). Although you can install a task with its Task Control Block (TCB) in secondary pool, the task itself cannot be installed in the secondary pool partition.

User Action: Install the task in a partition other than CPUPAR or SECPOL.

VMR—Illegal use of <pname> partition or region

Explanation: You attempted to load a driver into a common or secondary pool partition.

User Action: Reenter the command with a different partition.

VMR—Illegal value for symbol <symname> in file <filename>

Explanation: The symbol symname is defined to an illegal value in file ddDRV.STB. This can occur for several reasons, some general and some specific to individual symbols. For example:

- Symbols cannot have odd values.
- Most symbols cannot be defined as zero.
- The $ddTBL symbol must exist and have a value less than $ddEND.
- Mapped loadable drivers cannot reference $INTSV.

Since ddDRV.STB had to pass a previous inspection by the LOA command, there probably has been a procedural error or ddDRV.STB has been corrupted.

User Action: See the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver* for more information.

VMR—Incorrect assignment of SY: or LB:

Explanation: The device SY: or LB: is not assigned to the same device as the system image file you are trying to work with in VMR.

User Action: No user action is required. This is a warning message.

VMR—Indirect command syntax error <command-line>

Explanation: You entered an invalid indirect command file specification.

User Action: Reenter the command line with the correct indirect command file specification.

VMR—Indirect file depth exceeded <command-line>

Explanation: You attempted to reference more than two levels of indirect command files.

User Action: No user action is required. VMR is limited to two levels of nesting.

VMR—Indirect file open failure <command-line>

Explanation: The indirect command file specified in the command line could not be opened.

User Action: Check to see that the specified file exists and is not locked.

VMR—Install device not LB0:

Explanation: You attempted to install a task from other than the boot device.

User Action: No user action is required. This cannot be done.

VMR—Installed tasks or commons may no longer fit in partition

Explanation: You shortened a partition so that the tasks and/or commons that are installed in it may no longer fit. VMR shortens the partition even though this situation exists.

User Action: No user action is required. This is a warning message.

VMR—Interrupt vector already in use

Explanation: The interrupt vector for the device driver being loaded is already being used by another driver. (VMR assumes a vector is in use if it does not point to one of the nonsense-interrupt entry points in the system image.)

User Action: Reenter the CON command with a valid vector address (that is, an address that is not currently being used).

VMR—Invalid driver database at offset xxxx in file <filename>

Explanation: The driver name has an invalid database value at offset xxxx relative to the symbol $ddDAT. This can occur for one of the following reasons:

- An invalid value, for example, D.UCB was odd.
- The value for a word is not within the bounds of the loadable database. LOAD produces an error message citing the offending value.
- An element is missing from the loadable database. LOAD attempts to verify one kind of data structure as another kind. For example, if a DCB is missing, LOAD tries to verify the following UCB as the DCB. LOAD cites the DCB as being incorrect.

User Action: See the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver.*

VMR—Invalid interrupt vector

Explanation: You specified an interrupt vector address in the driver database that is too high in memory for the system image.

User Action: Reenter the command with a lower vector address.

VMR—Invalid keyword

Explanation: You specified a keyword that is incorrect in the context of the command.

User Action: Reenter the command line with the correct keyword.

VMR—Invalid record address for file <filename>

Explanation: LOAD attempted to write outside of the system image file or at a byte boundary. This error may be caused by either a corrupted system image or an STB file that does not match the system image.

User Action: If the problem is due to a corrupted system image, create a new system image file and run VMR again. If the problem is due to an incorrect STB file, locate the correct STB file before you create the new system image file.

VMR—Invalid speed

Explanation: VMR displays this message for either of the following reasons:

- You requested a speed that the multiplexer line you specified does not support.
- You specified unequal receive and transmit speeds for a multiplexer that does not support split-speeds.

User Action: Reenter the command with a speed that is supported by the corresponding multiplexer.

VMR—Invalid time parameter

Explanation: The time field you specified in the RUN command line is incorrect.

User Action: Reenter the command with the time in hours and minutes (for example, 13:00 for 1:00 P.M.).

VMR—Invalid UIC

Explanation: You specified an illegal value for either a group number or a member number.

User Action: Reenter the command with group and member numbers between 1 and 377_8.

VMR—I/O error on input file <filename>

Explanation: An input I/O error occurred on the indicated file. Either the file is corrupted or the device is damaged.

User Action: Check the device; if necessary, use another device. If the device is functioning properly, but the error persists, the file is probably corrupted. If possible, use the Backup and Restore Utility (BRU) to restore a copy of the file.

VMR—I/O error on output device

Explanation: SAVE has encountered an I/O error on the output device while writing the system image. This is a hardware-related problem.

User Action: Correct the problem with the output device drive. Then, create a new system image file and run VMR again.

VMR—I/O error on output file <filename>

Explanation: An output I/O error occurred on the indicated file. The device drive may be damaged or the file may be corrupt.

User Action: Correct the problem with the output device drive or use new media. Then, create a new system image file and run VMR again.

VMR—KRB <KRBname> interrupt vector <octal-number> in use

Explanation: You attempted to load a driver that specifies an interrupt vector that is not pointing to a nonsense-interrupt entry point address. This is usually caused by a previously loaded driver that has used this vector.

User Action: No user action is required. When you receive this message, LOAD has loaded the driver correctly, but it has not overwritten the interrupt vector.

VMR—KRB <KRBname> interrupt vector <octal-number> too high

Explanation: You attempted to load a driver that specifies an interrupt vector address that is higher than the highest permissible vector address in the system.

User Action: No user action is required. When you get this message, LOAD has loaded the driver correctly, but it has not overwritten the interrupt vector.

VMR—KRB <KRBname> not in loadable database

Explanation: You attempted to load the driver for a multicontroller device with a loadable database. The special symbol (KRBname) that defines the location of the Controller Request Block (KRB) in the multicontroller device's CTB table is not defined in the loadable database.

User Action: For more information on the special symbol KRB, refer to the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver.*

VMR—KRB table of CTB <CTBname> will not accept KRB <KRBname>

Explanation: You attempted to load the driver for a multicontroller device with a loadable database. If this operation is to succeed, then the following conditions must exist:

- The appropriate slot in the CTB must exist.
- The slot in the CTB must be unused.

When VMR issues this error message, one of these conditions has not been met.

User Action: See the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver.*

VMR—Length mismatch common block <blockname>

Explanation: The length parameter for the common block, as described in the label block for the task image, does not match the corresponding length parameter defined in the system image. A task's label block data must match system data for that task before it can be installed.

User Action: Rebuild the task and then install it.

VMR—Loadable driver larger than 4K

Explanation: This is a warning message. You are loading a driver that is larger than 4K words.

User Action: Make sure your driver is written to map the additional 4K words of memory when needed.

VMR—Loadable driver support not in system

Explanation: You attempted to load a device into a system image that does not contain the Executive routine $INTSI. The routine is needed to support loadable device drivers.

User Action: Include support for loadable device drivers during system generation. For more information, see the *RSX-11M-PLUS System Generation and Installation Guide.*

VMR—Logical device not in system

Explanation: The device you specified has not been defined and therefore ASSIGN could not find it in the logical device assignment table.

User Action: Use the DFL command to define the logical device name; then, reenter the command.

VMR—LUN out of range

Explanation: You attempted to reassign a logical unit number (LUN) for a task that is higher than the maximum number of LUNs allocated for the task when it was built.

User Action: No user action is required. This cannot be done.

VMR—No checkpoint space, assuming not checkpointable

Explanation: This is a warning message. You attempted to install a task as checkpointable, but the task was not built as checkpointable and therefore had no checkpoint space allocated in the task image file. Because the system image does not support the dynamic allocation of checkpoint space, INSTALL does not consider the task to be checkpointable.

User Action: No user action is required. This is an informational message.

VMR—No checkpoint space or dynamic checkpoint file

Explanation: You attempted to install a task built with external headers or as checkpointable, but the task does not have any checkpoint space and the system does not have a checkpoint file.

User Action: Reenter the command without the /XHR=YES or /CKP=YES keywords.

VMR—No ICB pool space for CPU <cpu>

Explanation: The Interrupt Control Block (ICB) pool space required to load the device driver has been exhausted. The ICB space allocated during system generation for the specified central processing unit (CPU) may be insufficient for your application. However, note that more ICB space will become available during the system startup procedure.

User Action: Load the device driver during the system startup procedure (when more ICB space is available), or run the system generation procedure again and specify a larger ICB pool.

VMR—No LUNs

Explanation: The task that you specified as an argument of the LUN command does not have any logical units assigned to it.

User Action: No user action is required. This is an informational message; it indicates that there are no assignments to display.

VMR—No pool space

Explanation: The pool space required to load the driver is not available in the system image.

User Action: Wait for pool to be restored; then, reenter the command.

VMR—No room available in STD for new task

Explanation: There is no dynamic storage available to make an entry in the System Task Directory (STD).

User Action: No user action is required. The task cannot be installed.

VMR—No table space for optional directive common

Explanation: You attempted to fix a directive common in the RSX-11M-PLUS system image, but the extra table space built into the system database for optional directive commons has already been used. You may have loaded the directive commons incorrectly.

User Action: Install all the directive commons sequentially (that is, install DIR11M first, then DR211M, then DR311M, and so on). Do not install other common regions between the directive commons.

For an example of how to load the directive commons, see the SYSVMR.CMD file in directory [1,54] on the pseudo device LB:.

VMR—Nonexistent memory

Explanation: You attempted to define a partition in nonexistent memory.

User Action: Reenter the command with valid memory addresses.

VMR—Not enough APRs for task image

Explanation: The Task Builder (TKB) allows you to specify, as a multiple of 4K, the virtual base address of a task image. Privileged tasks that access the Executive start at either virtual address 100000_8 for a 16K Executive or at virtual address 120000_8 for a 20K Executive. If the virtual base address of the task is set too high, there are not enough APRs remaining to map the task image. (See the *RSX-11M-PLUS and Micro/RSX Task Builder Manual* for information on privileged tasks.)

User Action: Reset the virtual base address of the task image; then, reenter the command. For information on setting the virtual base address for privileged tasks, see the *RSX-11M-PLUS and Micro/RSX Task Builder Manual.*

VMR—Old device attached

Explanation: You attempted to redirect an attached device.

User Action: Wait until the device is no longer attached; then, reenter the command.

VMR—Open failure on file <filename>

Explanation: The indicated file cannot be opened.

User Action: Check to see that the specified file exists and is not locked.

VMR—Operation cannot extend above first 124K of memory

Explanation: You attempted to create with the SET command a CPU partition whose upper limit is above 124K words.

User Action: No user action is required. This cannot be done.

VMR—Operation not allowed for <ptype> partitions

Explanation: You attempted a SET operation that is not allowed for the type of partition specified in the error message.

User Action: No user action is required. This cannot be done.

VMR—Optional directive common successfully loaded

Explanation: VMR has successfully fixed in the system image a directive common that is not one of the commons required by RSX-11M-PLUS.

User Action: No user action is required. This is an informational message.

VMR—Partition already exists

Explanation: You attempted to define a partition, using the SET command, with a name already in use as a partition name.

User Action: Reenter the command with a unique partition name.

VMR—Partition or common region is busy

Explanation: VMR displays this message for the following reasons:

- You attempted to fix a task or common region in a partition that is full. The task or region cannot be fixed.
- You attempted to load a device driver into a partition, but the partition is full.
- You attempted to remove a partition that still has tasks installed in it. The partition cannot be removed.
- You attempted to remove a region that is referenced by tasks installed in the system.

User Action: To free the occupied memory partition, use the UNFIX or REMOVE command (see Sections 5.5.19 or 5.5.13, respectively). Then, reenter the command with the name of the task you want to fix or remove.

VMR—Partition <pname> is a common

Explanation: You attempted to load a device driver into a common partition.

User Action: Load the device driver into a different partition.

VMR—Partition <pname> is not a common

Explanation: Partition <pname> was found, but it is not a common partition.

User Action: Define the partition as a common partition; then, reenter the command.

VMR—Partition <pname> is too small

Explanation: You attempted to load the driver into a partition that is either too small for the driver or that does not currently have enough space for the driver. Or, you attempted to install a task that is too large for the specified partition.

User Action: Load the driver into a different partition, or create more room in the partition by removing drivers, commons, or fixed tasks.

VMR—Partition <pname> not in system

Explanation: VMR displays this message for either of the following reasons:

- The partition you specified in the command line is not in the system image.
- You attempted to load a driver into a partition that does not exist in the system, or LOAD found a symbol $xxCOM in the driver and partition xxCOM does not exist in the system.

User Action: Install the partition before you attempt to load the driver.

VMR—Partition <pname> not in system, defaulting to GEN

Explanation: You attempted to install a task in a partition that does not exist. VMR will try to install the task in the partition GEN.

User Action: No user action is required. This is an informational message.

VMR—Partition reduced to executive common size

Explanation: When an Executive common is installed in its partition, the partition is reduced to the top of the common to eliminate any unused space in the partition.

User Action: No user action is required. This is an informational message.

VMR—Pool link error at xxxx Block=yyyy zzzz

Explanation: When SET /POOL traced the pool linkage, it detected an error at location xxxx in the system image. The contents of xxxx is yyyy and the contents of xxxx+2 is zzzz.

User Action: The system image file is corrupted. Create a new system image file and run VMR again.

VMR—Privileged task overmaps I/O page

Explanation: A privileged task that may need access to the I/O page does not have that access. This message occurs if the task is installed with the /IOP=YES option.

User Action: Install the task without the /IOP=YES option.

VMR—Pseudo device assignment error

Explanation: You attempted to assign a logical device name to a pseudo device.

User Action: No user action is required. Logical device names cannot be assigned to pseudo devices.

VMR—Pseudo device redirect error

Explanation: You attempted to redirect one pseudo device to another pseudo device.

User Action: No user action is required. This is not allowed.

VMR—R/O partition <pname> not in system, defaulting to task's

Explanation: You specified a read-only partition that does not exist in the system.

User Action: No user action is required. INSTALL will place the read-only portion of your multiuser task into the same partition as the read/write portion of the task.

VMR—Receive data or by reference list not empty

Explanation: You attempted to remove a task with entries in its receive queues from the system image.

User Action: No user action is required. This cannot be done.

VMR—Reference to memory beyond end of system image file

Explanation: VMR attempted to fix a task, to load a driver, or to install a common beyond the space limits of the system image file. As a result, the command does not execute.

User Action: No user action is required. This cannot be done.

VMR—Space used

Explanation: You attempted to create a partition in a storage area that is already occupied.

User Action: Reenter the command with an area that is available.

VMR—Specified partition for common block

Explanation: You attempted to install a task in a common block.

User Action: No user action is required. This cannot be done.

VMR—Specified partition too small

Explanation: This message can occur for the following reasons:

- You attempted to install a task into a partition that is smaller than the task.
- You attempted to install a driver into a partition that is smaller than the driver.

User Action: Use the /INC keyword to reinstall the task with a larger increment (see the description of the INSTALL command). If the error persists, submit a Software Performance Report (SPR).

VMR—Symbol <symname> is doubly defined by file <filename>

Explanation: The symbol symname is defined twice in the specified file. Duplicate symbols are illegal.

User Action: No user action is required. The driver is not loaded or unloaded.

VMR—Symbol <symname> is undefined in file <filename>

Explanation: LOAD found the symbol symname in the specified file, but the symbol is not defined.

User Action: Define or delete the symbol and reenter the command line.

VMR—Syntax error

Explanation: You have typed the command line incorrectly.

User Action: Check the command specification; then, retype the command line according to the correct command format.

VMR—System image is not RSX-11M-PLUS

Explanation: You attempted to use RSX-11M-PLUS VMR on a non-RSX-11M-PLUS system image file.

User Action: No user action is required. This cannot be done.

VMR—Task active

Explanation: The task you specified as an argument to the command is active.

User Action: If necessary, abort the task and then reenter the command.

VMR—Task or common region already fixed

Explanation: You attempted to fix a task or common region in memory that is already fixed.

User Action: Be sure that you specified the correct task name. Note that the name of the task to be fixed in memory must be unique.

VMR—Task has attached common regions

Explanation: This message can occur for the following reasons:

- You attempted to fix or unfix a task in memory that is linked to a shared region. VMR cannot fix or unfix such tasks.
- You attempted to remove a task that is linked to a shared region from the system image. VMR cannot remove such tasks.

User Action: No user action is required.

VMR—Task image currently installed

Explanation: You attempted to install a checkpointable task that has checkpoint space allocated in its task image. The task has already been installed.

User Action: To install checkpointable tasks more than once, perform the following steps:

1. Allocate checkpoint space for the task in a system checkpoint file.
2. Use a different name for the task each time you install it.

VMR—Task image I/O error

Explanation: The task cannot be installed or the driver cannot be loaded. VMR could not read the task image file or could not rewrite the task image header. The specified device may be write-locked.

User Action: Enable write access or specify another device. Then, install a valid task image and reenter the command.

VMR—Task image virtual address overlaps common block <blockname>

Explanation: The virtual addresses you reserved for the task image overlap those you reserved for the common block specified in the error message. Corruption of the task image file may have caused the overlap.

User Action: Rebuild the task; then, reenter the command.

VMR—Task is checkpointable

Explanation: You attempted to fix a checkpointable task in memory.

User Action: No user action is required. Checkpointable tasks cannot be fixed.

VMR—Task is not multi-user

Explanation: You attempted to fix the read-only portion of a multiuser task in memory, but the task you named in the command line is not a multiuser task.

User Action: No user action is required. This cannot be done.

VMR—Task name already in use

Explanation: You attempted to install a task of the same name as that of one already installed in the system image.

User Action: Rename the task and then install it.

VMR—Task not installed with external header

Explanation: This is a warning message to indicate that the specified task is installed with its header in the pool.

User Action: No user action is required. This is an informational message.

VMR—Task not in system

Explanation: You referenced a task that is not installed in the system.

User Action: Install the task with the INSTALL command; then, reenter the command.

VMR—Task not removed, CLI has messages enabled

Explanation: You attempted to remove a command line interpreter (CLI) task that is able to receive messages from the system.

User Action: No user action is required. The task may have messages that it has not received yet, so the remove request is aborted.

VMR—Task or common not in memory

Explanation: You attempted to remove a task or common from the system that is checkpointed and, therefore, is not in memory.

User Action: Wait for the task to come back in to memory; then, remove it.

VMR—Task or common region is not fixed

Explanation: You attempted to unfix a task that is not fixed in the system image.

User Action: No user action is required. The task is already unfixed.

VMR—Task or common region not in system

Explanation: You referenced a task or common region that has not been installed in the system or you attempted to run a prototype task.

User Action: Install the task or common region with the INSTALL command; then, reenter the command.

VMR—TI redirect error

Explanation: You attempted to redirect the pseudo device TI:.

User Action: No user action is required. TI: cannot be redirected.

VMR—TT redirect error

Explanation: You attempted to redirect a terminal to the null device.

User Action: No user action is required. This combination is not allowed.

VMR—Too many common block requests

Explanation: You attempted to install a task that contains too many common block requests.

User Action: No user action is required. A task is limited to seven common block references.

VMR—Too many LUNs

Explanation: You attempted to install a task that contains more than 255_{10} logical unit number (LUN) assignment requests.

User Action: No user action is required. A task is limited to 255_{10} LUN assignments.

VMR—Too many symbols of the form $xxTMO in file name

Explanation: LOAD is attempting to extract symbols of the form $xxTMO from the driver's symbol definition file. There are more symbols of this form than LOAD can process.

User Action: See the *RSX-11M-PLUS and Micro/RSX Guide to Writing an I/O Driver* for more information.

VMR—Top of pool set to <value> , requested amount not available

Explanation: You requested more pool space than the system can provide.

User Action: No user action is required. The system provides as much as possible in this case.

VMR—Undefined common block <blockname>

Explanation: You attempted to install a task that references a common block that is not defined in the system image. This message may indicate that the task was built for another system.

User Action: Rebuild the task with the common; then, reenter the INSTALL command.

VMR—Unknown partition

Explanation: You attempted an operation on a partition that does not exist in the system.

User Action: Be sure you entered the correct partition name.

VMR—User D-space not supported

Explanation: An attempt was made to install a task with separate instruction and data space on a system that does not support user data space.

User Action: Select support for user instruction and data space during system generation, or install the task without separate instruction and data space.

VMR—Virtual terminal error

Explanation: You attempted to enable, using the SET command, a terminal option for a virtual terminal.

User Action: No user action is required. This cannot be done.

VMR—Write-check not supported for device

Explanation: You attempted to enable write-checking for a device that does not support it.

User Action: Reenter the command without requesting write-checking, or specify a device that supports write-checking. Write-checking is supported for all disks except DL-type devices (RL01 and RL02) and DX-type devices (RX01s).

Chapter 6
The SHUTUP Program

SHUTUP is the system program that enables orderly shutdown of an RSX-11M-PLUS or Micro/RSX system. SHUTUP sends warning messages, aborts nonprivileged tasks, and dismounts devices on the system.

6.1 Prerequisites to Running SHUTUP

Before you use SHUTUP to shut down the system, be sure the following tasks are installed:

- INDIRECT, the Indirect Command Processor
- ACS, if dynamic checkpoint space is established
- BYE
- DMO, if there are any mounted devices

If the indirect command file LB:[1,2]SHUTUP.CMD is present, you can use it to install the BYE and DMO tasks. (For more information, see Section 6.5.) The system installs Indirect and the ACS task for you.

6.2 Invoking SHUTUP

Use the RUN command to invoke SHUTUP from a privileged terminal, as follows:

```
>
```

SHUTUP displays the following message when invoked:

```
RSX-11M-PLUS Shut down program
```

6.3 SHUTUP Input

SHUTUP prompts for the number of minutes as follows:

- Number of minutes to wait before the system is shut down
- Number of minutes between warning messages
- Number of minutes before disabling logins

If you specify less than 4 minutes to wait before shutdown, SHUTUP does not ask the last two questions. Instead, SHUTUP sets the interval between messages to 1 minute and disables logins immediately.

The number of minutes between messages must be greater than zero and less than the number of minutes before shutdown minus three. If you specify either value incorrectly, SHUTUP ignores your answer and displays the prompt again.

The following is an example of SHUTUP prompts and user input:

```
Enter minutes to wait before shutdown:
Enter minutes between messages:
Enter minutes to wait before disabling logins:
Reason for shutdown (<CR> for none):
```

After receiving your input, SHUTUP asks for confirmation as follows:

```
OK to shutdown? [Y/N]:
```

If your response to the question is N (No), SHUTUP terminates without taking any action. If your response is Y (Yes), SHUTUP sends warning messages to terminals.

The following section explains the messages issued by SHUTUP.

6.4 SHUTUP Output

SHUTUP sends warning messages to terminals that are logged in to the system. However, a terminal set to NOBROADCAST receives the warning messages only during the last 5 minutes before shutdown.

SHUTUP sends the following warning messages:

```
Please finish up, 15 minutes before shutdown -- WALNUT
Reason for shutdown: Field Service PM
```

Three minutes later, SHUTUP sends additional messages:

```
Please finish up, 12 minutes before shutdown -- WALNUT
Reason for shutdown: Field Service PM
```

The messages include a 6-character name that is a DECnet node name (if the DECnet package is active) or the system name that was selected during system generation (if DECnet is not active). In the previous messages, WALNUT is the 6-character name.

SHUTUP continues to issue the warning messages at the specified interval. When 3 minutes remain before system shutdown, SHUTUP changes the interval between messages to 1 minute.

When the specified delay before disabling logins expires (measured from the time SHUTUP is started), SHUTUP displays the following message on the invoking terminal:

```
All further logins are disabled
```

After SHUTUP logs out all terminals, aborts all nonprivileged tasks, and dismounts all devices, it displays the following message on the invoking terminal:

```
SHUTUP operation complete
```

Then, SHUTUP halts the processor.

At this point, you can either bootstrap another system or resume operation of the current system. To continue with the current system, press the continue switch (CONT) on the CPU console of the PDP–11 or, on a MicroPDP–11, the Restart button. After you press the switch, logins are enabled and the following message appears on the invoking terminal:

```
Logins are now enabled
```

Before you use the system, mount the devices and follow normal system startup procedures.

6.5 The SHUTUP.CMD File

The indirect command file LB:[1,2]SHUTUP.CMD customizes the system shutdown procedure. During the system shutdown procedure, SHUTUP submits this file to the Indirect Command Processor for execution.

The SHUTUP task-build file LB:[1,24]SHUBLD.CMD specifies the amount of time (the timeout interval) that SHUTUP.CMD has to execute. The default timeout interval is 120_{10} seconds. However, if you are using an RSX–11M–PLUS system, you can change the timeout interval for SHUTUP.CMD by modifying SHUBLD.CMD. (Information on modifying SHUBLD.CMD is contained in the file.)

If the timeout interval expires and SHUTUP.CMD is still executing, SHUTUP asks if you want to wait for SHUTUP.CMD to finish. For example:

```
Command (AT.) Timeout - Continue waiting? [Y/N]:
```

If you enter N (No), SHUTUP continues without waiting any longer. If you enter Y (Yes), SHUTUP waits an additional 30_{10} seconds. If SHUTUP.CMD has not finished at the end of 30_{10} seconds, SHUTUP continues without waiting any longer.

Use SHUTUP.CMD to install BYE and DMO (SHUTUP cannot execute properly unless these tasks are installed). If the DECnet package is installed on your system, use SHUTUP.CMD to shut down the network. Also, if you want to shut down your own application programs with SHUTUP, modify SHUTUP.CMD accordingly.

The following is an example of a SHUTUP.CMD file:

```
;THIS SHUTUP.CMD FILE INSTALLS BYE,
;DMO, AND ACC, IF THEY ARE NOT INSTALLED,
;AND ALSO ACTIVATES A COMMAND FILE (NETSTOP)
;TO SHUT DOWN THE NETWORK IF THE NETWORK
;IS ACTIVE.
;
.ENABLE SUBSTITUTION
.IFNINS ...BYE INS $BYE
.IFNINS ...DMO INS $DMO
.IFNINS ...ACC INS $ACC
.IFNACT NETACP .GOTO 20
@LB:[1,2]NETSTOP
.20:
```

For more information on writing indirect command procedures, see the *RSX-11M-PLUS Indirect Command Processor Manual.*

6.6 System Shutdown

The following sections describe the shutdown procedure for RSX-11M-PLUS and Micro/RSX systems. An annotated example of a system shutdown supplements the description.

6.6.1 System Shutdown Procedure

When the delay before system shutdown expires, SHUTUP performs the following functions:

1. Stops the Console Logger (if active)
2. Redirects the console terminal to the TI: of SHUTUP
3. Submits the indirect command file LB:[1,2]SHUTUP.CMD (if present) to the Indirect Command Processor for execution
4. Logs off logged-in terminals
5. Stops the Queue Manager (QMG; if active)
6. Stops Resource Accounting (if active)
7. Stops the Error Log Task (ERRLOG; if active)
8. Checkpoints all read/write commons
9. Deallocates checkpoint space and dismounts devices
10. Halts the processor

After SHUTUP halts the processor, you can either bootstrap another system or resume operation of the current system.

6.6.2 Example of a System Shutdown

The following is an example of a system shutdown:

```
>

RSX-11M-PLUS Shut down program

Enter minutes to wait before shutdown:
Enter minutes between messages:
Enter minutes to wait before disabling logins:
Reason for shutdown (<CR> for none):
OK to shutdown? [Y/N]:


05-MAR-87 16:34 Please finish up, 10 minutes before shutdown -- ALMOND
Reason for shutdown: FIELD SERVICE PM

All further logins are disabled


05-MAR-87 16:39 Please finish up, 5  minutes before shutdown -- ALMOND
Reason for shutdown: FIELD SERVICE PM

05-MAR-87 16:41 Please finish up, 3  minutes before shutdown -- ALMOND
Reason for shutdown: FIELD SERVICE PM

05-MAR-87 16:42 Please finish up, 2  minutes before shutdown -- ALMOND
Reason for shutdown: FIELD SERVICE PM

05-MAR-87 16:43 Please finish up, 1  minute  before shutdown -- ALMOND
Reason for shutdown: FIELD SERVICE PM


05-MAR-87 16:44 System is now shutting down -- ALMOND
Reason for shutdown: FIELD SERVICE PM

QUE /STO:QMG ❶
>
 STOP/ACCOUNTING SHUTUP   ❷
16:44:38  SYSLOG -- 47. Exiting
@LB:[1,2]SHUTUP    ❸
>
>@ <EOF>
>
 ELI /NOLOG  ❹
16:46:20: ERRLOG -- Error Logging stopped
>
 ACS DU0:/BLKS=0.  ❺
ACS -- Checkpoint file now inactive
>
 DMO DU0:/DEV/LOCK=V  ❻
DMO -- System disk being dismounted
DMO -- SYSTEM dismounted from DU0:  *** Final dismount initiated ***

16:46:43  *** DU0: -- Dismount complete
>
SHUTUP operation complete ❼
```

To ɔhut down the system in this example, SHUTUP does the following:

❶ Stops the Queue Manager (QMG).

❷ Stops Resource Accounting.

❸ Invokes the indirect command file LB:[1,2]SHUTUP.CMD.

The @ <EOF> message indicates the end of the command file SHUTUP.CMD.

❹ Runs the task ELI to terminate the Error Log Task (ERRLOG).

❺ Deallocates checkpoint files on DU0.

❻ Dismounts the disk on DU0.

The VIRTUAL option, /LOCK=V, is privileged. When you specify the VIRTUAL option, the DISMOUNT command (MCR command DMO) does not clear the volume as valid and does not spin down the disk. (Refer to the *RSX-11M-PLUS Command Language Manual,* the *RSX-11M-PLUS MCR Operations Manual* or the *Micro/RSX User's Guide, Volume 1* for additional information on the VIRTUAL option.)

❼ Displays a message when SHUTUP has finished executing.

Part II: Resource Monitoring and Control

Chapter 7
Resource Monitoring Display

The Resource Monitoring Display (RMD) is a privileged task that displays information about your system resources. This information includes the active tasks, their location in memory, the amount of memory they occupy, and the available pool space. On video terminals such as VT52s, VT100-series, VT200-series, and Professional series terminals, RMD provides dynamic displays. On hardcopy terminals such as LA120s, RMD provides "snapshot" displays. RMD automatically alters the display format according to terminal type; you do not have to set parameters or rebuild the RMD task image.

7.1 Introduction to RMD

RMD consists of "pages." A *page* contains up to 24 lines of information, which is equal to one screen on cathode-ray tube (CRT) terminals. The pages are divided into two types: display pages and setup pages.

7.1.1 Display Pages

There are eight display pages available:

- Memory (M)
- Active Task (A)
- Task Header (T)
- Help (H)
- I/O Counts (I)
- System Statistics (S)
- General Statistics About a Cache Region (C)
- Detailed Statistics About a Cached Device (D)

After you enter the RMD command, type the letter indicated in parentheses to view the corresponding display page.

7.1.2 Setup Pages

There are five setup pages. Each setup page is associated with a display page. (The Help display page has no associated setup page.) To access a setup page from a display page, press the ESCAPE key on your terminal. The setup page describes and prompts you for setup commands, which alter the content of the information displayed on the associated display page.

Setup parameters stay in effect until you alter them with subsequent setup commands (entered from the setup page or the Monitor Console Routine (MCR) command line).

7.2 Installing and Invoking RMD

Individual users can invoke RMD at their terminals. For large systems that support many terminals, you may want to run RMD continuously on a slaved terminal near the computer. To install and invoke RMD for use by individual users or to run RMD on a slaved terminal, use one of the procedures described in the following subsections.

7.2.1 Installing RMD at Individual Terminals

Privileged users can use the DIGITAL command language (DCL) or MCR command RUN to run an uninstalled version of RMD (LB:[3,54]RMD.TSK or $RMD). However, nonprivileged users cannot use RMD unless a privileged user installs it for them. To install RMD, enter the DCL or MCR command INSTALL at a privileged terminal, as follows:

```
>
```

When you specify a dollar sign ($) in the INSTALL command line, INSTALL searches for the file RMD.TSK in the library and system User Identification Codes (UICs; by default, [3,54] and [1,54], respectively) on device LB:. (For more information on the INSTALL command, see the *RSX-11M-PLUS MCR Operations Manual*, the *RSX-11M-PLUS Command Language Manual*, or the *Micro/RSX User's Guide, Volume 2*.)

7.2.2 Invoking RMD at Individual Terminals

To invoke an installed copy of RMD, enter the RMD command.

Format

```
RMD [page][,setupcommands] . . .
```

Parameters

page

Specifies one of the display page abbreviations (M, A, T, or H, I, S, C, and D). The default page is the Memory display (M).

setupcommands

Specify a list of valid setup commands for the display page that you have selected. The setup commands are the same as those available to you from the setup page associated with the display page. The default setup commands are discussed in Sections 7.3 to 7.10, which describe the content of each display page and how to use setup commands to alter display parameters.

From a privileged terminal, you can invoke an uninstalled copy of RMD by using the RUN command. However, if you plan to alter displays from the MCR command line, you must use an installed copy of RMD.

7.2.3 Running RMD on a Slaved Terminal

To run RMD on a slaved terminal and to allow users to invoke RMD at their own terminals, install RMD twice. Specify a different task image file and a different task name for each installation.

The task image file RMD.TSK (which has been task built at system generation with the task name ...RMD) is located in the library UIC (by default, [3,54]) on device LB:. The task RMD.TSK is the copy of RMD that the system installs for individual users to invoke. Logical unit numbers (LUNs) 1 and 2 in this task image are assigned to the issuing terminal (TI:) so that a user at any terminal can invoke RMD.

To run RMD on a slaved terminal, perform the following steps:

1. From a privileged terminal, create a copy of RMD.TSK in the system UIC on your system disk named RMDEMO.TSK. If your library UIC is [3,54] and your system disk is DR0, use one of the following command sequences:

   DCL Format

   ```
   DCL>
   DCL>
   DCL>
   ```

   MCR Format

   ```
   MCR>
   MCR>
   MCR>
   ```

2. From a privileged terminal or from a command file to be executed at a privileged terminal, install RMDEMO by using one of the following command sequences:

   DCL Format

   ```
   DCL>
   DCL>
   DCL>
   DCL>
   DCL>
   ```

   MCR Format

   ```
   MCR>
   MCR>
   MCR>
   MCR>
   MCR>
   ```

   The parameter ttnn is the terminal at which you want to run RMDEMO.

   This command sequence installs RMD with task name RMDEMO, reassigns LUNs 1 and 2 to the terminal you want to set slaved, slaves the terminal, and invokes the task.

Because the DCL command ASSIGN/TASK (MCR command REASSIGN) alters the task image on disk, you must use two task images rather than simply installing RMD twice with different task names.

7.3 The Help Display

The Help display describes how to select display pages. Other display pages (such as the Memory, Active Task, and Task Header displays) use the entire screen, so they cannot provide you with prompts or information on using the display. If you need to know how to access a setup page from a display page or how to select another display page, press the H key (for Help).

You can also display the Help page directly by entering the following MCR command line:

```
>
```

Example 7–1 shows the Help display page.

Example 7–1: RMD Help Display Page

```
Use the following keys to switch display pages:

A - Active Task Display
C - General Cache Statistics Display
D - Detailed Cache Statistics Display
H - Help Display (this page)
I - I/O Counts Display
M - Memory Display
S - System Statistics Display
T - <ESC> lets you enter a setup command if any setup commands
          are available for that display page
    <SPACE> replots the current display page
    <CTRL-C> or <CTRL-Z> exits RMD
```

Because you cannot modify the parameters for the Help display, there is no corresponding setup page.

7.4 The Memory Display

The Memory display graphically represents the entire system memory, including the approximate size and locations of partitions and active tasks. The display also shows pool statistics, the name of the task that is currently executing, and other information about the status of your operating system. To access the Memory display from another display page, press the M key (for Memory). To access the Memory display from the MCR command line, enter the following command:

```
>
```

If you invoke RMD without specifying a display page, RMD defaults to the Memory display.

The following annotated example shows a "snapshot" of a Memory display. The numbers in Example 7–2 correspond to the numbered items in the list that follows the example.

Example 7–2: Memory Display

```
❶                           ❷          ❸      ❹              ❺
RSX-11M-PLUS V4.0  BL40   (ATHENA)  1024K  UP 001:19:11  22-MAR-87 13:20:24
❻                              ❼
TASK=FRKT32                  FREE=   SY0:107640.  DB1:60608.
                                     DB0:DMO      DB2:22210.        PARS
❽                              ❾                                     ❿
POOL=1688.:1968.:47.         SECPOOL=3483.:4096.:85%            SYSPAR:D
     1688.:1968.:47.                 3483.:4096.:85%            SECPOL:P
⓫                                                                GEN   :D
IN: DRTD        F.TH P NN.                    C
34  ICTU        C.3R 0 TT.                    A
218 RTC:        S.4C 0 ...                    C }  ⓬
OUT 1.0         RAT. L ELS                    H
0   1.M         ET3. . CAE                    E
OK  M.          S.4. . LTA
   !!]!)       !!+>>==!!!+====================================
0*******120*****240*****360*****480*****600*****720*****840*****
EPD----P-------D------------------------------------------------ } ⓭
----------------------------------------------------------------
960*****1080****1200****1320****1440****1560****1680****1800****
===!+>>>>>>>>+>+--+>->>
    .DDDELBBNAMF . AB R                                         ERRSEQ
    .UBBVPAAETAA . TR M                                         28.  ⓮
    .000C2PPT.IL . .U D }  ⓬
    P013. 01AVL$ E VV V
    IFFF.   C5Q$ D 44 5
    A111.   P  $ T
```

This display contains the following information:

❶ Operating system type, version number, and base level.

❷ Six-character name that is a DECnet node name (if the DECnet package is running on your system) or the system name that you selected during system generation (if DECnet is not running on your system).

❸ Size of the system memory in K words.

❹ Time elapsed in units of days, hours, and minutes, since the system was last bootstrapped.

❺ Current date and time.

❻ Name of the task currently executing, or if no task is executing other than RMD, *IDLE* (which indicates that the Executive is executing the idle loop).

Note

> To determine which task is executing, RMD examines the Active Task List (ATL). Instead of displaying the task that is currently executing (which is always RMD, because RMD generates the display), RMD determines which task is scheduled to run immediately after RMD stops executing.

❼ Number of free blocks on the first four Files–11 devices in your system configuration. If a device is dismounted, RMD displays "DMO." If a device is off line, RMD displays "OFL." Magnetic tapes can also be displayed, but only if you previously selected them from the I/O Counts setup page.

❽ Pool information appears in the following format:

```
POOL=X:Y:Z
```

The variables X, Y, and Z are defined as follows:

X Specifies the number of words in the largest free block in pool.

Y Specifies the number of free words in pool.

Z Specifies the number of fragments in the pool free list.

The second line records the maximum pool depletion that has occurred since you invoked RMD. This line is useful if RMD has been running on a slaved terminal since the system was last bootstrapped.

❾ Secondary pool information appears in the following format:

```
SECPOOL=A:B:C%
```

The variable A, B, and C are defined as follows:

A Specifies the number of free blocks in secondary pool.

B Specifies the total number of blocks in secondary pool.

C% Specifies the percentage of free blocks in secondary pool.

The second line records the largest depletion of secondary pool that has occurred since you invoked RMD. This line is useful if RMD has been running on a slaved terminal since the system was last bootstrapped.

⑩ Partitions in memory are displayed in the following format:

`partitionname:type`

partitionname

Specifies the name of the memory partition being displayed. In this example, the partitions are as follows:

Partition Name	Description
SYSPAR	The system partition. Usually reserved for system tasks, such as the Task Termination Notification Program (TKTN).
SECPOL	The secondary pool partition. Note that this partition is extensible (see Chapter 8).
GEN	The general partition. This is the default for most user tasks in memory.

type

Specifies one of the following types of partitions:

Type	Description
D	Dynamic partition
P	Secondary pool partition

⑪ Number of tasks in memory (IN:) and the amount of memory they use (in 1024-word or 1K increments). Also, the number of active tasks swapped or checkpointed out of memory (OUT) and the amount of memory they would require (in 1024-word or 1K increments). In this example, there are 34 tasks in memory, and they use 218K of memory. There are currently no tasks waiting to run.

⑫ Name of each task, common, or driver in memory and its location in memory using the following symbols to designate size, type (task, common, or driver), and other attributes:

Symbol	Attribute
< >	Active task
[ ]	Task not active, yet occupies memory
! !	Named common
+ +	Unnamed common (displayed name is first attached task)
( )	Loaded driver that uses a device mnemonic
---	Task or region not fixed in memory
===	Task or region fixed in memory

The hyphens and equal signs represent the approximate amount of memory that each task, driver, or common occupies. Where the display shows only one delimiter and no hyphens or equal signs, the open delimiter is in the same location as the closing delimiter of the preceding task.

⓭ Partition size and location. The beginning of each partition is marked with the same symbols as those listed previously, plus the following additions:

Symbol	Meaning
E	Executive partition
D	Dynamic partition
P	Secondary pool partition

The lines of asterisks are approximate representations of the amount of memory occupied by each partition. The numbers are in 1024-word or 1K increments. Each numerical character also represents the same amount of memory as an asterisk. RMD always divides the system memory into groups of eight units.

⓮ System error count sequence recorded by the Error Logger (always zero if the Error Logger is not present in your system).

7.4.1 Altering the Memory Display from the Setup Page

To alter the Memory display, access the setup page for the Memory display by pressing the ESCAPE key. The setup page describes and prompts you for commands that you use to alter the Memory display. You can enter multiple commands after each prompt by using commas as separators.

The following setup commands are available for altering the Memory display:

FREEx=ddnn: Specifies the four Files–11 devices to be displayed (see Example 7–2). The variable x is a number from 1 to 4; ddnn specifies a device name and unit number. The default is your system disk SY: and the next three Files–11 devices in the system configuration.

RATE=s Specifies how often RMD replots the Memory display. The variable s is the replot rate in seconds. The default rate is once per second.

This command applies to CRT (video display) terminals only.

You can truncate the command names to 1 character (for example, F1=ddnn:). These setup parameters stay in effect until you alter them, even if you request another display page and then return to the Memory display.

7.4.2 Altering the Memory Display from the MCR Command Line

You can enter the FREE and RATE commands as part of the MCR command line. (The general format of the MCR command line is discussed in Section 7.2.2.) In the following example, RMD displays the number of free blocks on DM0 (instead of the third Files–11 device in your configuration) and replots the display every 3 seconds (instead of every second):

```
>

RSX-11M-PLUS V4.0  BL40   (ATHENA)  1024K  UP 001:19:14  22-MAR-87 13:23:55
TASK=*IDLE*                  FREE=  SY0:107640.  DM0:DM0
                                    DB0:DM0     DB1:60608.         PARS
POOL=1672.:1964.:47.         SECPOOL=3486.:4096.:85%
     1672.:1964.:47.                3486.:4096.:85%               SECPOL:P
                                                                  SYSPAR:D
IN: DRTD        F.TH P NN.                    C                   GEN   :D
35  ICTU        C.3R 0 TT.                    A
202 RTC:        S.4C 0 ...                    C
OUT 1.0         RAT. L ELS                    H
0   1.M         ET3. . CAE                    E
OK  M.          S.4. . LTA
   !!]!)       !!+>>==!!!+======================================
0*******120*****240*****360*****480*****600*****720*****840*****
EPD----P-------D------------------------------------------------
----------------------------------------------------------------
960*****1080****1200****1320****1440****1560****1680****1800****
===!+>>>>>>>>>+>+--+>>>
    .DDDELBBNAMF . ATA                                            ERRSEQ
    .UBBVPAAETAA . TTT                                            28.
    .000C2PPT.IL . .3.
    P013. 01AVL$ E V4V
    IFFF.   C5Q$ D 4 5
    A111.   P  $ T
```

To change the first and second Files–11 devices to DB1 and DB2 respectively, use the following MCR command line:

```
>
```

For example:

```
>

RSX-11M-PLUS V4.0  BL40   (ATHENA)  1024K  UP 001:19:15  22-MAR-87 13:24:28
TASK=F11ACP                  FREE=  DB1:60608.   DB0:DM0
                                    DB2:22210.   DB3:146741.       PARS
POOL=1604.:1856.:48.         SECPOOL=3481.:4096.:84%
     1604.:1856.:48.                3481.:4096.:84%               SYSPAR:D
                                                                  SECPOL:P
IN: DRTD        F.TH P NN.                    C                   GEN   :D
35  ICTU        C.3R 0 TT.                    A
211 RTC:        S.4C 0 ...                    C
OUT 1.0         RAT. L ELS                    H
0   1.M         ET3. . CAE                    E
OK  M.          S.4. . LTA
   !!]!)       !!+>>==!!!+======================================
0*******120*****240*****360*****480*****600*****720*****840*****
EPD----P-------D------------------------------------------------
----------------------------------------------------------------
960*****1080****1200****1320****1440****1560****1680****1800****
===!+>>>>>>>>>+>+--+>>>->
    .DDDELBBNAMF . ATAP                                           ERRSEQ
    .UBBVPAAETAA . TTTI                                           28.
    .000C2PPT.IL . .3.P
    P013. 01AVL$ E V4VT
    IFFF.   C5Q$ D 4 53
    A111.   P  $ T   6
```

These setup parameters stay in effect until you alter them with subsequent setup commands (using the setup page or an MCR command line). Even if you switch to another display page and then return to the Memory Display, RMD does not restore the default setup parameters.

Other RMD pages also allow you to select the devices to be displayed. After you select a device, it becomes common across all the display pages. For example, if you select F1=DM0 for the Memory display page, and then switch to the I/O Counts Display page, RMD displays DM0 as the first device (by default, the first device listed on the I/O Counts page is SY:).

7.5 The Active Task Display

The Active Task display shows you the active tasks in the system. To access this display from another display page, press the A key (for Active Task). Example 7–3 shows a typical Active Task display.

Example 7–3: Active Task Display

```
RSX-11M-PLUS V4.0  BL40    1024K OWN=ALL        22-MAR-87 13:20:29
❶      ❷             ❸       ❹     ❺    ❻
Name   Length   TI     Pri  I/O  Status flags

...LDR 00005000 C00    248. 0.   -CHK  STP -PMD  PRV  NSD
RMDV5  00041100 VT5    225. 0.    RUN -CHK -PMD  REM  PRV
NETACP 00023700 C00    200. 0.    STP  ACP -PMD  PRV  NSD
EVC... 00033200 C00    198. 0.    STP -PMD  PRV
MCR... 00031600 TT2    160. 0.   -CHK  STP -PMD  PRV  CLI  NSD
DB02F1 00014400 C00    149. 0.   -CHK  STP  ACP  REM  PRV  NSD
EM00F1 00014400 C00    149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DB07F1 00014400 C00    149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DR06F1 00014400 C00    149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DB03F1 00014400 C00    149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DR05F1 00014400 C00    149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DB01F1 00014400 C00    149. 0.   -CHK  STP  ACP  REM  PRV  NSD
F11ACP 00014400 C00    149. 0.   -CHK  STP  ACP  PRV  NSD
DB06F1 00014400 C00    149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DU00F1 00014400 C00    149. 0.    RUN -CHK  STP  ACP  REM  PRV  NSD
PMT... 00007100 C00    148. 0.    STP -PMD  PRV
HRC... 00057400 C00    140. 0.    STP -PMD  PRV  NSD
SYSLOG 00020400 C00    130. 0.    STP -PMD  PRV
BAP1   00047600 C00    80.  0.    STP -PMD  PRV  SLV
BAP0   00047600 C00    80.  0.    STP -PMD  PRV  SLV
```

This display contains the following six fields:

❶ Name of the task

❷ Length of the task (size) in octal bytes

❸ Terminal from which the request to run the task was issued

❹ Running priority of the task

❺ Outstanding I/O count

❻ Status flags

The status flags use the same mnemonics as the DCL command SHOW TASKS/ACTIVE/FULL (MCR command ATL). See the *RSX-11M-PLUS Command Language Manual,* the *RSX-11M-PLUS MCR Operations Manual,* or the *Micro/RSX User's Guide, Volume 2* for the definitions of the status flags.

7.5.1 Altering the Active Task Display from the Setup Page

To alter the Active Task display, press the ESCAPE key, which displays the setup page for the Active Task display. The setup page describes and prompts you for commands that you use to alter the Active Task display. You can enter multiple commands after each prompt by using commas as separators. The setup commands available for altering the Active Task display are as follows:

OWNER=ttnn: Displays only those tasks that have been issued by a particular terminal. The parameter ttnn represents the terminal that issued the task. The default is ALL, which displays tasks issued from all terminals.

PRIORITY=p Determines the highest priority of the tasks to be displayed. When the active task list is too long to fit on one screen, use this command to limit the number of tasks that are displayed.

The letter p is a task's running priority. The default is 250, which is the highest possible priority.

RATE=s Specifies how often RMD replots the Active Task display. The variable s is the replot rate in seconds. The default rate is once per second.

This command applies to CRT (video display) terminals only.

TASK=taskname Specifies the name of the task whose header you want to display. Note that the TASK command displays the Task Header display page. (The Task Header display is discussed in Section 7.6.) This is the only setup command that switches display pages.

There is no default for the TASK command.

You can truncate these commands to their shortest unique forms. These setup parameters stay in effect until you alter them, even if you switch to another display page and back to the Active Task display.

7.5.2 Altering the Active Task Display from the MCR Command Line

You can enter the same setup commands discussed in the previous section as part of the MCR command line.

To display only those tasks issued from your own terminal, use the setup command OWNER and specify the terminal TI:. For example:

```
$

RSX-11M-PLUS V4.0  BL40   1024K OWN=TI:         22-MAR-87 13:28:36

Name   Length   TI    Pri  I/O  Status flags

RMDV5  00041100 VT5   225. 0.    RUN -CHK -PMD  REM  PRV
AT.V5  00053500 VT5   64.  0.    SEF  STP  WFR -PMD  REM  PRV
```

Likewise, to display the Active Task display using 200 as the maximum task priority and replotting the display every 3 seconds, use the setup commands PRIORITY and RATE. For example:

```
>

RSX-11M-PLUS V4.0  BL40   1024K OWN=ALL         22-MAR-87 13:29:01

Name   Length   TI    Pri  I/O  Status flags

NETACP 00023700 C00   200. 0.    STP  ACP -PMD  PRV  NSD
EVC... 00033200 C00   198. 0.    STP -PMD  PRV
MCR... 00031600 TT2   160. 0.    RUN -CHK  STP -PMD  PRV  CLI  NSD
DB02F1 00014400 C00   149. 0.   -CHK  STP  ACP  REM  PRV  NSD
EM00F1 00014400 C00   149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DB07F1 00014400 C00   149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DR06F1 00014400 C00   149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DB03F1 00014400 C00   149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DB01F1 00014400 C00   149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DR05F1 00014400 C00   149. 0.   -CHK  STP  ACP  REM  PRV  NSD
F11ACP 00014400 C00   149. 0.   -CHK  STP  ACP  PRV  NSD
DB06F1 00014400 C00   149. 0.   -CHK  STP  ACP  REM  PRV  NSD
DU00F1 00014400 C00   149. 0.   -CHK  STP  ACP  REM  PRV  NSD
PMT... 00007100 C00   148. 0.    STP -PMD  PRV
HRC... 00057400 C00   140. 0.    STP -PMD  PRV  NSD
SYSLOG 00020400 C00   130. 0.    STP -PMD  PRV
BAP1   00047600 C00   80.  0.    STP -PMD  PRV  SLV
BAP0   00047600 C00   80.  0.    STP -PMD  PRV  SLV
QMG... 00033700 C00   75.  0.    STP -PMD  PRV
LP1    00016700 C00   70.  0.    STP -PMD  PRV  SLV
```

For comparison, the following display uses 50 as the maximum task priority:

```
$

RSX-11M-PLUS V4.0  BL40   1024K OWN=ALL         22-MAR-87 13:29:14

 Name  Length   TI    Pri  I/O  Status flags

T34T34 00012500 TT34  50.  0.   -CHK  WFR -PMD  REM
MAILQ  00140400 C00   50.  0.    WFR -PMD  PRV
PIPV4  00105600 VT4   50.  1.    RUN  WFR -PMD  REM
```

By using the setup command TASK, you can display selected tasks. In the following example, RMD displays information about the Queue Manager (QMG):

```
>

RSX-11M-PLUS V4.0  BL40   1024K                22-MAR-87 13:30:12

Task: QMG...   Partition: GEN    Status: STP -PMD PRV
 Own: C00:     I/O: 0.    Dpri: 75.  Pri: 75.  Spri: 70.   Len: 00033700

R0 = 122322  R1 = 123722  R2 = 122722  R3 = 001006  R4 = 000000  R5 = 000000
PC = 120724  PS = 170000  SP = 120406  $DSW = 1.    Eflg = 000000 000000
```

```
LUN File                          LUN File
--- ----                          --- ----
1.  DU1:QUEUE.SYS;0
2.  SP0:
3.  SY0:
4.  LP0:
5.  CO0:
```

If you specify the name of a task that is not currently active, RMD displays the message "Task not in system" and all the display fields are left blank. For example:

```
>

RSX-11M-PLUS V4.0  BL40   1024K                  22-MAR-87 13:29:43
Task not in system

Task: PIPV4   Partition:         Status:
 Own:         I/O:       Dpri:      Pri:      Spri:      Len:

R0 =          R1 =          R2 =          R3 =          R4 =          R5 =
PC =          PS =          SP =          $DSW =        Eflg =

LUN File                          LUN File
--- ----                          --- ----
```

7.6 The Task Header Display

The Task Header display contains the following information about the specified task:

- Name of the task
- Name of the partition in which the task runs
- Status flags, which have the same mnemonics as in the Active Task display
- Owner of the task by terminal number
- Outstanding I/O count
- Default priority
- Running priority
- Swapping priority
- Length, in octal bytes
- Contents of the six general-purpose registers, the program counter (PC), and the Processor Status Word (PSW)
- Contents of the Directive Status Word (DSW)
- Local event flags
- Logical unit number (LUN) assignments to a maximum of 26 LUNs

When RMD displays file names in the list of LUN assignments, the file name is the name of the file when it was created. If the file has been renamed, the display does not reflect the current file name.

If you invoke RMD from the MCR command line without specifying the name of a task, RMD displays the setup page first so that you can specify the task name (see Section 7.6.1). To access this display from another display page, press the T key (for Task Header).

7.6.1 Altering the Task Header display from the Setup Page

To alter the Task Header display, press the ESCAPE key, which displays the setup page for the Task Header display. Then, enter one of the following two commands:

RATE=s — Specifies how often RMD replots the Task Header display. The variable s is the replot rate in seconds. The default rate is once per second.

This command applies to CRT (video display) terminals only.

TASK=taskname — Specifies the name of the task whose header you want to display. There is no default for this command; you must specify a task name.

The setup page describes and prompts you for these commands, as follows:

```
The following commands are available for this page:

RATE=s     to change replot rate to 's' seconds

TASK=t     to display header for task 't'

<CR> to return

COMMAND>
```

After you enter a command at the COMMAND> prompt, RMD displays the prompt again. If you press the RETURN key, RMD displays the Task Header page for the specified task. In the following example, RMD displays the task header for the PMT task:

```
COMMAND>
COMMAND>

RSX-11M-PLUS V4.0  BL40   1024K                 22-MAR-87 13:20:44

Task: PMT...  Partition: GEN     Status: STP -PMD PRV
 Own: CO0:    I/O: 0.    Dpri: 148. Pri: 148. Spri: 143.  Len: 00007100

R0 = 000074  R1 = 000102  R2 = 000000  R3 = 146176  R4 = 140700  R5 = 000000
PC = 122040  PS = 170000  SP = 120212  $DSW = 1.    Eflg = 000100 000000

LUN File                               LUN File
--- ----                               --- ----
1.  CO0:
2.  SY0:
3.  SY0:
4.  SY0:
5.  TI0:
```

You can truncate these commands to their shortest unique forms. These setup parameters remain in effect until you alter them, even if you switch to another display page and back to the Task Header display.

To enter multiple commands after each COMMAND> prompt, separate the commands with commas.

7.6.2 Altering the Task Header Display from the MCR Command Line

Both the RATE and TASK commands are available to you from the MCR command line. To specify a task name or plot rate directly, invoke and enter the command on the same line, as follows:

```
>
```

Taskname is the name of the task whose header you want RMD to display, and s is the replot rate in seconds. For example:

```
>

RSX-11M-PLUS V4.0  BL40   1024K              22-MAR-87 13:31:10

Task: PMT...  Partition: GEN     Status: STP -PMD PRV
 Own: CO0:    I/O: 0.    Dpri: 148. Pri: 148. Spri: 143.  Len: 00007100

R0 = 000074  R1 = 000102  R2 = 000000  R3 = 146176  R4 = 140700  R5 = 000000
PC = 122040  PS = 170000  SP = 120212  $DSW = 1.    Eflg = 000100 000000

LUN File                          LUN File
--- ----                          --- ----
1.  CO0:
2.  SY0:
3.  SY0:
4.  SY0:
5.  TI0:
```

RMD displays the Task Header display for PMT, replotting the page every 2 seconds. Note that you must give the full task name. In this case the task name is "PMT...," not "PMT."

If you do not specify a task name in the MCR command line, RMD displays the setup page. Use the setup page to select a task whose header you want RMD to display (see Section 7.6.1).

7.7 The I/O Counts Display

The I/O Counts display shows you I/O and error logging counts for up to six error logging devices. By default, the first six error logging disk devices in your hardware configuration are displayed. To access this display from another display press the I key (for I/O). To access this display from the MCR command line, type the following:

```
>
```

Example 7–4 displays the I/O counts for the first six error logging devices in an RSX–11M–PLUS system.

Example 7–4: I/O Counts Display

```
RSX-11M-PLUS V4.0  BL40                                 22-MAR-87 11:42:43

Device    I/O Count    I/O      Cylinders   Words               Error
Name                   Per Sec  Per I/O     Per Sec             Count/limit

SY0:         60698.     4.00       2.50     1024.00  last          0/100    soft
                        2.18       2.08      558.54   avg          0/100    hard

DB0:      Dismounted                                 last                   soft
                                                      avg                   hard

DB1:         31262.     0.00       0.00       00.00  last          2/100    soft
                        0.46       0.00       89.82   avg          0/100    hard

DB2:          1175.     0.00       0.00        0.00  last          0/100    soft
                        0.00       0.00        0.00   avg          0/100    hard

DB3:          6786.     0.00       0.00        0.00  last          0/100    soft
                        0.00       0.00        0.00   avg          0/100    hard

DB4:      Dismounted                                 last                   soft
                                                      avg                   hard
```

The I/O Counts display contains the following information for each device:

- Device name of the displayed device
- Total number of I/O requests to the device
- Number of I/O requests issued to the device in the last second
- Average number of I/O requests issued to the device per second
- Average number of cylinders crossed per I/O request during the last second
- Average number of cylinders crossed per I/O request
- Number of words transferred during the last second
- Average number of words transferred per second
- Count and limit of soft I/O errors
- Count and limit of hard I/O errors

The I/O Counts display works only on video terminals.

7.7.1 Altering the I/O Counts Display from the Setup Page

To alter the I/O Counts display, you press the ESCAPE key, which displays the setup page for the I/O Counts display. The setup page lists the available commands you can use to alter the I/O Counts display and prompts you for a command. You can enter multiple commands after each prompt by using commas as separators. The setup commands available for altering the I/O Counts display are as follows:

DEVICEx=ddnn: Selects the error logging devices to be displayed. The letter x is a number from 0 to 6 and ddnn is an error logging device. By default, RMD displays the first six error logging disk devices in your hardware configuration.

You can select magnetic tape devices; however, RMD does not display fields 5 and 6 for tapes.

See the *RSX-11M-PLUS and Micro/RSX Error Logging Manual* for a list of valid error logging devices.

RATE=s Determines how often RMD replots the I/O Counts Display. The letter s is the replot rate in seconds. The default replot rate is once per second.

This command applies to CRT (video display) terminals only.

You can truncate these commands to their shortest unique forms. These setup parameters stay in effect until you alter them, even if you switch to another display page and then return to the I/O Counts display.

7.7.2 Altering the I/O Counts Display from the MCR Command Line

You can enter the DEVICE and RATE commands as part of the MCR command line. For example, when you enter the following command line, RMD displays I/O count information for DR5 and replots the display every 2 seconds:

```
>
```

The information for DR5 is displayed in the position that would normally display information for the second error logging device in your hardware configuration. For example:

```
>

RSX-11M-PLUS V4.0  BL40                                 22-MAR-87 13:40:03

Device     I/O Count    I/O      Cylinders  Words              Error
Name                    Per Sec  Per I/O    Per Sec            Count/limit

SY0:          70385.     0.00       0.00       0.00  last         0/100    soft
                         0.42      91.26     130.84   avg         0/100    hard

DR5:           6657.     0.00       0.00      00.00  last         0/100    soft
                         0.00       0.00       0.00   avg         0/100    hard

DB1:          31421.     0.00       0.00      00.00  last         3/100    soft
                         0.00       0.00       0.00   avg         0/100    hard

DB2:           1175.     0.00       0.00       0.00  last         0/100    soft
                         0.00       0.00       0.00   avg         0/100    hard

DB3:           6786.     0.00       0.00       0.00  last         0/100    soft
                         0.00       0.00       0.00   avg         0/100    hard

DB4:       Dismounted                                last                  soft
                                                      avg                  hard
```

Likewise, to change the first and second device display positions to DB1 and DB2, respectively, use the following MCR command line:

```
>
```

These setup parameters stay in effect until you alter them by using setup commands from the setup page. Even if you switch to another display page, and then return to the I/O Counts display, RMD does not restore the default parameters.

7.8 The System Statistics Display

The System Statistics display shows you general information about the operation of your system. Much of this information comes from Resource Accounting; therefore, this display is useful only if Resource Accounting was included when your system was generated and is currently active. To access this display from another display, press the S key. To access this display from the MCR command line, type RMD S.

Example 7–5 displays the System Statistics page for an RSX–11M–PLUS system.

Example 7–5: System Statistics Display

```
RSX-11M-PLUS V4.0  BL40                        22-MAR-87 13:21:46

Total tasks run:        4907.                  Current tasks:          37. ❶
Total logons:             57.                  Current users:           9. ❷
Shuffler runs:             0.  ❸
Error sequence:          147.  ❹

System pool:      1520.:1884.:49.              Memory utilization:      58%  ❺
Secondary pool:   3483.:4096.:85%              Checkpoint utilization:  0%   ❻

Item                        Total               Last          Avg/Sec

Directives               1376284.  ❼
QIOs                      616028.  ❽
CPU ticks                 682430.  ❾
Zero CPU intervals        396464.  ❿
Checkpoints                    0.  ⓫
```

This display contains the following information:

❶ Total number of tasks run and the number of currently active tasks

❷ Total number of times users have logged in to the system and the number of users currently logged in

❸ Number of times the Shuffler has executed (see Chapter 8 for more information on the Shuffler task)

❹ Error sequence count

❺ Primary pool information (in the same format as in the Memory display) and the percentage of system memory used

❻ Secondary pool information and the percentage of system checkpoint files used

❼ Total number of directives issued, the number of directives issued during the last second, and the average number of directives issued per second

❽ Total number of QIOs issued, the number of QIOs issued during the last second, and the average number of QIOs issued per second

❾ Total number of running central processing unit (CPU) *ticks* (clock intervals), the number of running CPU ticks during the last second, and the average number of running CPU ticks per second

❿ Total number of zero CPU intervals, number of zero CPU intervals during the last second, and the average number of zero CPU intervals per second

⓫ Total number of task checkpoints, number of task checkpoints during the last second, and the average number of task checkpoints per second

Note that the last and average (Avg/Sec) values for the last five fields do not appear in this example because the example was recorded on a hardcopy terminal (an LA12).

7.8.1 Altering the System Statistics Display from the Setup Page

You can alter the replot rate on the System Statistics display by pressing the ESCAPE key, which displays the setup page for the System Statistics Display. The setup page lists the command format for altering the replot rate and prompts you for a command.

The RATE=s setup command alters the replot rate of the System Statistics display. The variable s is the replot rate in seconds. The default rate is once per second.

The replot rate parameter stays in effect until you alter it, even if you switch to another display page and then return to the System Statistics display.

7.8.2 Altering the System Statistics Display from the MCR Command Line

You can enter the RATE command as part of the MCR command line. For example, when you enter the following MCR command line, RMD replots the System Statistics Display once every 2 seconds:

```
>
```

7.9 The General Statistics About a Cache Region Display

The General Statistics About a Cache Region display (RMD C display) shows you general statistics about a particular cache region. (However, you must have previously established disk data caching through the MOUNT or SET command to use this display. For more information on disk data caching, see Chapter 15 of this manual.) To access this display from another display, press the C key (for Cache Region). To access this display from the MCR command line, type RMD C.

Example 7–6 displays general cache statistics about the CACHE region.

Example 7–6: Cache Region Display (General Statistics)

```
RSX-11M-PLUS V4.0  BL40     Cache Statistics (General)       22-MAR-87 13:17:11

Cache Region Name: CACHE      Region Size: 47440 (2500. disk blocks)

Device          Total Hit  Fail Load      Total  Hit  Fail Defer     Total    Cache
Name            Reads Rate Rate Rate      Writes Rate Rate Rate      I/O Ops  Used
DB6:           58834.  92%   4%   2%      19393.  62%   0%   0%       78227.   96%
DB7:               0.   0%   0%   0%          0.   0%   0%   0%           0.    0%

Total          58834.  92%   4%   2%      19393.  62%   0%   0%       78227.   96%
```

The RMD C Display contains the following information about each cache region:

- The name of the cache region. By default, the name of the region is CACHE.
- The actual memory size (in octal) of the cache region in 32-word memory blocks, followed in parentheses by the decimal number of disk blocks available in the extent storage area. (The memory size includes the private cache pool area as well as the extent storage area.)
- The name of the device or devices for which caching is enabled.
- The total count of all types of read requests issued to that device. These include VIRTUAL, READAHEAD, DIRECTORY, LOGICAL, and OVERLAY read requests.
- The percentage of all read requests that were satisfied by the cache (the Read Hit Rate).
- The percentage of all read requests that could not be satisfied by the cache (the Read Fail Rate).
- The percentage of all read requests that resulted in data being loaded into the cache (the Load Rate).
- The count of all write requests issued to the device. These include VIRTUAL, DIRECTORY, and LOGICAL write requests.
- The percentage of all write requests that caused updating of blocks already loaded in the cache (the Write Hit Rate).
- The percentage of all write requests that could not be directed to the cache because of an extent overlap (the Write Fail Rate).
- The percentage of all write requests to temporary files that were deferred from write-through to the disk (the Defer Write Rate). The actual write operation to the disk was performed sometime after the user's write request was completed.
- The total of all read and write operations for the device.
- The percentage of the cache extent storage area used by the device.
- The grand totals for the cache region.

For more information on the RMD C display, see Chapter 15.

7.9.1 Altering the RMD C Display from the Setup Page

To alter the RMD C display, press the ESCAPE key, which displays the setup page for the General Statistics about a Cached Region display. The setup page lists the available commands you can use to alter the display and prompts you for a command. You can enter multiple commands after each prompt by using commas as separators. The setup commands available for altering this display are as follows:

REGION=name   Specifies the cache region to be displayed. The default region is CACHE.

RATE=s   Allows you to determine how often RMD replots the general statistics about a cache region. The variable s is the replot rate in seconds. The default rate is once per second.

This command applies to CRT (video display) terminals only.

You can truncate these commands to their shortest unique forms. (For this display page, the shortest unique form is RA for RATE and RE for REGION.) These setup parameters stay in effect until you alter them, even if you switch to another display page and then return to the RMD C Display.

7.9.2 Altering the RMD C Display from the MCR Command Line

You can enter the REGION and RATE commands as part of the MCR command line. For example, when you enter the following command line, RMD displays general statistics for the region named CACHE and replots the display every 2 seconds:

```
>

RSX-11M-PLUS V4.0  BL40      Cache Statistics (General)       22-MAR-87 13:33:13

Cache Region Name: CACHE       Region Size: 47440 (2500. disk blocks)

Device         Total Hit  Fail Load       Total  Hit  Fail Defer     Total    Cache
Name           Reads Rate Rate Rate       Writes Rate Rate Rate      I/O Ops  Used
DB6:          60951.  92%   4%   2%       19594.  62%   0%   0%       80545.   99%

Total         60951.  92%   4%   2%       19594.  62%   0%   0%       80545.   99%
```

For more information on the RMD C display, see Chapter 15.

7.10 The Detailed Statistics About a Cache Region Display

The Detailed Statistics About a Cache Region display (RMD D display) shows you detailed statistics about a particular cache region. (However, you must have previously established disk data caching through the MOUNT or SET command to use this display. For more information on disk data caching, see Chapter 15 of this manual.) To access this display from another display, press the D key (for Detailed Statistics.) To access this display from the MCR command line, type RMD D.

Example 7–7 displays detailed cache statistics about the CACHE region.

Example 7-7: Cache Region Display (Detailed Statistics)

```
RSX-11M-PLUS V4.0  BL40     Cache Statistics (Detailed)      22-MAR-87 13:17:22

Device Name:  SYO:                Region Name: CACHE     Region Size: 47440
Cache Status: Active              Requests Being Cached: Dir,Ovr,Vir,Log,Rdh

              Virtual  Readahead   Directory     Logical     Overlay       Total
Reads          14951.         0.     24398.       4842.      14676.      58867.
Read Hit Rate     87%         0%          93%         92%         95%         92%
Read Load Rate     3%         0%           1%          4%          4%          2%

Read Overlap       0%         0%           5%          1%          0%          2%
Extent Too Big     0%         0%           0%          0%          0%          0%
Max Extent Size 127.        127.           3.        127.        127.

Writes          8965.                  10380.         48.                 19393.
Write Hit Rate    33%                     86%         70%                     62%
Write Overlap      0%                      0%          0%                      0%

Total I/O      23916.         0.     34778.       4890.      14676.      78260.

Primary Pool Allocation Failure Rate (as a % of Total I/Os):       0%
Cache Pool Allocation Failure Rate (as a % of Total Read I/Os):  2%
Read Load Failure Rate (as a % of Cache Load I/Os):                0%
Deferred Write Rate (as a % of Total Write I/Os):                  0%
```

The RMD D Display contains the following information about each cached device:

- The name of the cache region.
- The region size (in octal, measured in 32-word memory blocks.)
- Whether the cache is ACTIVE, ENABLED, DEFER, or a combination of all three.
  - ACTIVE indicates that the device is being cached through the region.
  - ENABLED indicates that the device will automatically be cached when it is mounted (even if the MOUNT command does not explicitly request that the device be cached).
  - DEFER indicates that deferred write requests to temporary files are enabled for the device.
- The types of cache I/O operations enabled for the device. The types can be VIRTUAL (VIR), READAHEAD (RDH), DIRECTORY (DIR), LOGICAL (LOG), and OVERLAY (OVR). However, note that values for RDH are not currently recorded; RMD displays them as zeros.
- The number of read requests issued for each of the I/O types listed in the display, as well as the total of all read requests.
- The Read Hit Rate, Read Load Rate, Read Overlap Rate, and the Extent Too Big rate for each type of request. Each of these rates is computed as a percentage of the total number of read operations for the particular type of request.
- The maximum extent size for each of the I/O types.
- The number of write requests issued for each of the I/O types and the total of all write requests. Note that READAHEAD and OVERLAY write requests do not exist, so these fields are blank.

- The Write Hit Rate and Write Overlap Rate for each type of I/O request. Each of these rates is computed as a percentage of the total number of write operations of a particular type.
- The total number of each type of I/O request, as well as the total number of all I/O operations.
- The following rates, which are not accumulated by I/O type:
  - The Primary Pool Allocation Failure Rate line shows how often the allocation cannot be done as a percentage of the total number of I/Os issued for that device.
  - The Cache Pool Allocation Failure Rate is similar to the Primary Pool Allocation Failure Rate, except that allocation from the cache pool is only required on read operations. Therefore, the rate is computed as a percentage of the total number of read operations.
  - The Read Load Failure Rate is the percentage of all cache load attempts that failed because of an I/O error.
  - The Deferred Write Rate is the rate at which write requests to temporary files were able to be deferred. This is computed as a percentage of the total number of write operations.

  All cache I/O requests require that an I/O packet be allocated from primary pool.

For more information on the RMD D display, see Chapter 15.

7.10.1 Altering the RMD D Display from the Setup Page

To alter the display, press the ESCAPE key, which displays the setup page for the Detailed Statistics About A Cached Region display. The setup page lists the available commands you can use to alter the display and prompts you for a command. You can enter multiple commands after each prompt by using commas as separators. The setup commands available for altering this display are as follows:

DEVICE=ddnn: Specifies the cached device about which detailed statistics are to be displayed. The default device is SY:. Note that you can display only one device at a time.

RATE=s Determines how often RMD replots the detailed statistics about a cache region. The variable s is the replot rate in seconds. The default rate is once per second.

This command applies to CRT (video display) terminals only.

You can truncate these commands to their shortest unique forms. These setup parameters stay in effect until you alter them, even if you switch to another display page and then return to the RMD D display.

7.10.2 Altering the RMD D Display from the MCR Command Line

You can enter the DEVICE and RATE commands as part of the MCR command line. For example, when you enter the following command line, RMD displays detailed statistics for DB6 and replots the display every 2 seconds:

```
>

RSX-11M-PLUS V4.0  BL40     Cache Statistics (Detailed)      22-MAR-87 13:33:40

Device Name:  DB6:                Region Name: CACHE     Region Size: 47440
Cache Status: Active              Requests Being Cached: Dir,Ovr,Vir,Log,Rdh

             Virtual   Readahead   Directory     Logical     Overlay       Total
Reads          15103.          0.     24812.       5235.      15826.      60976.
Read Hit Rate    87%          0%         93%         93%         95%         92%
Read Load Rate    3%          0%          1%          3%          4%          2%

Read Overlap      0%          0%          5%          1%          0%          2%
Extent Too Big    0%          0%          0%          0%          0%          0%
Max Extent Size 127.        127.          3.        127.        127.

Writes          9049.                 10497.         48.                  19594.
Write Hit Rate   33%                     86%         70%                     62%
Write Overlap     0%                      0%          0%                      0%

Total I/O      24152.          0.     35309.       5283.      15826.      80570.

Primary Pool Allocation Failure Rate (as a % of Total I/Os):       0%
Cache Pool Allocation Failure Rate (as a % of Total Read I/Os):    2%
Read Load Failure Rate (as a % of Cache Load I/Os):                0%
Deferred Write Rate (as a % of Total Write I/Os):                  0%
```

These setup parameters stay in effect until you alter them by using setup commands from the setup page. Even if you switch to another display page, and then return to this display, RMD does not restore the default parameters.

7.11 Modifying the Task-Build Command File (RSX-11M-PLUS Systems Only)

To change the defaults for RMD on an RSX-11M-PLUS system, you can modify the task-build command file RMDBLD.CMD in directory [1,24] on device LB:. For more information, see the descriptions contained within the file.

Note

This option is available only to conventional RSX-11M-PLUS systems; pregenerated RSX-11M-PLUS (RL02 distribution kits) and Micro/RSX systems do not include the command file RMDBLD.CMD.

7.12 RMD Error Messages

RMD generates the following error messages:

RMD—Allocated screen buffer too small for this device

Explanation: RMD requires more internal memory to display the requested display on the type of terminal on which you are running RMD.

User Action: Rebuild RMD with a larger screen buffer by modifying the task-build command file as discussed in Section 7.11.

RMD—Illegal command—xxxxx

Explanation: You entered an illegal command xxxxx either on the MCR command line or in response to the COMMAND> prompt on a setup page.

User Action: Enter the correct command as described in this chapter.

RMD—Page does not exist

Explanation: You requested a display page from the MCR command line that does not exist.

User Action: Reenter the command line with a correct display mnemonic.

RMD—Segment 'xxxxxx' not found

Explanation: The module xxxxxx was not found in the task image for RMD. This denotes an error in how RMD was task built.

User Action: Check that RMD has been built with all the required modules.

RMD—Terminal type not defined

Explanation: The operating system and RMD do not recognize your terminal type.

User Action: Check your terminal type by using the DCL command SHOW TERMINAL (MCR command SET /TERM=TI:). If this setting is incorrect, use the DCL or MCR command SET to correct it. If the setting is correct, use a different type of terminal.

RMD—Terminal type not set

Explanation: You did not build RMD to display the requested display page on the type of terminal to which your terminal is set.

User Action: Check your terminal type by using the DCL command SHOW TERMINAL (MCR command SET /TERM=TI:). If this setting is incorrect, use the DCL or MCR command SET to correct it. If the setting is correct, use a different type of terminal.

RMD—Terminal type not yet supported

Explanation: RMD does not recognize your terminal type.

User Action: Check your terminal type by using the DCL command SHOW TERMINAL (MCR command SET /TERM=TI:). If this setting is incorrect, use the DCL or MCR SET command to correct it. If the setting is correct and you are using an RSX-11M-PLUS system, you can write a terminal characteristics module for the terminal similar to the module in the file [14,10]V52CHA.MAC. Then, assemble the module, include the new module in the ODL file, and rebuild RMD.

Chapter 8

Memory Management

This chapter describes the tasks supplied with the RSX–11M–PLUS operating system for managing the allocation and availablility of memory resources. It includes a description of the Pool Monitor Task (PMT), the secondary pool, and the Shuffler (SHF) task.

8.1 Evaluating System Requirements for Dynamic Memory

During its execution, the RSX–11M–PLUS Executive has varying needs for dynamic memory. Instead of allocating a fixed amount of memory for each requirement, the Executive makes use of a dynamically allocated memory space called dynamic storage or *pool*. When allocated space is no longer needed, the Executive releases it back to the pool of available memory.

Pool requirements for a system are dependent upon the configuration, application, and degree of system loading. Enough pool must be available to satisfy peak demands; otherwise, a degradation in system performance occurs.

Because nearly all Executive functions require pool, a system can exhaust pool when system activity is heavy. This can happen if too many tasks are installed, if too many volumes are mounted, or if a number of other conditions are present. When this happens, the system does not appear to have failed, but it is not functioning normally (for example, if there are data lights on the front of the processor, they will be flickering, but the system may not accept input). Under these conditions, the system often cannot display error messages because the Executive requires pool space to perform I/O. Once a system exhausts pool, you may need to restart the system.

You can avoid this condition in two ways. First, if you have an RSX–11M–PLUS system, take the maximum amount of pool compatible with your system configuration during system generation. The command file SYSVMR.CMD does this for you automatically during system generation. Second, use PMT (RSX–11M–PLUS systems only) and the Resource Monitoring Display (RMD) to monitor pool, and take appropriate action before pool gets too low. (For more information on RMD, see Chapter 7.)

8.2 Pool Monitoring Support

Pool monitoring support controls the use of the system's pool resources. This support monitors pool levels, restricts use of pool, and notifies you when pool is near depletion. PMT also responds to extreme fragmentation. However, due to the seriousness of this pool problem, PMT requires your interaction to aid in system recovery. To ensure that PMT functions correctly, respond to PMT's prompts (see Section 8.5.1).

Pool monitoring support is included in the system by default. You do not need to select any option; system generation establishes the support for you. Pool monitoring support consists of two parts: the Executive pool monitor code and the privileged Pool Monitor Task (PMT). However, note that PMT is only available with RSX-11M-PLUS systems; it is not included with Micro/RSX distribution kits.

8.2.1 The Executive Pool Monitor Code

The pool monitor code within the Executive monitors the amount of free pool and detects major pool events. When a major pool event occurs, the Executive notifies PMT of the event and of the current condition of pool.

The mechanism used by the Executive to detect pool events is called the *pool monitor interface.* Briefly, the interface functions as follows:

- Provides the Executive with the ability to declare information to PMT.
- Provides the Executive with the capability to receive a response or acknowledgement from PMT. Then, the Executive can use the acknowledgement to control further information displays.

The details of how this interface works are beyond the scope of this manual. However, it is fully described in the source code for the Executive module CORAL.MAC. CORAL.MAC is in the directory [11,10] on all RSX-11M-PLUS systems, except those supplied as RL02 distribution kits (that is, as pregenerated systems).

8.2.2 The Pool Monitor Task (RSX-11M-PLUS Systems Only)

The Pool Monitor Task (PMT) is supplied on your RSX-11M-PLUS distribution kit, or it can be task built and installed during the system generation procedure. Whenever the resultant system is bootstrapped, PMT is activated by the Executive.

If you decide not to use PMT, you can write and run your own pool monitor task. However, before you write your own task, you should be familiar with the Executive pool monitor interface. On all RSX-11M-PLUS systems, except those supplied as pregenerated systems (RL02 distribution kits), the interface is described in the Executive module [11,10]CORAL.MAC. Any task with the task name PMT... is considered to be a "pool monitor task."

PMT monitors pool conditions. The conditions that PMT responds to include those detected and reported by the Executive (see Section 8.4). PMT also monitors pool fragmentation. Through this monitoring, PMT can respond to another major pool event and react to extreme pool fragmentation. The other major pool event (not reported by the Executive) occurs when PMT detects that the largest free pool fragment is less than the minimum required fragment size. (See Section 8.5 for further discussion of extreme fragmentation.)

Once it has been activated by the Executive, PMT operates in a cycle. PMT determines the size of the largest pool fragment whenever it receives the Executive's notification of a pool condition, or at least at regular time intervals. It then compares this size to the minimum required fragment size. Based on the Executive's report on the current pool condition and what PMT detects when monitoring fragmentation, PMT determines what the current state of pool is and responds to it accordingly.

The minimum required fragment size is set by using the Monitor Console Routine (MCR) or Virtual Monitor Console Routine (VMR) command SET /PLCTL, or the DIGITAL Command Language (DCL) command SET SYSTEM/POOL/LIMITS. See Section 8.6.2 for more information on the minimum required fragment size.

8.3 Pool Conditions

The condition of pool is directly related to the acceptable high and low pool limits. (To set and manipulate pool limits, see Section 8.6.1). When the size of free pool crosses the high or low limit, a new pool condition exists. The possible pool conditions are low pool and high pool.

A *high pool condition* exists when the amount of free pool is sufficient to support current and less demanding system workloads. A *low pool condition* exists when the amount of free pool is nearing the point where system performance will degrade if the workload on the system increases.

Major pool events cause either a high or low pool condition to exist. PMT responds to pool fragmentation and to pool conditions; however, the response varies depending upon the current level of pool fragmentation. A high or low pool condition, combined with a specific pool fragmentation level, creates either a high or low pool *state*. The levels that influence PMT's response are as follows:

- If the largest fragment is equal to or greater than the minimum required fragment size, then PMT acts only on the condition of pool as reported by the Executive.
- If the largest fragment becomes smaller than the minimum required fragment size, PMT considers this to be a low pool event. If the system is not already in a low pool state, this event causes a transition to that state. PMT then responds accordingly.

Although extreme fragmentation is another pool event that PMT responds to, it is different from those events previously described (see Section 8.5). To determine the overall state of pool, PMT uses information concerning the current level of fragmentation and information provided by the Executive. The following subsections describe specific pool events and conditions and PMT's response to them.

8.3.1 Pool Level and Fragment Size Are Satisfactory

The Executive notifies PMT that the pool level is above the high limit and PMT detects that fragment size is greater than the minimum required fragment size. PMT concludes that a high pool state exists and executes actions appropriate for a high pool state (see Section 8.4.2).

8.3.2 Low Pool Condition

The Executive notifies PMT that a low pool condition exists.

PMT determines whether the change in pool reported by the Executive is temporary. The change is temporary if, by the time PMT responds to the Executive's notification, enough free pool has accumulated to push the total above the high limit. If the change is temporary, PMT ignores the notification and continues to execute high pool state actions. If the change is not temporary, PMT concludes that a low pool state exists and executes actions appropriate for a low pool state (see Section 8.4.1).

8.3.3 High Pool Condition

The Executive notifies PMT that a high pool condition exists.

PMT determines whether the change in pool reported by the Executive is temporary. The change is temporary if, by the time PMT responds to the Executive's notification, the total amount of free pool has moved below the high limit. If the change is temporary, PMT ignores the notification and continues to execute low pool state actions. If the change is not temporary, PMT initiates actions appropriate for a high pool state (see Section 8.4.2).

8.3.4 Largest Pool Fragment Is Too Small

PMT detects that the largest pool fragment is smaller than the minimum required fragment size.

PMT concludes that a low pool state exists and executes actions for a low pool state. Note that this low pool state overrides any pool conditions reported by the Executive. This pool event will cause the system to be in a low pool state even if the Executive reports a high pool condition.

8.3.5 Largest Pool Fragment Becomes Larger

PMT detects that the size of the largest pool fragment has become larger than the minimum required fragment size.

PMT decides that the current pool state must be based on the latest pool condition reported by the Executive. If the Executive reported a high pool condition, PMT decides that a high pool state exists. If the Executive reported a low pool condition, PMT decides that a low pool state exists. In either situation, PMT takes the appropriate action for that state.

8.4 Major Pool Events

The Executive pool monitor code responds to three major pool events. Two are classified as low pool events and one is classified as a high pool event. The occurrence of these events directly affects the condition of pool, creating either a high or low pool state. The Executive responds by notifying PMT of the pool event and the resultant pool condition. The Executive also activates PMT, so PMT is responsible for any further response to the pool condition.

8.4.1 Low Pool Event

A *low pool event* occurs when one of the following conditions exist:

- The total amount of free pool falls below the *low pool limit.* (To set the low pool limit, use the MCR or VMR command SET /PLCTL, or the DCL command SET SYSTEM/POOL /LIMITS. For more information on these commands, see Section 8.6.1.)
- The largest pool fragment is too small for the process requesting it (a pool allocation failure occurs).

Note

A pool allocation failure is not the same as pool fragmentation. For more information on pool fragmentation, see Section 8.3.

When a low pool event occurs, the system enters a *low pool state.* In response, PMT does the following:

- Disables checkpointing for itself.
- Sets its internal cycle interval to 20 seconds.
- Prevents nonprivileged users from logging in.
- Suppresses INSTALL/RUN/REMOVE sequences on nonprivileged terminals.
- Sends a warning message to all logged-in terminals. For example:

  ```
  08-AUG-87 13:35:10 -- Low pool -- Please exit active tasks - ZEPHYR
  ```

  In this example, if the DECnet package is not running on your system, ZEPHYR is the system name selected during system generation. If the DECnet package is running on your system, ZEPHYR is the DECnet node name.

  PMT sends this message at 20-second intervals until it concludes that a high pool state exists. Users cannot suppress the message with the DCL command SET TERMINAL /NOBROADCAST (MCR command SET /NOBRO). However, the effectiveness of this message depends on whether users exit from their active tasks. If they choose to ignore this message, a low pool state continues to exist.

- Sends a warning message to and displays pool statistics at the console terminal. For example:

  ```
  08-AUG-87 13:35:20 -- Warning -- Pool is critically low
  Total free pool = 1116. bytes
  Largest fragment = 112. bytes
  ```

  Following the warning message, PMT displays the total amount of free pool and the largest free fragment of pool existing at the onset of the low pool state. PMT broadcasts this message at 20-second time intervals until it concludes that a high pool state exists.

This recovery procedure continues until the total amount of free pool rises above the high pool limit and the largest pool fragment is large enough for the process requesting it (that is, until a *high pool event* occurs).

8.4.2 High Pool Event

A *high pool event* occurs when a system that has a low pool condition regains enough space to cause the amount of free pool to rise above the high pool limit. When the system attains a high pool state, its response to a low pool event is complete; it no longer needs to actively regain pool space.

When PMT concludes that a high pool state exists, it executes the following actions:

1. Sets its internal cycle interval to 1 minute
2. Enables checkpointing for itself
3. Removes any pool use controls

PMT then stops itself so that its impact on system resources is minimized. PMT restarts itself when the internal cycle interval has elapsed.

To set the high pool limit, use the MCR or VMR command SET /PLCTL, or the DCL command SET SYSTEM/POOL/LIMITS. For more information on these commands see Section 8.6.1.

8.5 PMT Actions During Extreme Fragmentation

Extreme fragmentation exists when the largest pool fragment is too small for any additional dormant tasks to become active. In this case, pool is exhausted; you may not be able to enter any commands to free pool space (including the ABORT command). At this point, PMT takes immediate action to prevent total pool depletion by executing the following steps:

1. Enters kernel mode, which prevents any other tasks from executing but allows the system to respond to device interrupts.
2. Prevents the Task Termination Notification Program (TKTN) from being invoked. (Invoking TKTN at this time depletes the available pool.) It does this by clearing the Task Control Block (TCB) pointer for TKTN, which is stored within the Executive.
3. Sends a warning message to the console terminal.
4. Prompts you to abort one or more of the tasks displayed. (You must be at the console to free pool in response to the PMT prompts.) After you select a task to abort, PMT returns to user mode until the abort operation is finished.
5. Displays a message at the terminal running the task and at the console terminal when a task is successfully aborted.
6. Displays at the console terminal (upon request) an updated list of abortable tasks.

8.5.1 PMT Messages During Extreme Pool Fragmentation

When extreme pool fragmentation occurs, PMT displays a message and a list of abortable tasks at the console terminal, as follows:

```
dd-mmm-yy hh:mm:ss -- Warning -- Free pool exhausted
Abortable tasks in memory:

tttttt   P   I/O=xxx.    ttnn:
```

Display Fields

dd-mmm-yy

Displays the current date (for example, 22-Jul-87).

hh:mm:ss

Displays the current time in hours (24-hour time), minutes, and seconds.

tttttt

Displays the name of the abortable task.

P

Indicates a privileged task (a blank space indicates a nonprivileged task).

xxx

Displays the total outstanding I/O count for the task.

ttnn

Displays the terminal from which the task is currently running.

PMT lists the tasks in order of decreasing priority but does not display the priority of individual tasks. Then, PMT prompts you to select one of the tasks from the list, as follows:

```
Enter a task to abort, or press RETURN to exit:
```

The abortable task list includes only those tasks that PMT considers eligible to abort. A task is abortable if it meets the following requirements:

- It must be active.
- It must be resident in memory.
- It cannot be privileged and running from the console terminal.
- It cannot be a command line interpreter (CLI) task, an Ancillary Control Processor (ACP), or the task loader.
- It cannot have any internal conditions that prevent it from exiting upon an abort request (for example, it is already aborted).

Aborting tasks releases pool resources for use by the remaining active tasks and allows the system to continue operation. However, use discretion when selecting tasks to abort; aborting certain tasks could render all or part of your application unusable. Therefore, only abort tasks that consume pool and are relatively expendable (such as the Peripheral Interchange Program, PIP, or the MACRO-11 assembler, MAC).

By default, PMT does not include the privileged tasks running from the console terminal in the abortable task list. Therefore, if you do not want PMT to consider a privileged task abortable, run the task from the console terminal.

PMT may also display the following messages during extreme fragmentation:

Task not installed, try again:
Task not aborted, try again:

Explanation: The task you specified is not installed, or it is not on the list of abortable tasks.

User Action: If you press the LINE FEED key in response to either prompt, PMT displays the list of abortable tasks. If you press the RETURN key, PMT returns to user mode for at least 7 seconds, giving the system an opportunity to recover from the low pool state.

Task not aborted, try again:

Explanation: The task is listed as abortable, but the task exited before PMT was able to abort it.

User Action: Enter another task name.

"tttttt" aborted

Explanation: PMT sends this message to the console terminal after it successfully aborts a task (taskname tttttt).

User Action: No user action is required. This is an informational message.

08-AUG-87 13:35:10—Warning—Free pool exhausted
"tttttt" aborted

Explanation: If the task was not running from the console terminal, PMT displays these messages at the terminal from which the task was running.

User Action: No user action is required. This message informs the user running a task that the task has been aborted.

Press LINE FEED for an updated task list,
enter a task to abort, or press RETURN to exit:

Explanation: PMT sends this message to the console terminal after it aborts a task.

User Action: Enter the name of a task to be aborted, or press the RETURN key to exit.

8.5.2 Annotated Example of PMT Output for Extreme Fragmentation

An annotated example of PMT output for extreme fragmentation is shown in Example 8–1. The numbers in the example correspond to the numbered items in the list that follows the example.

Example 8–1: PMT Output for Extreme Fragmentation

```
8-AUG-87 13:42:23 -- Warning -- Free pool exhausted ❶

Abortable tasks in memory:  ❷

RMDEMO   P   I/O=  0.   TT0:
COT...   P   I/O=  0.   TT0:
AT.T30   P   I/O=  1.   TT30:
EDIT14       I/O=  1.   TT14:
MAIT20       I/O=  1.   TT20:
TEST         I/O=  0.   CO0:
PIPT21       I/O=  1.   TT21:
MACT47       I/O=  1.   TT47:

Enter a task to abort, or press RETURN to exit:        ❸

"PIPT21" aborted  ❹

Press LINE FEED for an updated task list,
enter a task to abort, or press RETURN to exit:     ❺

Abortable tasks in memory:  ❻

RMDEMO   P   I/O=  0.   TT0:
COT...   P   I/O=  0.   TT0:
AT.T30   P   I/O=  1.   TT30:
EDIT14       I/O=  1.   TT14:
MAIT20       I/O=  1.   TT20:
TEST         I/O=  0.   CO0:
MACT47       I/O=  1.   TT47:

Enter a task to abort, or press RETURN to exit:     ❼

Task not installed, try again:         ❽

"EDIT14 aborted  ❾

Press LINE FEED for an updated task list,
enter a task to abort, or press RETURN to EXIT:      ❿
```

❶ Sends a warning message to logged-in terminals.

❷ Displays the list of abortable tasks at the console terminal.

❸ Prompts for a task to abort (PIPT21 is entered).

❹ Aborts PIPT21.

❺ Prompts for an updated task list, a task to abort, or to exit. (Press the LINE FEED key for an updated task list.)

❻ Displays the list of abortable tasks.

❼ Prompts for a task to abort (EDT is entered).

❽ Indicates that a task with the name EDT is not installed, and requests that you try again (EDIT14 is entered).

❾ Aborts EDIT14.

❿ Prompts for an updated task list, a task to abort, or to exit. (Press the RETURN key to exit.)

8.6 Setting Pool Limit Parameters

The DCL command SET SYSTEM/POOL/LIMITS (MCR command SET /PLCTL) sets the pool limit parameters used by the Executive pool monitor code and PMT. The syntax, parameters, and parameter limits for the VMR command are discussed in Chapter 5. For a description of the MCR command, see the *RSX-11M-PLUS MCR Operations Manual;* for the DCL command, see the *RSX-11M-PLUS Command Language Manual.* Guidelines for setting the four command parameters are discussed in the following subsections.

To determine whether the pool limits need to be changed during a low pool state, compare the current values with the information sent to the console terminal (see Section 8.4.1).

8.6.1 Setting Low and High Pool Limits

The first two parameters of the SET /PLCTL command establish the low and high pool limits. The Executive pool monitor code uses these limits to determine when major pool events occur.

In general, you should set the low limit value to the point where your system's workload cannot increase without the risk of a pool resource deadlock. The appropriate value for this limit varies with each system. You can determine the best value by experimenting with various limits. The default value for the low pool limit is 600_{10} bytes.

Set the high limit value to reflect the type of use your system receives. The high limit parameter controls the point where a low pool state is transformed into a high pool state. The high limit defaults to 1600_{10} bytes.

If pool use fluctuates considerably, a limit set much higher than the low limit can reduce the number of pool state transitions. If pool use is typically constant, a high limit set close to the low limit shows critical changes in the pool state more frequently. Since pool needs and use vary, you should experiment with this parameter to determine the optimum value.

8.6.2 Setting Pool Fragment Size

This parameter establishes the minimum size of the largest pool fragment. PMT uses this limit to determine when pool fragmentation problems exist. When the largest pool fragment is the minimum size or greater, a high pool state exists. A low pool state exists when the largest pool fragment is smaller than the minimum size. By default, the lowest permissible size of the largest pool fragment is 200_{10} bytes. Experiment with this parameter to determine the optimum value.

Fragmentation is one of the more frequent and less desirable pool problems. Thus, the value that you specify for this parameter can have an immediate impact on successful pool monitoring.

8.6.3 Setting Base Priority for Nonprivileged Tasks

This parameter establishes the base priority for nonprivileged tasks. The optimum value is application specific, but the default is 51_{10}. During a low pool state, PMT uses this parameter to restrict newly requested nonprivileged tasks from competing for memory resources (see Section 8.4.1). Only those tasks with a priority less than the specified base priority are restricted in this way.

8.7 PMT Task-Build Options

By altering PMT's task-build file PMTBLD.CMD and rebuilding the task image, you can modify the following features:

- Timing control
- PMT default Actions
- Pool use controls

Each of these features affect PMT's response to pool conditions.

To include a modified version of PMT in the system permanently, use VMR to change the task image, and then reboot the system. To run a modified version of PMT in the current system, use DCL or MCR to remove the previous version and then install the new version.

The steps for modifying PMT with VMR are as follows:

1. Use the VMR command REMOVE to remove PMT (the task name is PMT...) from the System Task Directory (STD).
2. Edit the task-build file (PMTBLD.CMD contains information on modifying PMT).
3. Task build PMT.
4. Use the VMR command INSTALL to install the modified task image.
5. Reboot the system to execute the modified version of PMT.

The steps for modifying PMT with DCL or MCR are as follows:

1. Abort PMT.
2. Use the DCL or MCR command REMOVE to remove PMT from the STD.
3. Edit the task-build file.
4. Task build PMT.
5. Use the DCL or MCR command INSTALL to install the modified version of PMT.
6. Use the DCL or MCR command RUN to run PMT.

The following sections describe the task-build options in more detail.

8.7.1 Controlling PMT Timing

PMT operates in a cycle, running at regular intervals to monitor fragmentation and stopping after the monitoring is done. When the Executive interrupts the cycle to report a new pool event, PMT resets the cycle to begin at the point at which it was interrupted.

PMT's cycle can have one of two possible time lengths, depending on the current state of pool. If a low pool state exists, the cycle lasts 20 seconds (plus the time PMT requires to execute the low pool state actions). If a high pool state exists, the cycle lasts 60 seconds (plus the time PMT requires to execute the high pool state actions).

To reduce the number of PMT cycles during a high pool state, lengthen the time it takes for each cycle to complete. (The 60-second value is the default as well as the minimum value for the cycle.) However, note that if you increase this value, PMT may not be as effective at monitoring pool fragmentation. To compensate for this problem, use the DCL command SET SYSTEM/POOL/LIMITS (MCR or VMR command SET /PLCTL), which increases the minimum size of the largest pool fragment parameter (see Section 8.6.2). Increasing the minimum fragment size increases PMT's chances of detecting pool fragmentation problems.

8.7.2 Controlling PMT Action

You can modify some of the actions that PMT executes during the cycle. The actions are as follows:

- Sending warning messages to logged-in terminals
- Checkpointing stopped tasks in memory
- Aborting certain privileged tasks

Sending Warning Messages to Logged-in Terminals

When a low pools state exists, PMT sends warning messages to logged-in terminals. However, by modifying PMTBLD.CMD, you can suppress these messages.

Checkpointing Stopped Tasks in Memory

When a low pool state exists, PMT requests the Executive to force checkpointing of all memory-resident tasks that have been stopped without outstanding nonbuffered I/O. (This includes tasks with outstanding buffered I/O, CLI tasks, and ACP tasks.) When the Executive checkpoints a task, it deallocates the task header, thereby reducing fragmentation.

You can modify PMTBLD.CMD to force checkpointing of memory-resident stopped tasks during a high pool state. On systems with a large number of tasks that do not use external headers, this reduces the effect of long-term fragmentation. Although this can increase the effective use of available pool space, it may also cause a noticeable increase in disk I/O as tasks are checkpointed. In addition, note that CLI tasks, ACP tasks, and tasks with outstanding buffered I/O are always exempt from checkpointing while a high pool state exists (even if you modify PMTBLD.CMD).

Aborting Certain Privileged Tasks

During extreme fragmentation, PMT displays a list of abortable tasks and prompts for tasks to abort. By default, PMT does not list or abort privileged tasks running from the console terminal. You can override this default by modifying PMTBLD.CMD. However, there are exceptions to this option. These exceptions (CLI tasks, ACP tasks, and the task loader) cannot be listed as abortable tasks.

8.7.3 Controlling Pool Use

When a low pool state is in effect, PMT executes the default pool use controls described in Section 8.4.1. You can suppress any of these controls by modifying PMTBLD.CMD and by task building PMT again.

8.8 Aborting PMT

To terminate PMT, abort the task with one of the following methods:

- Issue an ABORT command from a privileged terminal.
- Issue the Executive directive Abort Task (ABRT$) from a privileged task.

PMT then performs the following actions before it exits:

- Restores a Task Control Block (TCB) pointer for the Task Termination Notification program (TKTN), if PMT previously cleared the pointer and TKTN is currently installed (see Section 8.4.2). This restores normal TKTN operations.
- Prevents the Executive's pool monitor code from requesting PMT after PMT exits. PMT clears a TCB pointer for PMT... that was previously set by the DCL, MCR, or VMR INSTALL command.
- Cancels any outstanding pool use controls that were invoked by a low pool state.
- Displays the following message at the console terminal:

  ```
  Remove and reinstall PMT to restore pool monitoring
  "PMT..." aborted
  ```

  This indicates that the TCB pointer for PMT has been cleared so that the Executive can no longer run PMT. To restore the pointer, use DCL or MCR to remove and reinstall PMT.

8.9 Secondary Pool

Primary pool (usually referred to simply as "pool") is part of the Executive's partition in memory and contains the Executive's database. *Secondary pool* is allocated from main memory. Unlike primary pool, secondary pool is extensible. However, when you extend secondary pool, you do so at the expense of main memory. Each time you increase the amount of secondary pool space, you decrease the amount of available memory. As a system manager, you need to create an appropriate balance between the two resources.

The system requires secondary pool to function normally. In particular, it uses secondary pool to contain the following data structures:

- TCBs for prototype tasks (tasks named ...xxx or xxx$$$)
- Error Logging packets
- Terminal Unit Control Block (UCB) extensions, for instruction and data space (I- and D-space) systems
- UCB extensions for mass-storage devices
- Named directory context blocks
- Extended logical name data structures
- MCR command lines
- Executive send/receive data packets

- The map section of file windows (a stub file is kept in primary pool)
- Resource Accounting account blocks (System Account Blocks (SABs), User Account Blocks (UABs), and Task Account Blocks (TABs))

All other data structures are located in primary pool.

Certain conditions can deplete secondary pool rapidly. For example, a disconnected terminal can generate noise that MCR interprets as a steady stream of input characters. MCR is unable to send error messages as rapidly as it receives the input data, so the data packets accumulate in secondary pool. Similarly, if someone leans on one of the terminal keypad keys for an extended period of time, the escape sequences sent to MCR can create an overload of command line packets.

When secondary pool becomes very low, nonprivileged users cannot log in to the system. (The system permits privileged users to log in, although the accounting system does not record their system usage.) Users (privileged or nonprivileged) who are already logged in to the system cannot enter commands. Files and logical names become inaccessible, and devices cannot be placed on line. If the number of free blocks in secondary pool reaches zero, rebooting the system is the only way for you to regain control.

To avoid this situation, monitor secondary pool usage. If necessary, increase the total size or availability of secondary pool (by removing installed tasks, for example). The subsections that follow describe how to monitor, extend, and increase the availability of secondary pool.

8.9.1 Monitoring Secondary Pool Usage

The DCL command SHOW SECONDARY_POOL (MCR command SET /SECPOL) displays secondary pool use on the system.

Display Format

```
SECPOL=secfr:secsiz:pctfr
```

secfr

Displays the number of free blocks in secondary pool in units of 32_{10}-word blocks.

secsiz

Displays the total size of secondary pool in units of 32_{10}-word blocks.

pctfr

Displays the percentage of free blocks in secondary pool.

Example

```
DCL>
SECPOL=285.:640.:44%
```

```
MCR>
SECPOL=285.:640.:44%
```

Displays the amount of secondary pool currently being used.

8.9.2 Displaying Secondary Pool Partitions

The DCL command SHOW PARTITIONS (MCR command PAR) displays a description of each memory partition in the system, including secondary pool (SEC POOL). The display includes the following information:

- Partition name (SECPOL)
- Address of the Partition Control Block (PCB)
- Starting address of the partition (in octal)
- Size of the partition (in octal)
- Partition type (SEC POOL)

Use this command to determine whether or not a secondary pool partition is currently in use. (For more information on the SHOW PARTITIONS command, see the *RSX-11M-PLUS Command Language Manual.* For information on the PAR command, see the *RSX-11M-PLUS MCR Operations Manual.*)

Formats

```
DCL>SHOW PARTITIONS
```

```
MCR>PAR[TITIONS]
```

Examples

```
>
SYSPAR 117734 00174400 00010000 MAIN
       065230 00174400 00010000 TASK <TKTN >
DRVPAR 117624 00216400 00140000 MAIN
       117340 00216400 00003100 DRIVER (DB:)
       117274 00221500 00001300 DRIVER (DK:)
       117164 00226100 00001300 DRIVER (DS:)
       117120 00227400 00004000 DRIVER (DM:)
    .
    .
    .
LDRPAR 117560 00356400 00002600 MAIN
       116054 00356400 00002600 TASK   <...LDR>
SECPOL 117450 00421200 00022000 SEC POOL
GEN    117404 00443200 04734600 MAIN
       115454 00443200 00070600 TASK   <F11ACP>
       073400 00534000 00006000 TASK   <TT36 >
       052274 00542600 00003500 RO COM +...EDI+
       060774 00571400 00020100 TASK   <QMG...>
       037210 00620100 00036100 TASK   <HRC...>
```

Displays the PCB address as 117450, the starting address for the secondary pool partition as 421200_8, and the total size of the partition as $22{,}000_8$ bytes.

8.9.3 Extending Secondary Pool

Loading additional secondary pool into memory increases the availability of secondary pool, but it reduces the amount of memory available for tasks and common regions. To avoid an unacceptable decrease in system performance, add secondary pool in small increments and monitor the system after each addition.

8.9.3.1 Using the LOAD Command

The LOAD /EXP=SEC command creates an additional secondary pool subpartition (called a *region*; see Chapter 1) in the GEN partition. You can enter the LOAD /EXP=SEC command many times, increasing the amount of secondary pool slightly each time. Note that the LOAD command does not alter the SECPOL partition; the size of SECPOL is determined when the partition is created during system generation (see Section 8.9.5).

To decrease the size of secondary pool after you have extended it with the LOAD command, you must rebootstrap the system. You cannot unload secondary pool with the UNLOAD command.

Note

> The LOAD command does not initiate checkpointing to free a space in a partition.

Format

```
LOAD /EXP=SEC [/SIZE=parsize]
```

Keywords

/EXP=SEC

Specifies the name of the extended Executive partition (SEC.TSK) to be loaded into memory. By default, SEC is loaded into the partition.

/SIZE=parsize

Specifies the minimum size (in words) of the partition into which secondary pool is to be loaded and creates a memory region of this size.

The value of parsize is rounded up to the next highest unit of 100_8 words and must be less than $100,000_8$ words.

Examples

```
>
```

Loads 500_8 additional words of secondary pool into a region in the GEN partition.

```
DCL>
SECPOL 117734 00120000 00040000 SEC POOL
SYSPAR 117670 00160000 00274700 MAIN
       117624 00160000 00104600 RO COM !DIR11M!
       117270 00264600 00004100 TASK   <...LDR>
       116664 00270700 00031600 TASK   <MCR...>
              .
   .
   .
GEN    114040 00454700 03323100 MAIN
       113500 00531700 00040000 RO COM !FCSRES!
       056520 00604500 00014400 TASK   <F11ACP>
       060624 01043000 00022000 TASK   <SYSLOG>
              .
   .
   .
       057350 03600000 00001000 RW COM !SEC.PL!
```

Displays the total secondary pool size as $41,000_8$ bytes. The SECPOL partition is $40,000_8$ bytes; the region of secondary pool added to the GEN partition is 1000_8 bytes. (See the previous description of the SHOW PARTITIONS and PAR commands.)

8.9.3.2 Modifying the Configuration Data File, SYSPARAM.DAT

The SECONDARY_POOL parameter in the configuration data file SYSPARAM.DAT specifies the amount by which secondary pool is increased when the system startup procedure executes. On pregenerated RSX–11M–PLUS and Micro/RSX systems, you can edit the file LB:[1,2]SYSPARAM.DAT and change the value specified for the SECONDARY_POOL parameter. Then, restart the system to obtain the increase (or decrease) in secondary pool.

Format

```
SECONDARY_POOL=n
```

n

Specifies the number of 32-word blocks by which secondary pool is increased. To load additional secondary pool, increase the value for n. To unload secondary pool, decrease the value for n.

The default is SECONDARY_POOL=0; no additional secondary pool is loaded.

8.9.4 Increasing the Availability of Secondary Pool

Depending on your application, increasing the total size of secondary pool may not be appropriate. If your requirements for memory generally exceed your requirements for secondary pool, then you may decide to increase the availability of secondary pool (instead of the overall size).

By using DCL (or MCR) commands, you can create free pool space, as follows:

- Use the SHOW TASKS/INSTALLED (TAL) and SHOW TASKS/ACTIVE (ATL) commands to determine which tasks are currently using secondary pool. By default, when you install a prototype task (that is, a task with a name in the form ...xxx or xxx$$$), the Task Control Block (TCB) is placed in secondary pool. When the tasks are run, the system copies the TCBs into primary pool.

If you use the REMOVE command to eliminate some of the installed prototype tasks, the space that was previously required for the TCBs is freed for other uses.

- Use the SHOW ASSIGNMENTS/ALL (ASN /ALL) and SHOW LOGICALS/ALL (DFL /ALL) commands to display all the logical assignments for your system, group, and current terminal session. If possible, delete some of the assignments with the DEASSIGN (ASN or DFL) command.
- Use the Resource Accounting POOL_RESERVE parameter to limit the use of secondary pool. The parameter POOL_RESERVE allows you to alter the secondary pool limit when you start Resource Accounting. The format is as follows:

  The value parameter specifies the number of blocks of secondary pool (in decimal) that Resource Accounting must leave free after each allocation. If Resource Accounting cannot leave this amount free, the allocation fails. This prevents the swamping of secondary pool.

  The default for value is a quarter of the total secondary pool size. Use of the default is recommended.

  By modifying the secondary pool limit, you can restrict the amount of secondary pool used by Resource Accounting. If you do not use Resource Accounting, deactivate it by entering the STOP /ACCOUNTING command.

  For more information on Resource Accounting, see Chapter 10.

8.9.5 Saving Secondary Pool (RSX-11M-PLUS Systems Only)

After you load additional secondary pool into memory (using the LOAD command), you may want to make the increase in secondary pool permanent. You can do this in one of two ways:

- Re-create the system image with VMR
- SAVE the current system image

If the system image file is too small, use the VMR command SET /PAR to place secondary pool in a lower portion of memory, or use the Peripheral Interchange Program (PIP) to make the system image file larger. Then, run VMR again to reinitialize the file. For more information on the correct procedure, see the *RSX-11M-PLUS System Generation and Installation Guide.*

The SAVE command copies the current RSX-11M-PLUS operating system image (the contents of main memory) into the system image file from which the current system was booted. (The system image file is a special task image file named, by convention, RSX11M.SYS.) The command saves the image so that a hardware bootstrap or the BOOT command can later be used to reload and restart it. In addition, the SAVE command saves secondary pool, which contains prototype TCBs.

There is no size restriction; the size of the system image file determines the maximum size that can be saved. However, the system image file must be contiguous and it must be large enough to contain all of secondary pool.

For a description of the SAVE command format and keywords, see the *RSX-11M-PLUS MCR Operations Manual.*

8.10 Memory Compaction—The Shuffler

The Shuffler (SHF) is a privileged task designed to compact space in partitions when a memory allocation failure occurs. This section describes the Shuffler structure, ways to monitor the Shuffler, and the Shuffler algorithms for the operating system.

8.10.1 Executive and Memory Usage

The Shuffler task image is approximately 700_{10} words long and is installed during system generation. By default, it is built for the partition SYSPAR, which is also used by the MCR dispatcher and the Task Termination Notification program (TKTN). To obtain better performance with a heavily loaded system, fix the Shuffler in its own partition. Then, the Shuffler does not have to compete for memory with MCR and TKTN.

The Shuffler is useful on a system with few tasks loaded in memory, because such a system only requires the Shuffler intermittently. In contrast, if your system is heavily loaded with tasks, the Shuffler may execute too often (due to frequent memory allocation failures). However, you can control the frequency at which the Shuffler executes by modifying a timer in the Executive common region.

The word $SHFTM in the Executive system common region serves as a timer. Each time the Executive requests the Shuffler, this timer is checked. If the timer registers zero, the Shuffler can be requested; otherwise, the request for the Shuffler is ignored.

The amount of time (in clock ticks) that elapses before the Shuffler is requested again is stored in the system common area $SHFCT. When a request to the Shuffler is successful, $SHFTM is initialized for the new interval and the time interval is stored in $SHFCT.

As a privileged user, you can modify the value in $SHFCT by using the MCR command OPEN. To change $SHFCT, obtain a copy of the Executive map file RSX11M.MAP that is produced during system generation. Use the map to determine the address of the global symbol $SHFCT. Then, use the OPEN command to examine or to modify the value. If the value is set too high, fragmentation may become a problem because the Shuffler is not requested often enough. If the value is set too low (or to zero), system response may decrease because the Shuffler is being requested too frequently.

8.10.2 Monitoring the Shuffler

During peak system usage periods, several memory allocation failures may result in repeated requests to the Shuffler, which produces a noticeable decrease in response time. To monitor the Shuffler's effect on your system, use the Resource Monitoring Display (RMD). RMD is a privileged task that displays information concerning task activity on the operating system. (For more information on RMD, see Chapter 7.)

To monitor Shuffler activity with RMD, dedicate a video terminal to running the RMD display. Monitor the memory display page for the following information:

- The currently executing task. *IDLE* indicates that no task is executing.
- The number of active tasks in memory and the amount of memory they occupy
- The number of active tasks checkpointed (on disk) and the total memory they would require

If the Shuffler is active, tasks are moving down in the partition, and the currently executing task field registers the Shuffler as the active task. (Occasionally, even though the Shuffler registers as the active task, tasks cannot be seen moving down in the partition.)

To verify that tasks are not moving when the Shuffler is executing, check the number of active tasks checkpointed (on disk) and the number of tasks waiting for memory. If this count remains the same over a period of time, the Shuffler is executing but cannot create enough space for waiting tasks. If this happens, the Shuffler is consuming system resources and should be aborted.

If the Shuffler task continues to consume system resources, remove the Shuffler from your system with the DCL or MCR command REMOVE.

8.10.3 The Shuffler Algorithms

When a memory allocation failure occurs, the Executive requests the Shuffler. The Shuffler searches memory for a partition that has at least one region waiting for space. When the Shuffler locates such a partition, it attempts to move (shuffle) regions in the partition to create the contiguous area necessary to load waiting regions.

The Shuffler shuffles common regions created by Executive memory management directives. Because the system maintains an outstanding I/O count on a region-by-region basis, the Shuffler can determine whether outstanding I/O is pending. When there is no I/O pending, common regions can be shuffled.

The Shuffler uses two algorithms to move regions in memory. Each algorithm executes several steps repeatedly until the partition reaches a stable state.

The First Pass Algorithm

In the first pass, the Shuffler attempts to move each region to the base of the partition. When regions are moved to the base of the partition, free space is created at the top of the partition so that a waiting region can enter memory. Shuffling occurs in the following steps:

1. The Shuffler calls the Executive to unconditionally checkpoint all task regions of checkpointable tasks that have been stopped or blocked with a command line interpreter (CLI) command.
2. The Shuffler attempts to move each region to the base of the partition and blocks each executable task from further execution. If a region has outstanding I/O, the Shuffler waits for the outstanding I/O count to reach zero. If the I/O count does not reach zero in approximately 0.5 second, the Shuffler marks the region as having outstanding I/O and restarts the first pass algorithm (step 1).
3. After shuffling is complete, each task mapped to the region is unblocked and resumes execution.

Unless you fix tasks in dedicated partitions, fixed tasks can also be shuffled.

If fragmentation is eliminated at the end of the first pass and there is enough space available, then a region may be loaded into memory. However, space may not be available if there are several regions already loaded in memory, or if fragmentation is dividing the partition into two or more *partition sections*. A *partition section* is a subset of the area in a partition. Each partition section is flanked on at least one side by a region that cannot be shuffled. For example, if a device driver is loaded in the middle of a partition, that partition is divided into two partition sections.

If the Shuffler encounters a region that cannot be shuffled during the first pass, it shuffles the regions that can be moved to the base of the partition section. The Shuffler does not shuffle regions around a region that cannot be moved. The free space in each partition section cannot be merged with the free space in another partition section; however, it can be merged at the top of each section.

If memory space is not available for waiting regions or if partition sections remain after the first pass, the Shuffler initializes the second pass algorithm.

The Second Pass Algorithm

The second pass algorithm is executed once for each partition section within a partition. This pass attempts to create a reverse priority list of regions that can be checkpointed by the waiting task.

During the second pass, the Shuffler creates a list in ascending priority order of the regions in each partition section. Then, the Shuffler compares the size of the waiting region with the sum of free space in the partition section, plus the total of one or more lower priority, checkpointable regions. If the waiting region is smaller, the Shuffler tells the Executive to checkpoint the lower priority region or regions.

Checkpointing the lower priority region or regions creates available space in memory, but the space may not be contiguous. The Executive can initiate the first pass algorithm to accumulate the free space in the partition section and to allocate it to the waiting region. As long as there are regions in a partition that cannot be moved, memory stays divided into two or more partition sections.

If the Shuffler completes the second pass without finding room for a waiting region, the Shuffler searches for the next partition with waiting regions. When no other regions are waiting for memory space, the Shuffler exits.

Chapter 9

The Console Logger (RSX-11M-PLUS Systems Only)

The Console Logger consists of a driver (CODRV) and a task (COT...). Together, they control I/O to the console output device (pseudo device CO:) and record time-stamped system messages on a terminal, in a log file, or both. You select support for Console Logging during system generation. Then, you enter Monitor Console Routine (MCR) SET commands from a privileged terminal to start and stop Console Logging, to disable the log file or the console terminal, and to reassign the log file and console terminal.

9.1 The Console Output Device

Your system has a console output device (CO:) and a console terminal. When the Console Logger is active, I/O to CO: is controlled by the console output task (COT...) and the console driver (CODRV). When the Console Logger is not active, CO: is a pseudo device; I/O to CO: is controlled by the terminal driver. In both cases, output to CO: can be forwarded to a terminal that you assign as the console terminal.

9.1.1 Using CO: as a Pseudo Device

When the Console Logger is not active, CO: is a pseudo device. The terminal driver controls all I/O to CO: that is redirected to the console terminal. If you start Console Logging and then decide you want to use CO: as a pseudo device, you can stop Console Logging by specifying the option =OFF with the MCR command SET /COLOG (see Section 9.4.3).

You can redirect CO: to any terminal on your system by using the DCL command ASSIGN/REDIRECT (MCR command REDIRECT). The ASSIGN/REDIRECT (REDIRECT) command redirects all I/O requests previously directed to one physical or pseudo device unit (such as CO:) to another physical or pseudo device unit (such as a terminal).

Format

```
DCL>ASSIGN/REDIRECT oddnn:=nddnn:

MCR>REDIRECT nddnn:=oddnn:
```

Parameters

oddnn

Specifies the old device unit from which requests have been redirected.

nddnn

Specifies the new device unit to which subsequent requests will be directed.

For more information on the ASSIGN/REDIRECT command, see the *RSX-11M-PLUS Command Language Manual* or the *Micro/RSX User's Guide, Volume* 2. For information on the REDIRECT command, see the *RSX-11M-PLUS MCR Operations Manual.*

9.1.2 Using CO: with the Console Logger

When the Console Logger is active, the console driver (CODRV) and the console output task (COT...) control all I/O to CO:. If you stop Console Logging and then decide you want to use CO: with the Console Logger, you can start Console Logging by specifying the option =ON with the MCR command SET /COLOG (see Section 9.4.2).

Console Logging allows you to record messages sent to CO: in a log file. You can record messages on the console terminal, in the log file, or both. When you do not need an online display of Console Logging, you can record messages in the log file and leave the terminal free for a different use. (For information on the corresponding commands, see Section 9.4.)

Prior to sending the first message after a change of date, the Console Logger sends the current time and date. The Console Logger also time-stamps messages to CO: that have not been time-stamped by the task issuing the message. The time appears in the following format:

```
hh:mm:ss
```

For example, if a message was sent at 5 minutes and 5 seconds after 1 o'clock in the afternoon, the time-stamp would appear as follows:

```
13:10:05
```

Note that the hours are in shown in 24-hour time.

9.2 Installing COT... and Loading CODRV

On RSX-11M-PLUS operating systems, if you select Console Logging during system generation, the system generation procedure installs the console output task (COT...) and loads the console driver (CODRV); the necessary commands are included in the command file LB:[1,54]SYSVMR.CMD. Pregenerated RSX-11M-PLUS (RL02 distribution kits) and Micro/RSX operating systems install COT... and load CODRV during the system startup procedure.

However, if one of the following two conditions exists, you need to reinstall COT... and reload CODRV:

1. You have removed CODRV, COT..., or both, and you want to restore Console Logging.
2. You have altered SYSVMR.CMD to eliminate the commands for installing COT... and for loading CODRV.

To install COT..., use the DCL, MCR, or VMR INSTALL command, as follows:

```
>
```

To load CODRV, use the MCR or VMR command LOAD, as follows:

```
>
```

For more information on the DCL and MCR commands, see the appropriate command language manuals. For a description of the VMR commands INSTALL and LOAD, see Chapter 5.

9.3 Communicating with the Console Logger

You can use MCR commands to communicate with the Console Logger directly. Because COT... is a privileged task, you must issue the commands from a privileged terminal. Console Logging commands start and stop Console Logging, assign the console terminal, and specify the console log file. These commands are processed by the COT... task.

There are no DCL equivalents for the Console Logging commands.

Table 9–1 summarizes the Console Logging commands. Section 9.4 describes each Console Logging command in detail.

Table 9–1: Summary of Console Logging Commands

Command	Function
SET /COLOG	Displays the current console terminal and log file assignments.
SET /COLOG=ON	Starts Console Logging.
SET /COLOG=OFF	Stops Console Logging.
SET /COLOG/COT[ERM][=TTnn:]	Enables the console terminal or changes the console terminal assignment.
SET /COLOG/NOC[OTERM]	Disables the console terminal.
SET /COLOG/LOG[FILE][=[filespec]]	Enables the console log file or changes the logfile assignment.
SET /COLOG/NOLOGFILE	Disables the log file.

You can also communicate with the Console Logger by running tasks that use the driver CODRV. Tasks communicate with CODRV by issuing Executive directive Queue I/O requests (QIO$). CODRV handles I/O to the console device CO: for read and write requests.

Tasks can attach to the console device; however, the attach does not provide the task with exclusive access to CO: (it is considered a null operation or no-op). All other I/O functions to the console device are passed to the terminal driver as if they were issued directly to the console.

The subsections that follow describe task communication with the Console Logger.

9.3.1 Writing to CO:

Tasks can issue QIO$ requests to print system messages on the console terminal or to write system messages in the console log file. For example, HELLO issues a QIO$ to print a message to the console terminal each time a user logs in to the system.

Messages to CO: can begin with a CTRL/G sequence (that is, a bell sound) to ensure that COT... sends the message to the most recently assigned console terminal. If you disable the console terminal by specifying the /NOCOTERM option, and you want the message to go to the most recently assigned console terminal, begin the message by pressing CTRL/G. (For a description of the /NOCOTERM option, see Section 9.4.5.)

The buffer in COT... accepts messages not longer than 256_{10} bytes. Note that the MCR command SET /BUFFERSIZE, which sets the buffer size of device CO:, does not affect this restriction. (For more information on the SET /BUFFERSIZE command, see the *RSX-11M-PLUS MCR Operations Manual.*)

9.3.2 Reading from CO:

Tasks can issue QIO$ requests to read input from the console terminal. As a result, you can use the console terminal to respond interactively to system prompts.

When messages are read from the console terminal, COT... does not record the messages in a log file.

9.4 Console Logging Command Descriptions

Console Logging commands perform the following functions:

- Display the current console terminal and log file assignments.
- Start and stop Console Logging.
- Change the console terminal and log file assignments.
- Disable the console terminal or the log file.

Use the MCR command SET for all the Console Logging command formats.

General Format

```
SET /COLOG[MCRoption]...
```

Keyword

/COLOG

Specifies the Console Logging keyword.

Parameter

MCRoption

Specifies one of the Console Logging functions. You can truncate the options to three letters.

9.4.1 Displaying Current Console and Log File Assignments

To display the current console terminal and log file assignments, enter the MCR command SET /COLOG without specifying a Console Logging function, as follows:

```
>
```

Use the preceding command to determine if the Console Logger is active. When the Console Logger is active, COT... displays the current console terminal and log file assignments. For example:

```
>
COT --
Console = TT000:
Logfile = DB000:[1,4]CONSOLE.LOG;1
```

If the Console Logger is not active, COT... displays the console and log file assignments as "None." For example:

```
COT --
Console = None
Logfile = None
```

9.4.2 Starting Console Logging (SET /COLOG=ON)

To start Console Logging, specify the option =ON with the command SET /COLOG.

Format

```
SET /COLOG=ON
```

When you start Console Logging, COT... retains the current console terminal assignment. If you have just bootstrapped the system, COT... assigns CO: to the terminal TT0.

The default location and name of the log file is LB:[1,4]CONSOLE.LOG. The COT... task always creates a new version of the default log file when you start Console Logging.

9.4.3 Stopping Console Logging (SET /COLOG=OFF)

To stop Console Logging, specify the option =OFF with the command SET /COLOG.

Format

```
SET /COLOG=OFF
```

When you stop Console Logging, the current console terminal assignment is retained. Use the DCL command SHOW DEVICES (MCR command DEV) to display the current console terminal. (For a description of how to monitor CO: redirection, see Section 9.5.)

Note

Do not abort the COT... task. Aborting COT... causes severe pool fragmentation.

9.4.4 Enabling or Reassigning CO: (/COTERM)

To enable the most recent console terminal assignment or to reassign the default assignment to a different device, specify the /COTERM option with the command SET /COLOG.

Format

```
SET /COLOG/COTERM[=ttnn:]
```

Parameter

ttnn:

Specifies the newly assigned console terminal.

When you bootstrap your system, the default console terminal assignment is TT0. When you start Console Logging, the default console terminal assignment is the current console output device.

9.4.5 Disabling CO: (/NOCOTERM)

If you cannot dedicate one of your terminals to receiving system messages, disable the console terminal. The Console Logger can maintain a log file of system messages without using a console terminal.

To disable the console terminal, specify the /NOCOTERM option with the SET /COLOG command.

```
>
```

Then, assign CO: to the null device (NL:). Use one of the following commands:

```
MCR>
```

```
DCL>
```

If you do not assign CO: to a null device, the Console Logger displays the message "Device not ready" whenever it receives a message beginning with a CTRL/G sequence (a bell sound).

9.4.6 Enabling or Reassigning the Log File (/LOGFILE)

To enable the log file or to reassign it to a different file specification, specify the /LOGFILE option with the command SET /COLOG.

Format

```
SET /COLOG/LOG[FILE][=[filespec]]
```

Parameter

filespec

Represents any valid file specification. The default file specification is LB:[1,4]CONSOLE.LOG.

For a description of valid file specifications, see the *RSX-11M-PLUS MCR Operations Manual* or the *Micro/RSX User's Guide, Volume 1*.

Using the /LOGFILE option, you can do the following:

- Change only part of the log file specification.
- Create a new version of the log file.
- Reassign the log file back to the default specification.

These changes are described in the following subsections.

9.4.6.1 Changing the Log File Specification

To change only part of the file specification for the log file, type only that part of the file specification after the equal sign (=). When you do not specify the complete file specification, the COT... task uses the default file specification LB:[1,4]CONSOLE.LOG to supply the missing values.

For example, if the system default device LB: is DR0, specify the log file DB0:[301,55]CONSOLE.LOG by entering the following command line:

```
>
```

If you want the log file to be TESTLOG.LOG in the default User Identification Code (UIC) on device LB:, enter the following command line:

```
>
```

9.4.6.2 Creating a New Version of the Log File

When you use the /LOGFILE option to reassign the log file, the COT... task creates a new version of the current log file. However, you can also use /LOGFILE to create a new version without reassigning the file, as follows:

```
>
```

Although a new version of the log file is created, the rest of the file specification does not change.

If you disabled the log file with the /NOLOGFILE option, this command line restores the most recent log file assignment.

9.4.6.3 Reassigning the Log File Back to the Default

To change a log file back to the default, specify the /LOGFILE option without the equal sign, as follows:

```
>
```

The preceding command line creates and opens a new version of the default log file.

9.4.7 Disabling the Log File (/NOLOGFILE)

To prevent the Console Logger from recording system messages in the log file, specify the /NOLOGFILE option with the command SET /COLOG.

Format

```
SET /COLOG/NOLOGFILE
```

After you disable the log file, you still receive system messages (time-stamped by COT...) at the console terminal.

9.4.8 Using Multiple Console Logging Commands

You can enter multiple Console Logging commands in a command line by separating each option with a slash (/).

The following examples show how to use several Console Logging commands in a single command line.

Examples

```
>
```

Starts Console Logging and assigns the console terminal to TT17. This command line also opens a log file named TESTLOG.TST in the default User File Directory (UFD) on the default system device (LB:).

```
>
```

Disables the console terminal and restores the log file to the default.

9.5 Monitoring the Status of the Console Terminal

If you do not specify a Console Logging function, the SET /COLOG command displays the console terminal assignment. For example:

```
>
COT --
Console = TT051:
Logfile = DB000:[1,4]CONSOLE.LOG
```

To determine the status of CO: while the Console Logger is active, you can also use the DCL command SHOW DEVICES (MCR command DEV). If CO: is active, the system responds with a message to indicate that the console driver (CODRV) is loaded and COT... is handling system messages to CO:. For example:

```
DCL>
CO0: LOADED
```

```
MCR>
CO0: LOADED
```

If you disable Console Logging, the system displays the terminal to which you have redirected the CO: pseudo device. For example:

```
DCL>
CO0: TT51:
```

```
MCR>
CO0: TT51:
```

Before COT... exits, it redirects CO: to the current console terminal. If there is no console terminal when you stopped Console Logging (that is, if you disabled it with the /NOCOTERM option), then CO: is redirected to the last terminal COT... assigned as the console terminal. In the previous example, the terminal driver handles messages directed to CO: and redirects them to TT51.

If you disable the console terminal with the /NOCOTERM option but continue to record system messages to CO: in a log file, use the SHOW DEVICES (DEV) command to determine the status of CO:. For example:

```
DCL>
CO0: LOADED
```

```
>
CO0: LOADED
```

In this example, the console driver is loaded. COT... handles all messages directed to CO:; however, system messages are not directed to a terminal. Use the display option SET /COLOG to verify that CO: is not assigned to a terminal.

9.6 Redirecting the Console Terminal

Use the DCL command ASSIGN/REDIRECT (MCR command RED) to redirect CO: to a terminal when the Console Logger is not active. If you try to redirect CO: while the Console Logger is active, ASSIGN/REDIRECT (RED) returns the following error message:

```
 -- Device not redirectable
```

For example, assume that the Console Logger is active and the console terminal is assigned to TT2. When you stop Console Logging, COT... redirects CO: to TT2 before exiting. To verify that CO: is redirected to TT2, use the SHOW DEVICES (DEV) command. Then, to assign the console terminal to TT0, use the ASSIGN/REDIRECT (RED) command, as follows:

```
DCL>
```

```
MCR>
```

When you restart Console Logging, COT... assigns TT0 as the console terminal.

9.7 Reading the Log File

To read the log file, use the DCL command TYPE with the /SHARED qualifier or use the Peripheral Interchange Program (PIP) with the Shared Read (/SR) switch. When you specify the /LOGFILE option, the log file is open for writing; so, to read the file, you must specify that it is shareable. Use one of the following command lines:

```
DCL>
```

```
MCR>
```

If you omit the /SHARED qualifier (/SR switch), you will receive the following error message:

```
xxx -- Open failure on input file
DR0:[1,4]CONSOLE.LOG;1 -- Accessed for write
```

The variable xxx represents the first three letters of the command (that is, TYP or PIP).

For more information on PIP, see the *RSX-11M-PLUS Utilities Manual.* For more information on the TYPE/SHARED command, see the *RSX-11M-PLUS Command Language Manual* or the *Micro/RSX User's Guide, Volume 2.*

Example

```
16:43:19 Logout user [1,4] TT53:
16:43:35 Login  user ALPHA         [7,334] TT53:
16:45:07 Login  user BRAVO         [7,42] TT13:
16:46:21 *** DM0: -- Dismount complete
16:46:32 Login  user DECNET        [240,240] TT30:
16:47:22 Login  user CHARLIE       [7,373] TT22:
16:47:38 *** LP0: -- Not ready
16:47:49 Logout user [1,4] TT53:
16:47:58 Logout user [7,373] TT22:
16:48:01 Login  user ALPHA         [7,334] TT53:
16:49:40 Login  user DECNET        [240,240] HT0:
```

Displays a sample log file.

9.8 Sample Console Logging Sessions

The examples that follow show typical Console Logging terminal sessions. They illustrate how to use the Console Logging commands described in this chapter.

These examples assume that the pseudo device LB: is assigned to DB0.

Examples

```
>
COT --
Console = None
Logfile = None
```

Displays the current console terminal (none) and log file (none) assignments, indicating that the Console Logger is not active.

```
DCL>
COO: TTO:
```

```
MCR>
COO: TTO:
```

Uses the DCL command SHOW DEVICES (MCR command DEV) to show how CO: is redirected. CO: is redirected to TT0, which means that the Console Logger is not active and that the terminal driver redirects CO: to TT0.

```
>
>
COO: LOADED
```

Starts Console Logging and then shows that the Console Logger is active because CODRV is loaded.

```
>
COT --
Console = TT000:
Logfile = DB000:[1,4]CONSOLE.LOG;1
```

Shows that COT... redirects CO: to TT0 and that DB0:[1,4]CONSOLE.LOG;1 is the log file. These are both the defaults when you bootstrap your system.

```
>
>
```

Assigns the console terminal to TT51 and the log file to DR:[301,55]TEST.LOG. The COT... task fills in the missing fields of the file specification with values from the default file specification. In this example, COT... supplies the file type.

```
>
COT --
Console = TT051:
Logfile = DR000:[301,55]TEST.LOG;1
```

Displays the current console terminal (TT51) and log file (DR:[301,55]TEST.LOG;1) assignments resulting from command lines in the preceding example.

```
>
>
COT --
Console = None
Logfile = DR000:[301,55]TEST.LOG;1
```

Disables the console terminal and retains the current log file (DR:[301,55]TEST.LOG;1). The display option confirms this.

```
>
COO: LOADED
```

Shows that when you disable the console terminal with the /NOCOTERM option (see the previous example), the Console Logger is still active (CODRV is loaded), but CO: is not redirected to any terminal.

```
>
>
COT --
Console = TT051:
Logfile = None:
```

Illustrates the multiple commands that restore the most recent console terminal assignment (TT51) and disable the log file. The display option confirms this.

```
>
>
COT --
Console = TT017:
Logfile = DR000:[301,55]TEST.LOG;2
```

Illustrates multiple commands, which assign the console terminal to TT17 and which restore the most recent log file. The display option confirms these assignments. The version number of the log file is incremented.

```
>
>
COT --
Console = None
Logfile = None
>
CO0: TT17:
```

Stops Console Logging. The display option confirms that the Console Logger is not active. CO: is redirected to TT17, which was the last console terminal assignment before COT... exited.

```
MCR>
MCR>
CO0: TT51:
```

```
DCL>
DCL>
CO0: TT51:
```

Redirects CO: to TT51 with the DCL command ASSIGN/REDIRECT (MCR command RED) and confirms this with the DCL command SHOW DEVICES (MCR command DEV).

```
>
>
COT --
Console = TT051:
Logfile = DB001:[7,301]CONSOLE.LOG;1
```

Illustrates multiple commands on a command line, which start Console Logging, retain the current console terminal assignment (TT51) redirected with the ASSIGN/REDIRECT (RED) command in the previous example, and open a log file DB001:[7,301]CONSOLE.LOG;1. The display option confirms these assignments.

```
>
>
COT --
Console = TT000:
Logfile = DB000:[1,4]CONSOLE.LOG;2
```

Illustrates multiple commands on a command line, which assign the console terminal to TT0 and restore the default log file. The display option confirms these assignments. The version number of the log file is incremented.

```
DCL>
TYP -- Open failure on input file
DB0:[1,4]CONSOLE.LOG;3 -- Accessed for write

MCR>
PIP -- Open failure on input file
DB0:[1,4]CONSOLE.LOG;3 -- Accessed for write
```

Illustrates what happens when you try to read an open log file without the DCL TYPE command qualifier /SHARED (the PIP Shared Read switch /SR).

```
DCL>
16:43:19  Logout user [1,4] TT53:
16:43:35  Login  user ALPHA         [7,334] TT53:
16:45:07  Login  user BRAVO         [7,42] TT13:
16:46:21  *** DM0: -- Dismount complete
16:46:32  Login  user DECNET        [240,240] TT30:
16:47:22  Login  user CHARLIE       [7,373] TT22:
16:47:38  *** LP0: -- Not ready
16:47:49  Logout user [1,4] TT53:
16:47:58  Logout user [7,373] TT22:
16:48:01  Login  user ALPHA         [7,334] TT53:
16:49:40  Login user DECNET         [240,240] HT0:

MCR>
16:43:19  Logout user [1,4] TT53:
16:43:35  Login  user ALPHA         [7,334] TT53:
16:45:07  Login  user BRAVO         [7,42] TT13:
16:46:21  *** DM0: -- dismount complete
16:46:32  Login  user DECNET        [240,240] TT30:
16:47:22  Login  user CHARLIE       [7,373] TT22:
16:47:38  *** LP0: -- not ready
16:47:49  Logout user [1,4] TT53:
16:47:58  Logout user [7,373] TT22:
16:48:01  Login  user ALPHA         [7,334] TT53:
16:49:40  Login user DECNET         [240,240] HT0:
```

Shows how to read an open log file by using the /SHARED qualifier (PIP /SW switch). Each message in the log file begins with time-stamping.

```
>
COT -- Not logging on a terminal or to a file, COT exiting
>
CO0: TT0:
```

Disables the console terminal and the log file. COT... exits because no Console Logging is in progress. COT... displays a message informing you that it is exiting. The MCR command DEV shows that CO is redirected to TT0, which was the console terminal assignment before COT... exited.

9.9 Console Logging Error Messages

COT... generates the following error messages:

COT—LUN assignment to console terminal failed

Explanation: You specified an incorrect terminal number as the console terminal.

User Action: Use the /COTERM option to assign the console terminal to a terminal in your hardware configuration.

COT—Illegal console terminal

Explanation: You tried to assign a device that is off line or that is not a terminal to CO:. The device may be illegal because it is a spooled terminal.

User Action: Use the /COTERM option to assign a terminal in your hardware configuration to CO:.

COT—Console driver not loaded, COT exiting

Explanation: The console driver, CODRV, is not loaded.

User Action: Load the console driver and start Console Logging.

COT—Maximum consecutive write errors, disabling logfile

Explanation: An error occurred on five consecutive write operations to the log file. COT... closes the log file and disables logging to the log file.

User Action: Determine the problem that caused the write errors and correct it. Then, restart the COT... task.

COT—Not logging on a terminal or to a file, COT exiting

Explanation: You have disabled the console terminal with the /NOCOTERM option and you have disabled the log file with the /NOLOGFILE option.

User Action: If you did not intend to stop Console Logging, start it again by specifying either a log file, console terminal, or both.

COT—Unable to redirect CO: to a terminal-CO: is offline, COT exiting

Explanation: CO is off line. There is a problem with the most recent console terminal assignment. It may be off line, a spooled terminal, or not a terminal.

User Action: Start COT... and specify a valid console terminal with the /COTERM option. Then, stop Console Logging.

COT—Syntax error

Explanation: You made an error in entering a Console Logging command.

User Action: Check the command description in Section 9.4 and enter the command correctly.

COT—Will attempt console LUN assignment to TTnn:

Explanation: This error message may occur when you stop Console Logging. If COT... cannot redirect the console terminal to the terminal currently assigned as the console terminal, it tries to redirect CO to the previous console terminal assignment. If that fails, it tries TT0. If that fails, it generates the following error message:

```
COT -- LUN assignment to console terminal failed
```

User Action: Start Console Logging again by specifying a valid console terminal. Then, stop Console Logging.

COT—Illegal command—Console logging has not been turned on

Explanation: You issued a Console Logging command before starting Console Logging.

User Action: Start Console Logging and then reenter the command.

COT—Logfile x error
filespec—QIO Error Code -nn.

Explanation: An I/O error occurred while COT... was performing an operation on the log file. The function being performed, x, is WRITE, OPEN, or CLOSE. The filespec is any valid file specification designated for the log file. The error code, -nn., is an I/O error code.

User Action: Find the I/O Error Code in the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual* and take the appropriate action.

Chapter 10
Resource Accounting

10.1 Introduction to Resource Accounting

Resource Accounting provides a transaction file of system usage information. The data collected in the transaction file pertains to your computing environment. How you interpret and analyze this data depends on your objectives. For example, are you interested in billing or system measurement? To tailor the accounting information and format to your application, you can write a report program (Section 10.4). This program accesses the transaction file, reads the required data fields, and writes a report for you.

For conventional RSX-11M-PLUS operating systems (that is, excluding pregenerated systems), Resource Accounting is a system generation option. For pregenerated RSX-11M-PLUS (RL02 distribution kits) and Micro/RSX operating systems, Resource Accounting is included by default during the system startup procedure.

If you are using a conventional RSX-11M-PLUS system, you can also write a privileged program to define and to create your own transactions. By using specific entry points in the Executive Accounting module, you can write these transactions into the transaction file (see Section 10.5).

Table 10-1 lists some of the system usage information gathered by Resource Accounting and saved in the transaction file.

Table 10-1: Summary of Transaction File Information

Type of Data	Transaction File Information
User	Session identification, terminal number, and account number
	Billing stop date and time
	Central processing unit (CPU) usage time
	Tasks active at time of logout or system failure
	Number of tasks run
	Number of directives and QIOs issued
	Status at logout or system failure
Task	Session identification, terminal, and account number
	Disk overlay loads
	Number of times checkpointed
	Number of times context has been loaded
	Highest running priority
	Task begin time and end time
	CPU usage time
	Number of directives and QIOs issued
	Status at logout or system failure
System	Accounting start date and time
	Accounting stop date and time (0 if system fails)
	Shutdown code
	Transaction file ID, sequence number, and device
	Last scan date, time, and scan rate in seconds
	Total CPU time and total CPU zero intervals
	Total number of tasks executed
	Total number of logins
	Number of current users
	Number of checkpoints
	Number of Shuffler (SHF) runs
	Number of directives issued
	Number of QIOs issued

Table 10-1 (Cont.): Summary of Transaction File Information

Type of Data	Transaction File Information
Login	User name, ID, terminal, and account number
	Login User Identification Code (UIC), date, and time
	Illegal password (Invalid login only)
	Session ID, terminal, and account number
Device allocation	Allocation or deallocation date, time, and device
Device deallocation	
Device mount	Session ID, terminal, and account number
Device dismount	Mount date, time, and device
	Volume label for Files-11 mounted device
	Type of mount (foreign, shared, and so on)
	Owner UIC
	Volume protection code
	Name of Ancillary Control Processor (ACP) for device
Print Job Data	Session ID, terminal, and account number
	Print job date and time
	Job name and page count
	Number of files printed, printer device, and forms number
	Job priority
Card Reader Job Data	Session ID, terminal, and account number
	Batch or print job name
	Number of cards read
	Reader device name and number
System Time Change	Old Time
	New Time
Device Usage Data	I/O counts
	Error counts
	Tuning information

10.2 Controlling Resource Accounting

The following Resource Accounting commands start, stop, and modify the accounting subsystem:

Command	Function
START/ACCOUNTING	Starts up the accounting subsystem.
STOP/ACCOUNTING	Shuts down the accounting subsystem.
SET ACCOUNTING	Changes parameters in an accounting subsystem that has already been started.

All the commands for controlling Resource Accounting are privileged.

10.2.1 START/ACCOUNTING Command

The START/ACCOUNTING command starts up the accounting subsystem. The SYSLOG and ...ACC tasks must be installed before you issue the START/ACCOUNTING command. If you do not supply the optional parameters, the system uses the default values.

Format

```
START/ACCOUNTING [parameter_1] . . . [ parameter_n]
```

Parameters

CRASH_REASON:yes/no
EXTEND_SIZE:value
FILE:filespec
POOL_RESERVE:value
SCAN_RATE:value
STATISTICS_SCAN[:rate]
SYSTEM_STATISTICS:yes/no
TASK:yes/no

Parameter Descriptions

CRASH_REASON:yes/no

Instructs Resource Accounting to display the prompt Reason for crash: when you restart accounting after a system failure.

If you enter CRASH_REASON:YES and a system failure occurs while scanning is active, Resource Accounting prompts you to enter a reason for the crash when you restart accounting. The reason can be as many as 60 characters in length and is stored in the Crash Recovery Transaction Block.

If you enter CRASH_REASON:NO, a prompt does not appear. This is the default.

EXTEND_SIZE:value

Specifies the size of the initial allocation for the transaction file and for each extension when it is needed. The value can be any amount up to 65,535$_8$ blocks. If there is low accounting activity, a small extension size is adequate because few extensions are required and overhead is low. If accounting activity is high, a larger extension size lowers the overhead if disk space is available for the larger extension.

The default value is 10$_{10}$ blocks.

FILE:filespec

Specifies the name of the transaction file that Resource Accounting creates to store the data. The default is LB:[1,6]ACNTRN.SYS.

POOL_RESERVE:value

Specifies the number of blocks (in decimal) of secondary pool that Resource Accounting must leave free after each allocation. If Resource Accounting cannot leave this amount free, the allocation will fail and prevent the swamping of secondary pool.

The default is a quarter of the total secondary pool size. Use of the default is recommended.

SCAN_RATE:value

Specifies the time interval between the periods when Resource Accounting writes the SAB and active User Account Blocks (UABs) to the scan file called LB:[1,6]SYSSCAN.TMP. Writing the SAB and UABs to the scan file protects against loss in the event of a system "crash."

Specify the value as a decimal number n, followed by an M, for minutes, or an S, for seconds. If you specify 0, no scanning is done; therefore, if the system fails ("crashes"), the active UABs and the SAB are not recorded in the transaction file and the data is lost.

The default is SCAN_RATE:5M.

If Resource Accounting finds a scan file when you start, it outputs a crash transaction and copies the data in the scan file to the new transaction file. A copy of this crash transaction, followed by the transaction file, is shown as the output of the SHOW ACCOUNTING /TRANSACTION_FILE command in Section 10.3.2. Resource Accounting copies all the old data before it writes the startup transaction and before it processes any new data.

When you stop Resource Accounting, the scan file is deleted.

STATISTICS_SCAN[:rate]

Specifies a scan rate for the collection of device statistics. These device statistics measure seek-optimization parameters and provide information on disk activity and throughput. The device statistics are collected either once or periodically.

To indicate a one-time-only collection, do not specify a rate. To collect statistics periodically, specify a value for the rate in minutes (nM) or seconds (nS).

If you specify this parameter, you must also specify the SYSTEM_STATISTICS parameter (unless you are making a one-time request). If you specify SYSTEM_STATISTICS:YES, the default value for STATISTICS_SCAN is 1 minute. If you specify SYSTEM_STATISTICS:NO or STATISTICS_SCAN:0, the system does not accumulate device statistics.

SYSTEM_STATISTICS:yes/no

Specifies to Resource Accounting whether or not to accumulate systemwide statistics. Systemwide statistics are the contents of the accumulation fields in the System Account Block (SAB). These fields are as follows:

Accumulation Field	System Statistic
B.CPU	Total CPU time used
B.DIR	Total directive count
B.QIO	Total QIO$ count
B.TAS	Total task count

The default is SYSTEM_STATISTICS:YES.

TASK:yes/no

Requests data on individual tasks running on the system. You can run task accounting when the system is running at peak load times and the task accounting data can be analyzed on a per task basis to identify heavy resource users.

If you enter TASK:YES, task accounting is initiated. If you enter TASK:NO, task accounting is stopped. The default is TASK:NO.

Note

> Task accounting requires substantial disk space because the transaction file can become quite large; therefore, request task accounting only when you require the task accounting data.

Example

The type of device statistics collected by Resource Accounting is shown in the following output from a transaction file:

```
Device Statistics
     Time of Device Statistics = 15-JUN-87 16:27:23    Device = DR1:
     IO Count = 0.              Words Transferred Count = 0.
     Soft Error Limit = 8.             Soft Error Count = 0.
     Hard Error Limit = 5.             Hard Error Count = 0.
     Cylinder Crossed Count = 0.       Current Fairness Count = 0.
     Fairness Count Limit = 10.

Device Statistics
     Time of Device Statistics = 15-JUN-87 16:27:23    Device = DR2:
     IO Count = 13832.         Words Transferred Count = 9002804.
     Soft Error Limit = 8.             Soft Error Count = 0.
     Hard Error Limit = 5.             Hard Error Count = 0.
     Cylinder Crossed Count = 1053     Current Fairness Count = 0.
     Fairness Count Limit = 10.

Device Statistics
     Time of Device Statistics = 15-JUN-87 16:27:24    Device = DR3:
     IO Count = 68490.         Words Transferred Count = 80640306.
     Soft Error Limit = 8.             Soft Error Count = 0.
     Hard Error Limit = 5.             Hard Error Count = 0.
     Cylinder Crossed Count = 313.     Current Fairness Count = 7.
     Fairness Count Limit = 10.
```

This example lists the transaction file device statistics for DR1, DR2, and DR3. Definitions for terms appearing in the example are as follows:

I/O count — Displays the number of I/O requests to the device.

Words transferred count — Displays the amount of data transferred in decimal words.

Soft error — Indicates a recoverable error (see Chapter 11). The I/O operation completes successfully.

Hard error — Indicates a nonrecoverable error (see Chapter 11). The I/O operation does not complete.

Also, see the *RSX-11M-PLUS and Micro/RSX Error Logging Manual* for more information about hard and soft errors.

Cylinder crossed count

Displays the number of *cylinders* crossed while accessing data. *Cylinders* are intervals of disk space. This data can be analyzed for I/O queue optimization purposes (see Chapter 14).

Fairness count

Indicates a I/O queue optimization parameter that indicates how many times an I/O request was passed over (see Chapter 14).

10.2.2 SET ACCOUNTING Command

The SET ACCOUNTING command allows you to change the value of parameters specified when Resource Accounting was started with the START/ACCOUNTING command or previously modified with a SET command. (A transaction is written containing the old and new values.)

The following options can be changed while Resource Accounting is running:

- The file that transactions are written to
- The size of the extensions
- The scan rate for the scan file
- The scan rate for collecting device statistics
- The state of task accounting (on or off)

Format

```
SET ACCOUNTING [[/]parameter_1] . . . [ [/]parameter_n]
```

Parameters

EXTEND_SIZE:value
FILE[:filespec]
SCAN_RATE:value
STATISTICS_SCAN:value
TASK:yes/no

Parameter Descriptions

EXTEND_SIZE:value

Changes the size for each extension of the transaction file specified with the START/ACCOUNTING EXTEND_SIZE parameter.

FILE[:filespec]

Changes the file to which transactions are written. The default is the current file specification.

SCAN_RATE:value

Changes the scan rate for the scan file specified with the FILE parameter. To disable this parameter, specify SCAN_RATE:0.

STATISTICS_SCAN:value

Changes the scan rate for collecting device statistics. To disable this parameter, specify STATISTICS_SCAN:0.

TASK:yes/no

Requests data on individual tasks running on the system. Task accounting creates Task Account Blocks (TABs) for all active tasks in the system. All data for each task is gathered starting at this time.

If you enter TASK:NO, task accounting is stopped. Task accounting on a per task basis is terminated and the data for all active tasks is written to the transaction file. However, tasks that have a CPU time limit specified require task accounting data to check the time limit. Therefore, the data for these tasks is written to the transaction file when the task exits.

10.2.3 STOP/ACCOUNTING Command

The STOP/ACCOUNTING command shuts down the Resource Accounting subsystem. The parameter is provided to indicate why the shutdown is occurring and must be entered with the command. Resource Accounting enters the reason for the shutdown in the System Account Block (SAB).

Formats

```
STOP/ACCOUNTING reason

STOP/ACCOUNTING CLEAN_UP
```

Reasons for Shutdown

MAINTENANCE

Stops Resource Accounting to perform system maintenance.

REBOOT

Stops Resource Accounting to rebootstrap the system.

SCHEDULED_SHUTDOWN

Stops Resource Accounting because the system is being shut down or used for other purposes and accounting is not needed.

SHUTUP

Stops Resource Accounting to shut down the system. This is the default reason for the SHUTUP program (see Chapter 6).

OTHER

Specifies a reason if none of the other reasons are adequate.

CLEAN_UP

Stops Resource Accounting in an organized manner; that is, it "cleans up" all the data structures to allow accounting to restart successfully.

Specify STOP/ACCOUNTING CLEAN_UP if the SYSLOG task aborts or if accounting encounters a fatal error. Accounting accepts this command only after Resource Accounting fails or aborts.

When you specify CLEAN_UP, the system assumes that all accounting data in memory is invalid. It deallocates the data back to secondary pool without writing it to the transaction file. *If the Resource Accounting data structures are corrupted, the system may fail while it attempts to deallocate secondary pool.*

You can restart accounting normally after you issue this command.

10.3 Accessing Information in the Transaction File

To access the system usage information contained in the transaction file, use the SHOW ACCOUNTING commands. They are as follows:

Command	Function
SHOW ACCOUNTING/TRANSACTION_FILE	Converts the transaction file to a readable format. You can display the formatted transaction file on a terminal or store it for later use.
SHOW ACCOUNTING/INFORMATION	Displays accounting information about a specific terminal.
SHOW ACCOUNTING/DATATRIEVE	Converts a transaction file to a file that is readable by DATATRIEVE-11 (see Section 10.4).

The subsections that follow describe the SHOW ACCOUNTING/INFORMATION and SHOW ACCOUNTING/TRANSACTION_FILE commands. The SHOW ACCOUNTING/DATATRIEVE command is described in Section 10.4.

SHOW ACCOUNTING/INFORMATION

10.3.1 SHOW ACCOUNTING/INFORMATION Command

This command displays, on the issuing terminal (TI:), the accounting information for the specified terminal. Nonprivileged users can receive their own accounting data. A privileged user can obtain any accounting data.

Format

```
SHOW ACCOUNTING/INFORMATION [parameter]
```

Parameters

ttnn:

Displays accounting data for the terminal ttnn. A privileged user has access to the accounting data of any terminal on the system. Nonprivileged users can access information about their own account.

If you do not specify a terminal, the accounting data of the terminal issuing the command is displayed.

CO

Displays the accounting data of the system tasks that are currently running.

SYS

Displays the current system totals at your terminal.

TASK=taskname

Shows accounting data for the specified task if the system is maintaining task accounting. (See the description of the TASK parameter for the START/ACCOUNTING and SET ACCOUNTING commands.)

10.3.2 SHOW ACCOUNTING/TRANSACTION_FILE Command

The SHOW ACCOUNTING/TRANSACTION_FILE command converts the transaction file into a readable format. It either displays this file on a terminal or writes it into another file. The SHOW ACCOUNTING/TRANSACTION_FILE command uses the current transaction file as the default file. (When Resource Accounting is not running, there is no default file.)

Format

```
SHOW ACCOUNTING/TRANSACTION_FILE[:infilespec] outfilespec
```

Parameters

infilespec

Specifies the transaction file to be read. If you do not specify an input file, the current transaction file is used.

outfilespec

Specifies either a terminal (specified as ttnn or TI:) or a valid file specification. The default file is the transaction file named in the START/ACCOUNTING command.

Notes

1. The *record size* in the transaction file is equal to the *transaction length*. (The only exception to this is the Resource Accounting startup transaction that only occurs once and is 512_{10} bytes long. The startup transaction consists of a file header only, but its size is increased to 512_{10} bytes to force a file extension.)
2. The directory [126,10] on the RSX-11M-PLUS distribution kit includes the source code for the SHOW ACCOUNTING/TRANSACTION_FILE command. The code serves as an example of how to read the transaction file.
3. For a detailed description of the unformatted version of the transaction file, see Section 10.6 or the file [126,10]ACNDF.DOC on the RSX-11M-PLUS distribution kit.

Example

The SHOW ACCOUNTING/TRANSACTION_FILE command produces a formatted display of the transaction file. The following example shows the format of a typical transaction file:

```
ACCOUNTING DATA - CURRENT FILE          11-JUN-87 16:56:04

SYSTEM CRASH RECORD
TIME OF LAST SCAN = 7-JUN-87 10:48:17   SCAN RATE (SEC) = 300. ❶
RESTART TIME = 7-JUN-87 19:58:15
REASON = ❷
    .
    .
    .

USER - ALPHA     P ❸
      SESSION ID = F1143        TI: = TT40:    ACCOUNT = 1.
      LOGIN UIC = [7,111]       LOGGED ON = 7-JUN-87 08:58:53
      LOGGED OFF = 00-000-00 00:00:00    BILLING STOPPED = 7-JUN-87 10:48:17
      CPU = 4461.          TASKS ACTIVE = 1.       TASKS RUN = 95.
      QIOS = 6215.                       DIRECTIVES = 10240.
```

```
STATUS     ACT   CRH
   .
   .
   .

USER - SYSTEM TASKS
      SESSION ID = $SY0              TI: = CO:      ACCOUNT = 0.
      LOGON UIC = [0,0]      LOGGED ON = 6-JUN-87 15:48:16
      LOGGED OFF = 00-00-00 00:00:00  BILLING STOPPED = 7-JUN-87 10:48:17
      CPU = 103866.       TASKS ACTIVE = 9.       TASKS RUN = 601.
      QIOS = 128967.             DIRECTIVES = 667250.
 STATUS    ACT   CRH

TOTAL SYSTEM STATISTICS ❹
      ACCOUNTING STARTED = 6-JUN-87 15:48:16
      ACCOUNTING STOPPED = 00-000-00 00:00:00    SHUTDOWN CODE = 0
      TRANS FILE ID = 1422       SEQ NUM = 1          DEVICE = DB1:
      TIME OF LAST SCAN = 7-JUN-87 10:48:17    SCAN RATE (SEC) = 300.
      TOTAL CPU = 441264.       ZERO CPU INTERVALS = 374665.
      TOTAL TASKS = 3236.         TOTAL LOGONS = 60.
      CURRENT USERS = 27.       CHECKPOINTS = 23.
      SHF RUNS = 0.            DIRECTIVES = 2373564.      QIOS = 556015.

ACCOUNTING STARTUP
      ACCOUNTING STARTED = 7-JUN-87 19:58:16

LOGIN - BETA          B
      SESSION ID = RXP1        TI: = TT26:    ACCOUNT = 0.
      LOGON UIC = [7,102]      TIME = 7-JUN-87 19:58:35

LOGIN - ALPHA         P
      SESSION ID = F112        TI: = TT55:    ACCOUNT = 1.
      LOGON UIC = [7,111]      TIME = 7-JUN-87 20:05:44

DEVICE MOUNT
      SESSION ID = F112       TI: = TT55:    ACCOUNT = 1.
      TIME = 7-JUN-87 20:06:37    DEVICE = DR1:     VOL ID = WELCOMEBACK
      OWNER UIC = [0,0]       ACP NAME =F11ACP      VOL PROT MASK = 0

USER - ALPHA          P
      SESSION ID = F112        TI: = TT55:    ACCOUNT = 1.
      LOGON UIC = [7,111]      LOGGED ON = 7-JUN-87 20:05:44
      LOGGED OFF = 7-JUN-87 20:06:43    BILLING STOPPED = 7-JUN-87 20:06:43
      CPU = 141.          TASKS ACTIVE = 0.        TASKS RUN = 17.
      DIRECTIVES = 472.   QIOS = 166.

LOGIN - OMEGA         D
      SESSION ID = RXP3        TI: = TT37:    ACCOUNT = 4.
      LOGON UIC = [7,71]       TIME = 7-JUN-87 20:08:11

DEVICE ALLOCATION
      SESSION ID = RXP4        TI: = TT37:    ACCOUNT = 4.
      TIME = 7-JUN-87 20:09:12     DEVICE = DL0:

LOGIN - THETA         D
      SESSION ID = SPR5        TI: = TT55:    ACCOUNT = 0.
      LOGON UIC = [7,327]      TIME = 7-JUN-87 20:09:14
```

```
DEVICE MOUNT
      SESSION ID = RXP4        TI: = TT37:    ACCOUNT = 4.
      TIME = 7-JUN-87 20:09:15     DEVICE = DLO:     VOL ID =
      OWNER UIC = [7,107]      ACP NAME = TSTACP    VOL PROT MASK = 0

DEVICE DEALLOCATION
      SESSION ID = RXP4        TI: = TT37:    ACCOUNT = 4.
      TIME = 7-JUN-87 20:09:15     DEVICE = DLO:

USER - SYSTEM TASKS
      SESSION ID = $SYO        TI: = CO:      ACCOUNT = 0.
      LOGON UIC = [0,0]        LOGGED ON = 7-JUN-87 19:58:16
      LOGGED OFF = 00-000-00 00:00:00     BILLING STOPPED = 7-JUN-87 20:13:21
      CPU = 1603.         TASKS ACTIVE = 4.        TASKS RUN = 64.
      DIRECTIVES = 2986.

TOTAL SYSTEM STATISTICS
      ACCOUNTING STARTED = 7-JUN-87 19:58:16
      ACCOUNTING STOPPED = 7-JUN-87 20:13:21     SHUTDOWN CODE = 4
      TRANS FILE ID = 205         SEQ NUM = 20        DEVICE = DRO:
      STATISTICAL SCAN RATE (SEC.) = 3600.     FILE EXT. SIZE = 10.
      TIME OF LAST SCAN = 7-JUN-87 20:13:17     SCAN RATE (SEC) = 300.
      TOTAL CPU = 4260.         ZERO CPU INTERVALS = 3063.
      TOTAL TASKS = 243.          TOTAL LOGONS = 7.
      CURRENT USERS = 1.        CHECKPOINTS = 0.
      SHF RUNS = 0.            DIRECTIVES = 10241.       QIOS = 4946.
```

❶ To minimize loss of data, Resource Accounting writes the System Account Blocks (SABs) and currently active User Account Blocks (UABs) to a temporary file every n seconds or minutes (unless you specify SCAN_RATE:0 in the START/ACCOUNTING or SET ACCOUNTING command line).

The process of writing this file is called a *scan*. When you restart Resource Accounting, it copies any existing temporary file of accounting data into the accounting file. If the scan rate is 0, the SABs are memory resident until you stop Resource Accounting or until the system fails.

The currently active UABs are memory resident until a user logs out. Therefore, if a system failure ("crash") occurs with the scan rate at 0, the SAB and current UAB data is lost.

❷ The crash reason appears here if Resource Accounting prompts you for a reason for the system failure. The prompt appears only if you specify the CRASH_REASON:YES parameter in the START/ACCOUNTING command line.

❸ The User Account Block (P. Alpha) shows a logged out date and time of 0 because the system crashed. There may or may not be a status line as the last line of the UAB. The *status line* indicates the bit setting of the status mask byte (B.STM) in the UAB. Three of the bits cause this line to appear if they are on. These bits are as follows:

Status Bit	Meaning
ACT (BS.ACT)	Indicates that the UAB was written to the file while still active (all activity had not ceased). This can happen when UABs are copied from the scan file or when all the data structures are written out because accounting has stopped.
LGO (BS.LGO)	Indicates that the user has logged out with tasks remaining active or with clock queue requests pending.
CRH (BS.CRH)	Indicates that this UAB came from the scan file and was copied into the transaction file when Accounting was started after a system crash.

❹ The TOTAL SYSTEM STATISTICS block contains the shutdown code as a digit in the range 0 to 5. These digits have the following meanings:

Digit	Meaning
0	Accounting not stopped due to crash
1	Maintenance
2	Rebootstrap
3	Scheduled shutdown
4	Shutup
5	Other

The ZERO CPU INTERVALS field shows the possible error in the central processing unit (CPU) time measurements. The error occurs because the system clock is usually run at 60 Hz and is slower than the context switch time of the Executive. Therefore, when the time needed to execute a task is less than the resolution of the clock, the execution time is recorded as *zero ticks*. Resource Accounting rounds each zero tick to one-half tick of CPU time and counts them. The ZERO CPU INTERVALS field contains the number of zero ticks that have occurred.

10.4 Using DATATRIEVE–11 to Write an Accounting Report Program

One way to write a specialized accounting report is to use DATATRIEVE–11. By using DATATRIEVE–11, you can customize the content and format of the transaction file generated by your system.

10.4.1 Converting the Transaction File to a DATATRIEVE–11 Readable File

Before a DATATRIEVE–11 program can access the transaction file, the file must be converted to a DATATRIEVE–11 readable file. To convert the file, use the Resource Accounting SHOW ACCOUNTING/DATATRIEVE command.

Format

```
SHOW ACCOUNTING/DATATRIEVE[:trnsfilespec] outfilespec
```

Parameters

trnsfilespec

Specifies the name of an existing Resource Accounting transaction file.

outfilespec

Specifies the name of the output file where the converted transaction file is to reside.

10.4.2 Using a DATATRIEVE–11 Report Generator

RSX–11M–PLUS operating systems (excluding pregenerated systems) are supplied with a sample DATATRIEVE–11 report generator. To produce a formatted accounting report on an RSX–11M–PLUS system, use the sample report generator, as follows:

1. Use the SHOW ACCOUNTING command (see Section 10.4.1) to convert the appropriate transaction file to a file that is readable by DATATRIEVE–11. Name the file ACCOUNT.DAT (for this example, ACCOUNT.DAT is the name used in the file ACNTRN.CMD).

2. Copy the following files from the system disk to your directory:

   [1,2]QUERY.DIC
   [126,24]ACNTRN.CMD

3. After the files are copied, use the following command line to generate a report:

   ```
   >:
   ```

   This command line calls DATATRIEVE–11 and executes the DATATRIEVE–11 commands in the ACNTRN.CMD file. A brief description is then displayed on your terminal screen with a portion of the DATATRIEVE–11 commands that make up the report generator.

4. After the description, DATATRIEVE–11 displays the following prompt:

   ```
   Enter DEVICE OR FILE:
   ```

   Enter a device or file specification that Resource Accounting can use for recording information. For example, specify your terminal as the output device, as follows:

   ```
   Enter DEVICE OR FILE:
   ```

To send the information to a file named ACT.REP, enter the following response:

```
Enter DEVICE OR FILE:
```

While creating the report, you are prompted to respond with a destination for the report displays. These prompts appear at points in the report-generating process where certain parts of the report can be either displayed or directed to a file.

For example, the first prompt you receive is at a point where the STARTUP TRANSACTION part of the report can be displayed. The next prompt appears when the first part of the USER ACCOUNT BLOCK TRANSACTION is complete. At each prompt, you can direct that part of the report to the same report file, to a different report file, or to a device.

To stop the Resource Accounting report process before it is finished, abort DATATRIEVE–11. Then, recopy the file [1,2]QUERY.DIC back into your directory before using the example Resource Accounting report program.

On RSX–11M–PLUS operating systems (excluding pregenerated systems), an example of a simple report generator written in DATATRIEVE–11 is shown in LB:[126,24]ACNTRN.CMD.

10.5 Creating Your Own Transaction Interface

You can define and create your own transactions and write them into the transaction file by using specific entry points in the Executive Accounting module. To define a transaction interface, do the following:

- Write a privileged task.
- Include code in the task to perform the following operations:
  - Store an address for a defined area or *packet* within the task in register 0 (R0).
  - In the packet, supply a value for the fields B.LEN and B.TYP (see Section 10.5.2). The packet will contain the header and the information that is to be written as a transaction.
- Include a call to the Executive directive Switch State ($SWSTK) in the task code.
- From the system state, issue a call to one of the following entry points in the Executive module ACSUB:

  $QTRAN
  $QTRN1
  $QTRN2

The action taken by the ACSUB code depends on the entry point called by the privileged task.

10.5.1 Calling $QTRAN

The $QTRAN routine automatically fills in the user-ID fields of the packet from those used by the TI: terminal of the calling task. In effect, this shows that the new transaction is coming from the same user who ran the task.

To call this entry point, the task must do the following:

- Store the address of the transaction packet in register R0.
- Record the transaction length in the field B.LEN.
- Record a transaction code (a value between 128 and 255) in the field B.TYP.
- Store the values for B.LEN and B.TYP in the transaction packet.

$QTRAN obtains the transaction header from the User Account Block (UAB) of the privileged task. It copies the header (except for B.LEN and B.TYP) into the packet, time-stamps the packet, and links the packet into a queue. The SYSLOG task takes the packet from the queue and writes it into the transaction file.

$QTRAN does not alter the task's space; instead, it copies the packet into secondary pool space before copying the header and queuing the packet into the SYSLOG queue.

Example 10–1 is an example of the code needed to call $QTRAN.

Example 10–1: Call to the $QTRAN Entry Point

```
       .MCALL ACNDF$
       ACNDF$            ;DEFINE ACCOUNTING SYMBOLS

TRAN:  .BLKB      B.MLEN
                   .
                   .
                   .
       This area contains code that
       describes the data to be
       written into the transaction
       file.
                   .
                   .
                   .
       MOV     #TRAN,RO               ;GET PACKET ADDRESS
       MOVB    #BT.MOV,B.TYP(RO)      ;FILL IN TYPE
       MOVB    #B.MLEN,B.LEN(RO)      ;FILL IN LENGTH
       CALL    $SWSTK, .....          ;SWITCH TO SYSTEM STATE
       CALL    $QTRAN                 ;TO CALL QUEUE TRANSACTION EXEC ROUTINE
```

10.5.2 Calling $QTRN1

The $QTRN1 routine allows the user-ID fields to be obtained from a specified place, instead of assuming the current task.

To call this entry point, the task must do the following:

- Store the address of the transaction packet in register R0.
- Record the transaction length in the field B.LEN.
- Record a transaction code (a value between 128 and 255) in the field B.TYP.

- Store the values for B.LEN and B.TYP in the transaction packet.
- Store an Active Page Register (APR) value in register 1 (R1) that points to a User Account Block (UAB) or Task Account Block (TAB). These account blocks, UAB and TAB, contain the user-ID data to be copied into the transaction.

The privileged task writes a transaction associated with another task. The $QTRN1 routine time-stamps the packet and obtains the transaction header from the UAB of the other task. The routine then places the packet in the SYSLOG queue to be written into the transaction file.

The $QTRN1 routine does not alter the packet within the task; instead, it copies identifying information from the specified UAB into secondary pool. Then, it copies data into the packet and places the packet in the SYSLOG queue.

Example 10–2 is an example of the code needed to call $QTRN1.

Example 10–2: Call to the $QTRN1 Entry Point

```
      .MCALL ACNDF$
      ACNDF$                        ;DEFINE ACCOUNTING SYMBOLS
                 .
                 .
                 .
      This example assumes that R1 contains the pointer
      to the TCB of the task for which the transaction
      is to be posted.
                 .
                 .
                 .
TRAN: .BLKB B.MLEN                  ;START OF TRANSACTION
                 .
                 .
                 .
      MOV         T.ACN(R1),R1      ;GET OTHER TASK'S UAB OR TAB ADDRESS
      BEQ         10$               ;IF EQ NO ACCOUNTING DATA PRESENT
      MOV         #TRAN,RO          ;POINT TO TRANSACTION HERE
                 .
                 .
                 .
      MOVB        #BT.MOV,B.TYP(RO) ;FILL IN TRANSACTION TYPE
      MOVB        #B.MLEN,B.LEN(RO) ;FILL IN TRANSACTION LENGTH
      CALL        $SWSTK, .....     ;SWITCH TO SYSTEM STATE
                                    ;TO CALL EXECUTIVE ROUTINES
      CALL        $QTRN1
```

10.5.3 Calling $QTRN2

The $QTRN2 routine assumes the calling task completely fills in the user-ID area of the transaction. The Executive queues the packet with no modification, except for time-stamping.

To call this entry point, the task must do the following:

- Store the address of the transaction packet in register R0.
- Record the transaction length in the field B.LEN.
- Record a transaction code (a value between 128 and 255) in the field B.TYP.

- Store the values for B.LEN and B.TYP in the transaction packet.
- Supply $QTRAN with the following data:
  - Header and data to be written to the transaction file
  - User session ID
  - Account number
  - Terminal ID

The $QTRN2 routine does not obtain a header from any source other than the transaction packet. Therefore, except for the transaction time, the privileged task must have completely filled in the transaction header and the transaction data contained within the packet.

The $QTRN2 routine fills in the transaction time field in the transaction header area of the packet created by the task. Then, it queues the complete packet to the SYSLOG queue to be written to the transaction file.

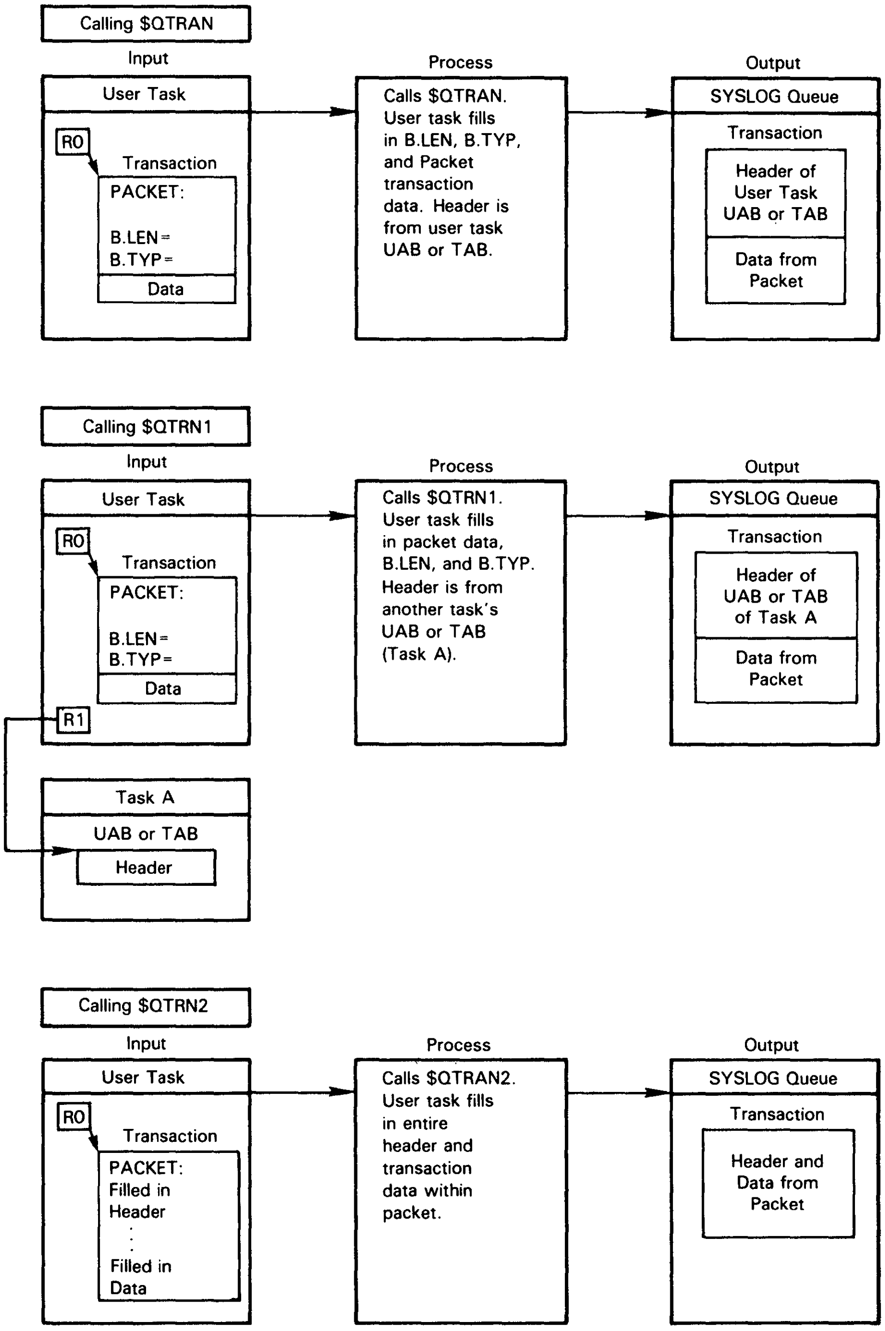
Example 10–3 is an example of the code needed to call $QTRN2.

Example 10–3: Call to the $QTRN2 Entry Point

```
      .MCALL ACNDF$
      ACNDF$                       ;DEFINE ACCOUNTING SYMBOLS
                .
                .
TRAN: .BLKB B.MLEN
                .
                .
      This area contains the code that
      describes the header and data to
      be written into the transaction
      file.
                .
                .
                .
      MOV     #TRAN,RO             ;POINT TO TRANSACTION
      MOVB    #BT.MOV,B.TYP(RO)    ;FILL IN TRANSACTION TYPE
      MOVB    #B.MLEN,B.LEN(RO)    ;FILL IN TRANSACTION LENGTH
      MOV     UID,B.UID(RO)        ;USER SESSION ID
      MOV     UID+2,B.UID+2(RO)    ;SECOND HALF
      MOV     ACN,B.ACN(RO)        ;ACCOUNT NUMBER
      MOV     TID,B.TID(RO)        ;TERMINAL ID
      CALL    $SWSTK, 10$          ;SWITCH TO SYSTEM STATE
      CALL    $QTRN2
      RETURN
                                   ;TO USER
10$:
```

Figure 10–1 describes this privileged task interface. Section 10.6.1.1 contains a description of the header area fields.

Figure 10–1: Privileged Task Interface Overview


Calling $QTRAN
Input
User Task
R0
Transaction
PACKET:
B.LEN=
B.TYP=
Data
Process
Calls $QTRAN. User task fills in B.LEN, B.TYP, and Packet transaction data. Header is from user task UAB or TAB.
Output
SYSLOG Queue
Transaction
Header of User Task UAB or TAB
Data from Packet
Calling $QTRN1
Input
User Task
R0
Transaction
PACKET:
B.LEN=
B.TYP=
Data
R1
Task A
UAB or TAB
Header
Process
Calls $QTRN1. User task fills in packet data, B.LEN, and B.TYP. Header is from another task's UAB or TAB (Task A).
Output
SYSLOG Queue
Transaction
Header of UAB or TAB of Task A
Data from Packet
Calling $QTRN2
Input
User Task
R0
Transaction
PACKET:
Filled in Header
:
Filled in Data
Process
Calls $QTRAN2. User task fills in entire header and transaction data within packet.
Output
SYSLOG Queue
Transaction
Header and Data from Packet

ZK-165-81

10.6 Account Blocks

Resource Accounting gathers system usage information by collecting the data in account blocks in secondary pool and by writing each account block as a record in the transaction file. There are two types of Resource Accounting account blocks: *memory resident* and *transaction*.

Memory-resident account blocks are as follows:

- The Task Account Block (TAB) accumulates accounting data per task and stays in secondary pool as long as a task is active.
- The User Account Block (UAB) for each logged-in user accumulates the accounting data and stays in secondary pool as long as the user is logged in.
- The System Account Block (SAB) accumulates data for tasks running on the console terminal (CO:). It stays in secondary pool and accumulates system accounting data as long as Resource Accounting is active.

Transaction account blocks are created and filled when any transaction occurs. They are copied by Accounting into secondary pool and are written immediately into the transaction file. After these operations complete, the space is deallocated back to secondary pool.

10.6.1 Account Block Data Structures

All the account blocks have a *header area*. The fields and offsets for the header area are the same for all types of account blocks (memory resident or transaction). In addition, the memory-resident account blocks (TAB, UAB, and SAB) use the same offsets for the data *accumulation fields* that follow the header. Information that is specific to the TAB, UAB, or SAB appears after the accumulation fields.

Note

> If you plan to write a program that uses the offsets, refer to the [126,10]ACNDF.DOC file on the RSX–11M–PLUS distribution kit for the correct numeric offsets. In MACRO–11 programs, use the symbolic names defined by the ACNDF$ macro in [1,1]EXEMC.MLB. The names will not change even if their associated offsets do.
>
> The files ACNDF.DOC and EXEMC.MLB are not supplied with pregenerated RSX–11M–PLUS (RL02 distribution kits) or Micro/RSX operating systems.

The following subsections provide detailed descriptions of the data structures associated with each type of account block.

10.6.1.1 Account Block Header Area Fields

The header area fields precede the accumulation fields for the TAB, UAB, and SAB. The header area fields also precede all transaction blocks.

The header provides user identification in two fields for every transaction. The fields are as follows:

- The *unique session identification* field is two words long. The first word, the *session identification,* is three RAD50 characters supplied by the user when the system account is created using the ACNT program. The second word is a unique number assigned by the system at login. Thus, it is possible to match every transaction with the login session that produced it.
- The *account number* field is a binary word (entered through the Account File Maintenance Program (ACNT)) that is copied into every transaction.

Resource Accounting does not use either of these fields, except to ensure that they appear in every transaction.

Table 10–2 describes all the header area fields.

Table 10–2: Header Area Fields

Offset	Size or Value	Definition
B.LNK::	.BLKW 1	Link to next packet in SYSLOG queue
B.TYP::	.BLKB 1	Transaction type: 000–127 reserved for DIGITAL use 128–255 reserved for your use
	BT.SAB==1	System Account Block (SAB)
	BT.UAB==2	User Account Block (UAB)
	BT.TAB==3	Task Account Block (TAB)
	BT.SS==11	Accounting startup transaction
	BT.INV==12	Invalid login transaction
	BT.TIM==13	System time change transaction
	BT.ALL==14	Allocate device transaction
	BT.DEA==15	Deallocate device transaction
	BT.MOU==16	Mount device transaction
	BT.DMO==17	Dismount device transaction
	BT.PRT==20	Print despooler transaction
		Reserved
		Reserved
	BT.LOG==23	Login transaction

Table 10–2 (Cont.): Header Area Fields

Offset	Size or Value	Definition
	BT.CRH==24	Crash recovery transaction
	BT.DST==25	Device statistics (Unit Control Block Extension or UCBX)
	BT.RTP==26	Reset transaction parameters
	BT.INP==27	Card reader spooling transaction
B.LEN::	.BLKB 1	Transaction length in bytes
B.TIM::	.BLKW 3	Ending time of transaction (supplied by $QTRAN, $QTRN1, or $QTRN2)
B.HID==.		Start of header identification area
B.UID::	.BLKW 2	Unique session identification First word—RAD50 Second word—binary
B.ACN::	.BLKW 1	Account number
B.TID::	.BLKB 1	American Standard Code for Information Interchange (ASCII) terminal type: V—Virtual T—Real B—Batch C—Console
	.BLKB 1	Unit number
B.HEND==.		End of header identification area

10.6.1.2 Account Block Accumulation Fields

Table 10–3 describes the accumulation field offsets for the UAB, TAB, and SAB. For an overview of the linkages between the three types of account blocks, see Section 10.6.2.

Table 10–3: Accumulation Fields for UAB, TAB, and SAB

Offset	Size or Value	Definition
B.CPU::	.BLKW 2	Total central processing unit (CPU) time used
B.DIR::	.BLKW 2	Total directive count
B.QIO::	.BLKW 2	Total QIO$ count
B.TAS::	.BLKW 2	Total task count (not used in TAB)
	.BLKW 3	Reserved
B.BEG::	.BLKW 3	Beginning/login time

Table 10–3 (Cont.): Accumulation Fields for UAB, TAB, and SAB

Offset	Size or Value	Definition
B.CPUL::	.BLKW 2	CPU time limit in TAB or UAB (not used in SAB)
B.PNT::	.BLKW 1	Pointer to SAB or UAB or is equal to 0. In the SAB, B.PNT is 0 (zero). In the TAB, B.PNT points to the UAB. In the UAB, B.PNT points to the SAB (if you specify SYSTEM_STATISTICS:YES) or is 0 (if you specify SYSTEM_STATISTICS:NO).
B.STM::	.BLKB 1	Status mask:
	BS.ACT==200	Control block active (UAB, TAB)
	BS.CRH==100	Record from TMP file after crash (UAB and SAB)
	BS.LGO==40	Logged out with outstanding activities (UAB only)
	BS.CO==40	Task's TI: is CO: (TAB only)
	BS.TML==20	TAB exists only for time limit (TAB only). TABs with this bit set will not be written to the transaction file
	BS.SIL==20	Silent login/logout (UAB only)
	BS.ZER==10	Last CPU interval was zero length (TAB and UAB)
	BS.SCN==4	Transaction ready for write to scan file (UAB and SAB)

10.6.1.3 Memory-Resident Account Blocks and Offsets

Figures 10–2, 10–3, and 10–4 illustrate the data structures that are unique to the Task Account Blocks (TAB), the System Account Block (SAB), and the User Account Blocks (UAB), respectively. The offsets for the data structures are described in the tables that follow each figure.

Figure 10–2: Task Account Block

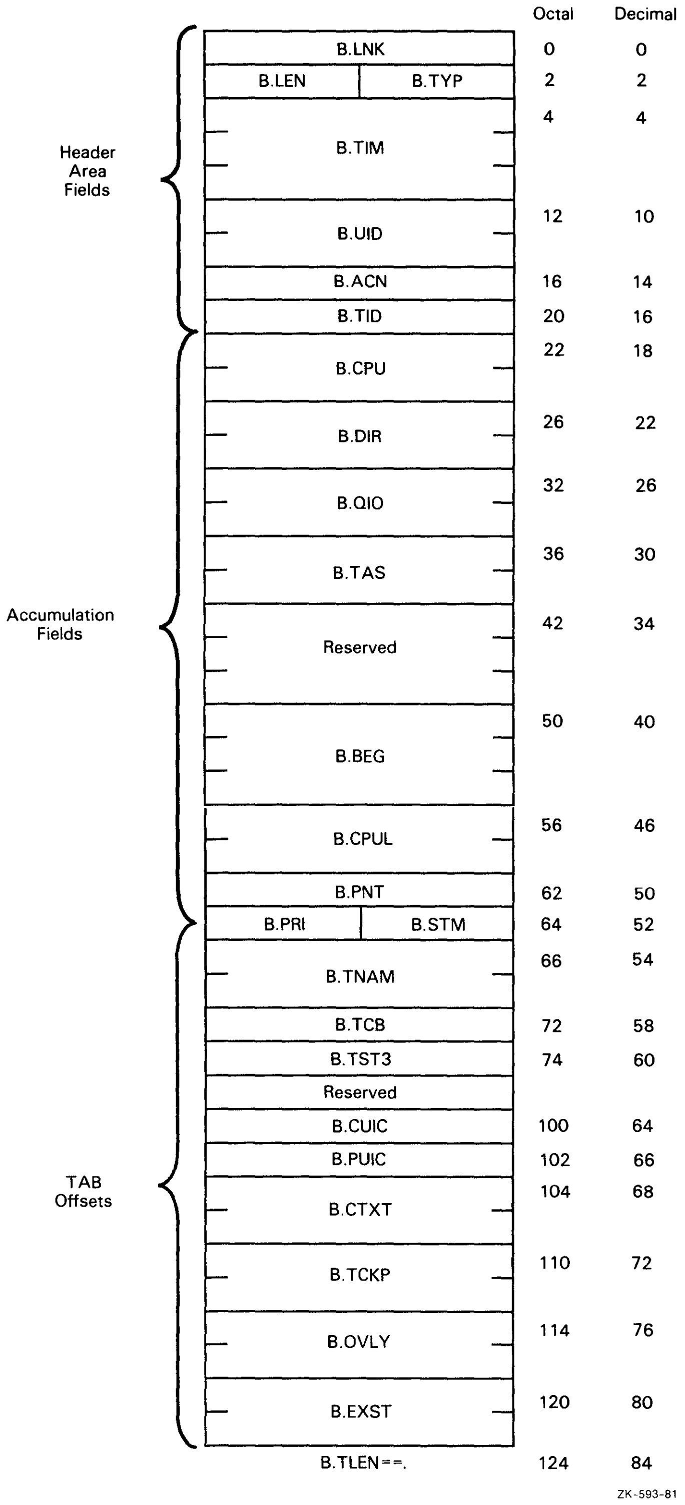


Table 10–4 defines the TAB offsets. Note that TABs must end on a word boundary.

Table 10–4: Task Account Block Offsets

Offset	Size or Value	Definition
B.PRI::	.BLKB 1	Highest running priority
B.TNAM::	.BLKW 2	Task name
B.TCB::	.BLKW 1	Task Control Block (TCB) address
B.TST3::	.BLKW 1	T.ST3 from task's TCB
	.BLKW 1	Reserved for future status bits
B.CUIC::	.BLKW 1	Current User Identification Code (UIC) of task
B.PUIC::	.BLKW 1	Protection UIC of task
B.CTXT::	.BLKW 2	Number of context loads
B.TCKP::	.BLKW 2	Times task has been checkpointed
B.OVLY::	.BLKW 2	Number of disk overlay loads
B.EXST::	.BLKW 2	Exit status and abort code (a list of abort codes follows this table)
B.TLEN==.		TAB length

Valid abort codes for the TAB offset B.EXST are as follows:

Abort Code	Definition
S.CACT=-4$_{10}$	Task still active
S.CEXT=-2$_{10}$	Task exited normally
S.COAD=0$_{10}$	Odd address and traps to 4
S.CSGF=2$_{10}$	Segment fault
S.CBPT=4$_{10}$	Breakpoint or trace trap
S.CIOT=6$_{10}$	IOT instruction
S.CILI=8$_{10}$	Illegal or reserved instruction
S.CEMT=10$_{10}$	Non-RSX EMT instruction
S.CTRP=12$_{10}$	TRAP instruction
S.CFLT=14$_{10}$	PDP–11/40 floating-point exception
S.CSST=16$_{10}$	Synchronous system trap (SST) abort bad stack
S.CAST=18$_{10}$	Asynchronous system trap (AST) abort bad stack

Abort Code	Definition
S.CABO=20_{10}	Abort via directive
S.CLRF=22_{10}	Task load request failure
S.CCRF=24_{10}	Task checkpoint read failure
S.IOMG=26_{10}	Task exit with outstanding I/O
S.PRTY=28_{10}	Task memory parity error
S.CPMD=30_{10}	Task aborted with Postmortem Dump (PMD) request
S.CELV=32_{10}	TI: virtual terminal was eliminated
S.CINS=34_{10}	Task installed in two different systems
S.CAFF=36_{10}	Task aborted due to bad affinity (required by runs are off line or not present)
S.CCSM=38_{10}	Bad parameters or bad stack
S.COTL=40_{10}	Task has run over its time limit
S.CTKN=42_{10}	Abort via directive with no Task Termination Notification program (TKTN) message

Figure 10-3: User Account Block

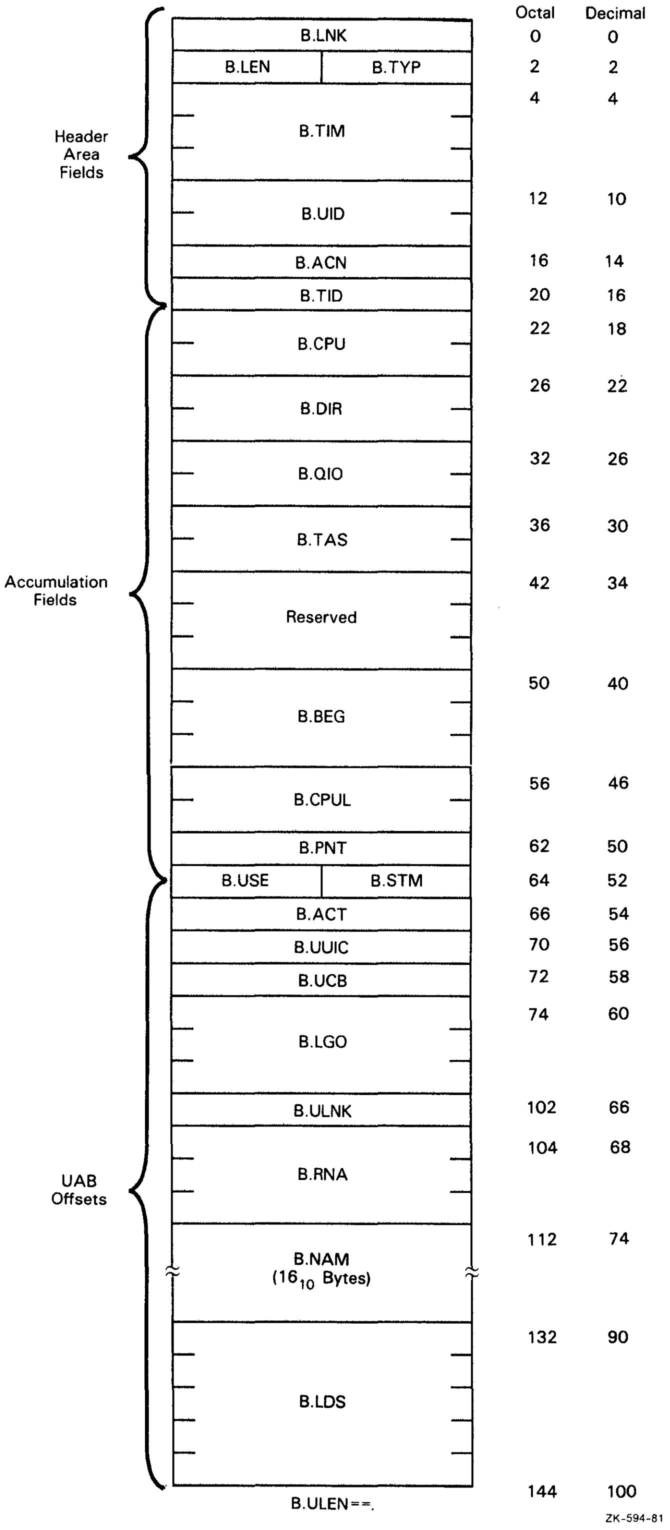
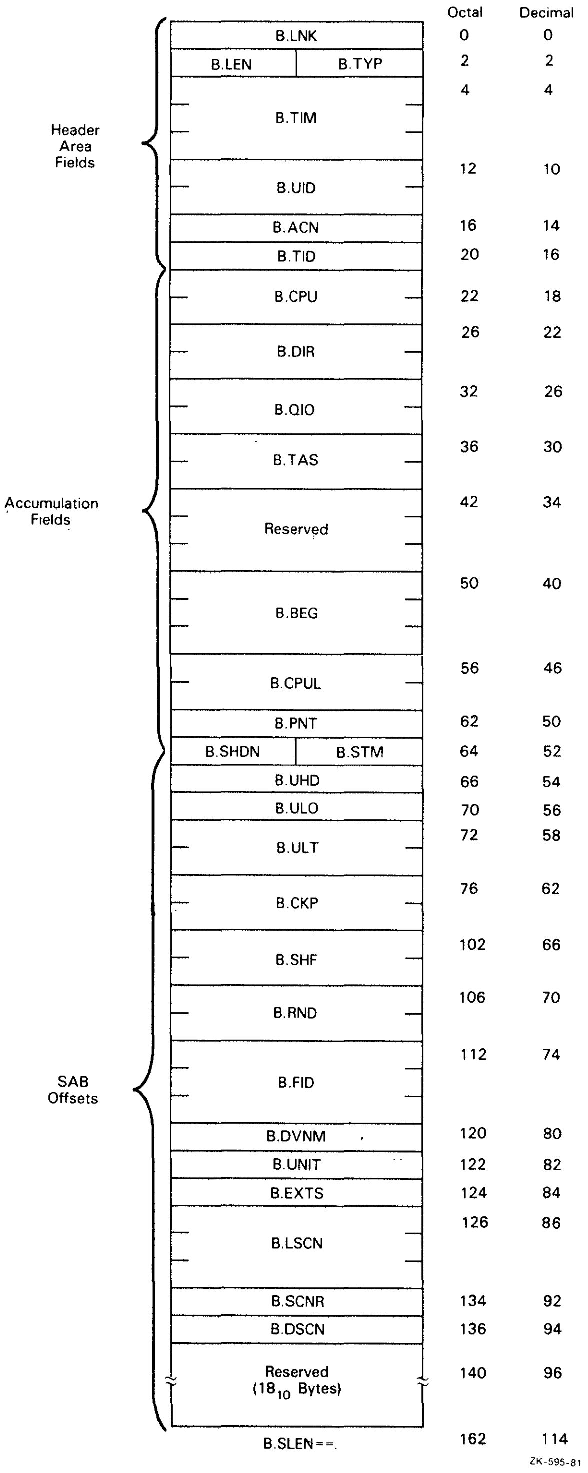


Table 10–5 defines the User Account Block (UAB) offsets. Note that UABs must end on a word boundary.

Table 10–5: User Account Block Offsets

Offset	Size or Value	Definition
B.USE::	.BLKB 1	Use count
B.ACT::	.BLKW 1	Number of currently active tasks
B.UUIC::	.BLKW 1	Login UIC
B.UCB::	.BLKW 1	Pointer to Unit Control Block (UCB)
B.LGO::	.BLKW 3	Logout time
B.ULNK::	.BLKW 1	Link to next UAB
B.RNA::	.BLKW 3	Location in system account file (inaccurate if the account file is changed by additions, deletions, or sorts): First word—Offset Second word—Upper virtual block number Third Word—Lower virtual block number
B.NAM::	.BLKB 14_{10}	Last name of user
	.BLKB 1	First initial of user
	.BLKB 1	Flag byte for UAB (for example, BS.SIL)
B.LDS::	.BLKB 10	Login directory string
B.ULEN==.		UAB length

Figure 10–4: System Account Block



	Octal	Decimal
B.LNK	0	0
B.LEN B.TYP	2	2
B.TIM	4	4
B.UID	12	10
B.ACN	16	14
B.TID	20	16
B.CPU	22	18
B.DIR	26	22
B.QIO	32	26
B.TAS	36	30
Reserved	42	34
B.BEG	50	40
B.CPUL	56	46
B.PNT	62	50
B.SHDN B.STM	64	52
B.UHD	66	54
B.ULO	70	56
B.ULT	72	58
B.CKP	76	62
B.SHF	102	66
B.RND	106	70
B.FID	112	74
B.DVNM	120	80
B.UNIT	122	82
B.EXTS	124	84
B.LSCN	126	86
B.SCNR	134	92
B.DSCN	136	94
Reserved (18_{10} Bytes)	140	96
B.SLEN==.	162	114

Table 10–6 defines the SAB offsets.

Table 10–6: System Account Block Offsets

Offset	Size or Value	Definition
B.SHDN::	.BLKB 1	Accounting shutdown reason code: 1–Maintenance 2–Reboot 3–Scheduled shutdown 4–Accounting shutdown by Shutup task 5–Other
B.UHD::	.BLKW 1	UAB listhead
B.ULO::	.BLKW 1	Number of users currently logged in
B.ULT::	.BLKW 2	Total number of logins
B.CKP::	.BLKW 2	Total number of checkpoints
B.SHF::	.BLKW 2	Total number of Shuffler (SHF) runs
B.RND::	.BLKW 2	Number of CPU intervals rounded up to 1/2
B.FID::	.BLKW 3	File ID of transaction file
B.DVNM::	.BLKB 2	Device of transaction file
B.UNIT::	.BLKW 1	Unit of transaction file
B.EXTS::	.BLKW 1	Extend size for transaction file
B.LSCN::	.BLKW 3	Time of last scan
B.SCNR::	.BLKW 1	Scan rate in seconds
B.DSCN::	.BLKW 1	Device statistical scan rate in seconds
	.BLKW 9_{10}	Reserved
B.SLEN==.		SAB length

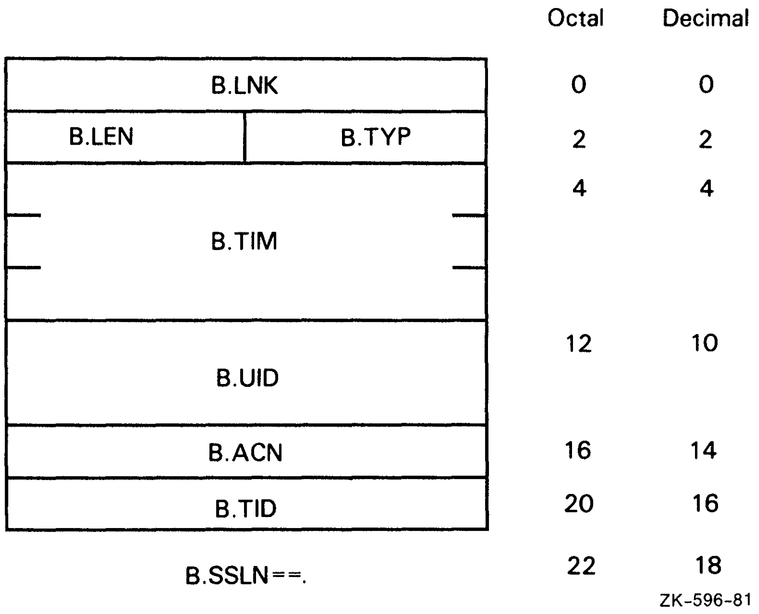
10.6.1.4 Transaction Block Diagrams and Offsets

This section describes the offsets for the following transaction account blocks:

Transaction Block	Format	Offset Definitions
Accounting Startup	See Figure 10–5	
Reset Accounting	See Figure 10–6	See Table 10–7
Crash Recovery	See Figure 10–7	See Table 10–8
Login	See Figure 10–8	See Table 10–9
Invalid Login	See Figure 10–9	See Table 10–10

Transaction Block	Format	Offset Definitions
Allocate Device	See Figure 10–10	See Table 10–11
Deallocate Device	See Figure 10–11	See Table 10–11
Mount Device	See Figure 10–12	See Table 10–11
Dismount Device	See Figure 10–13	See Table 10–11
System Time Change	See Figure 10–14	See Table 10–12
Print Despooler	See Figure 10–15	See Table 10–13
Card Reader Spooling	See Figure 10–16	See Table 10–14
Device Statistics	See Figure 10–17	See Table 10–15

Figure 10–5: Accounting Startup Transaction Block



In the Accounting Startup Transaction Block, the offset B.SSLN==. defines the transaction length.

Figure 10–6: Reset Accounting Transaction Block

		Octal	Decimal
B.LNK		0	0
B.LEN	B.TYP	2	2
B.TIM		4	4
B.UID		12	10
B.ACN		16	14
B.TID		20	16
B.OFID		22	18
B.ODNM		30	24
B.OUNT		32	26
B.NFID		34	28
B.NDNM		42	34
B.NUNT		44	36
B.OEXS		46	38
B.NEXS		50	40
B.OSCR		52	42
B.NSCR		54	44
B.ODSC		56	46
B.NDSC		60	48
B.RTLN==.		62	50

ZK-597-81

Table 10–7 defines the offsets for the Reset Accounting Transaction Block.

Table 10–7: Reset Accounting Transaction Block Offsets

Offset	Size or Value	Definition
B.OFID::	.BLKW 3	File ID of old transaction file
B.ODNM::	.BLKB 2	Device of old transaction file
B.OUNT::	.BLKW 1	Unit of old transaction file
B.NFID::	.BLKW 3	File ID of new transaction file
B.NDNM::	.BLKB 2	Device of new transaction file
B.NUNT::	.BLKW 1	Unit of new transaction file
B.OEXS::	.BLKW 1	Extend size for old transaction file
B.NEXS::	.BLKW 1	Extend size for new transaction file
B.OSCR::	.BLKW 1	Old scan rate in seconds
B.NSCR::	.BLKW 1	New scan rate in seconds
B.ODSC::	.BLKW 1	Old statistical scan rate
B.NDSC::	.BLKW 1	New statistical scan rate
B.RTLN==		Transaction length

Figure 10-7: Crash Recovery Transaction Block

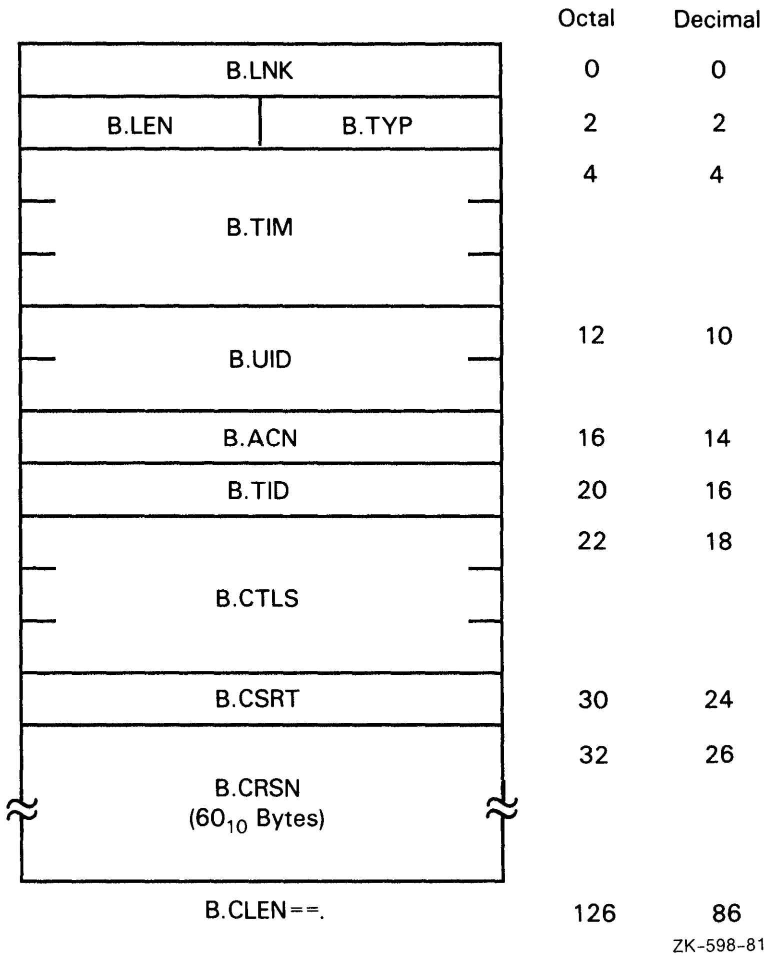


Table 10-8 defines the offsets for the Crash Recovery Transaction Block.

Table 10-8: Crash Recovery Transaction Block

Offset	Size or Value	Definition
B.CTLS::	.BLKW 3	Time of last scan before crash
B.CSRT::	.BLKW 1	Scan rate before crash
B.CRSN::	.BLKB 60_{10}	ASCII text explaining crash
B.CLEN==.		Transaction length

Figure 10–8: Login Transaction Block

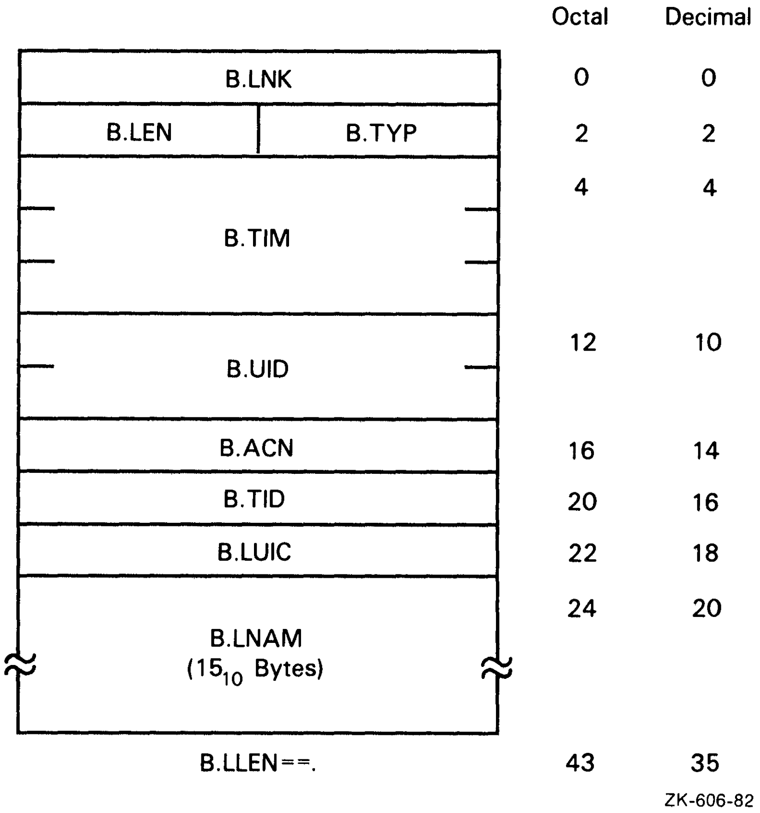


Table 10–9 defines the offsets for the Login Transaction Block.

Table 10–9: Login Transaction Block Offsets

Offset	Size or Value	Definition
B.LUIC::	.BLKW 1	Login UIC
B.LNAM::	.BLKB 14_{10}	User's last name
	.BLKB 1	User's first initial
B.LLEN==.		Transaction length

Figure 10–9: Invalid Login Transaction Block

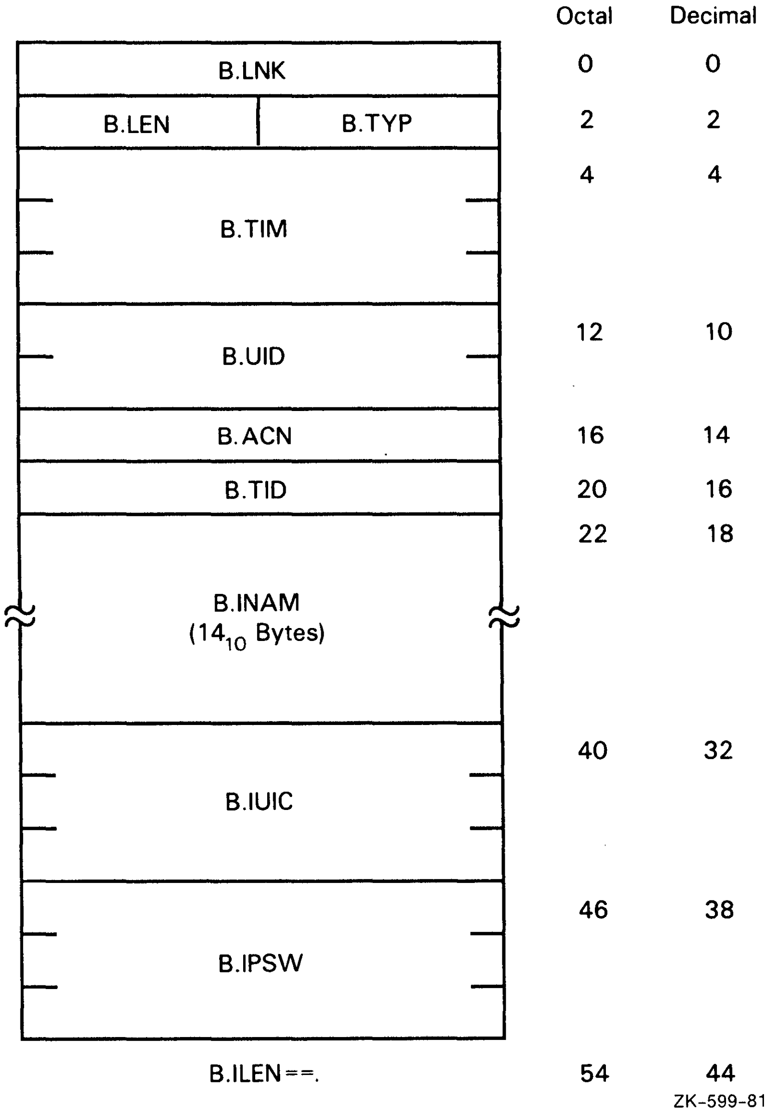


Table 10–10 defines the offsets for the Invalid Login Transaction Block.

Table 10-10: Invalid Login Transaction Block Offsets

Offset	Size or Value	Definition
B.INAM::	.BLKB 14_{10}	Name from login line
B.IUIC::	.BLKB 6_{10}	UIC from login line
B.IPSW::	.BLKB 6_{10}	Password from login line
B.ILEN==.		Transaction length

Figure 10-10: Allocate Device Transaction Block

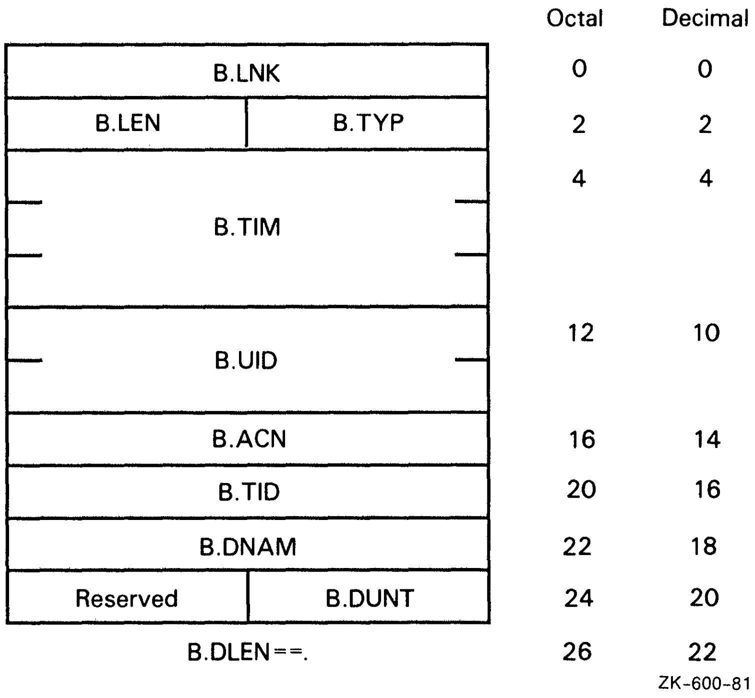
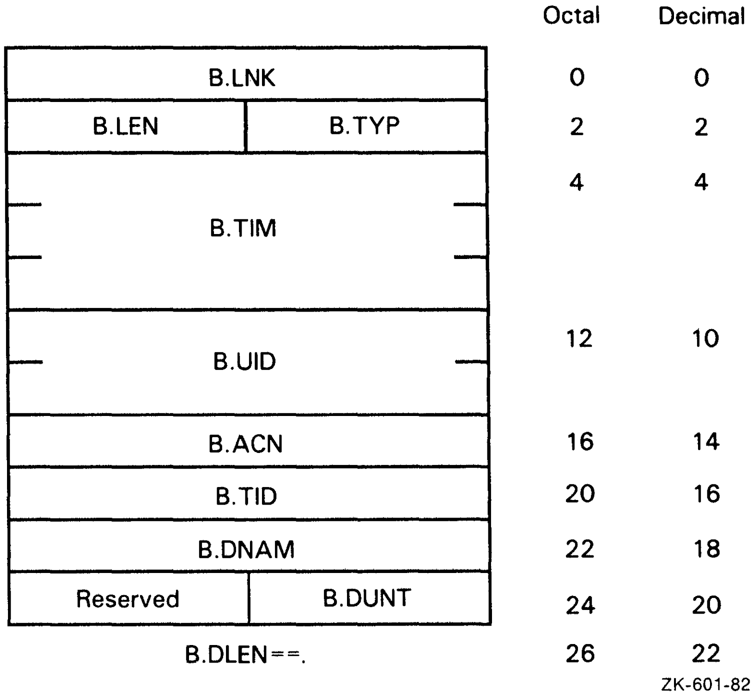


Figure 10–11: Deallocate Device Transaction Block



		Octal	Decimal
B.LNK		0	0
B.LEN	B.TYP	2	2
B.TIM		4	4
B.UID		12	10
B.ACN		16	14
B.TID		20	16
B.DNAM		22	18
Reserved	B.DUNT	24	20
B.DLEN==.		26	22

ZK-601-82

Figure 10–12: Mount Device Transaction Block

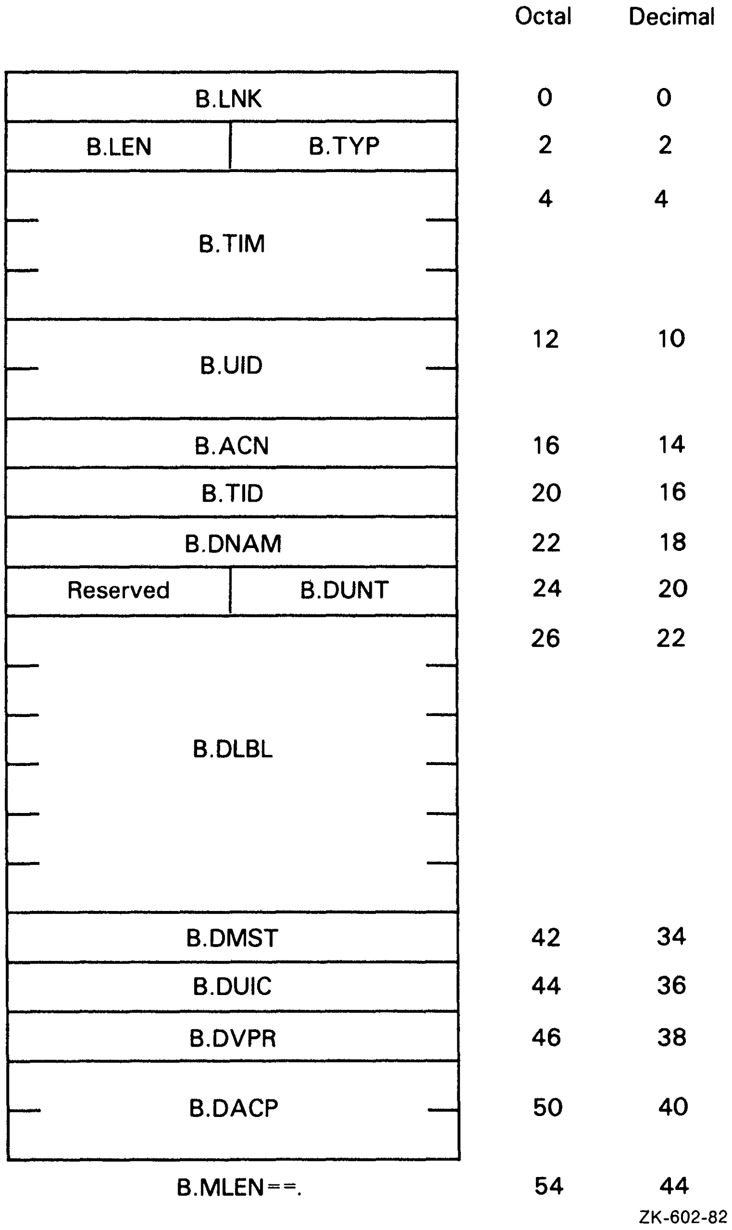


Figure 10–13: Dismount Device Transaction Block

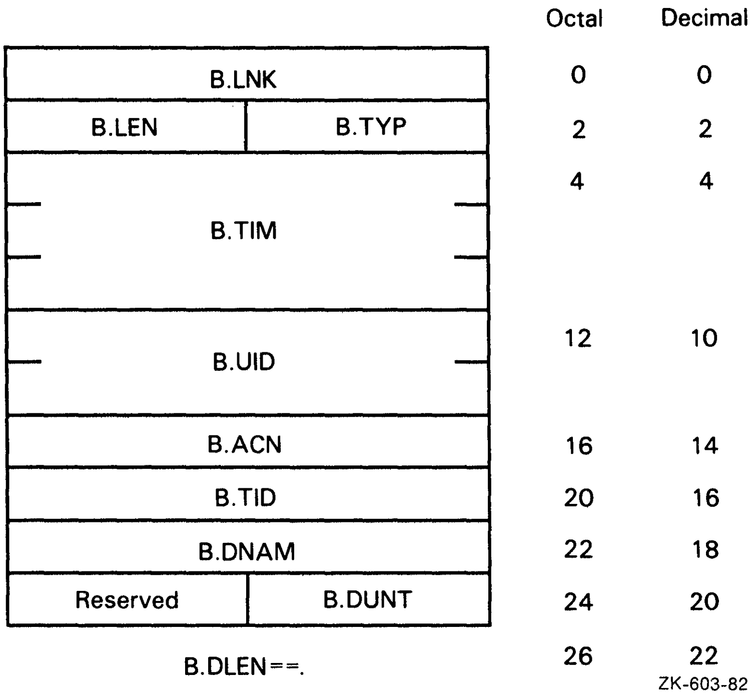


Table 10–11 defines the offsets for the Allocate, Deallocate, Mount, and Dismount Device Transaction Blocks. A list of offsets that appear only for MOUNT transactions follows the table.

Table 10–11: Device Transaction Offsets

Offset	Size or Value	Definition
B.DNAM::	.BLKW 1	ASCII device name
B.DUNT::	.BLKB 1	Octal device unit number
B.DLEN==.		Transaction length for ALLOCATE, DISMOUNT, and DEALLOCATE

The following offsets appear only for MOUNT transactions:

<table>
<tr><th>Offset</th><th>Size or Value</th><th>Definition</th></tr>
<tr><td></td><td>.BLKB 1</td><td>Unused byte</td></tr>
<tr><td>B.DLBL::</td><td>.BLKW 6</td><td>Volume label</td></tr>
<tr><td>B.DMST::</td><td>.BLKW 1</td><td>Mount status bits</td></tr>
<tr><td>B.DUIC::</td><td>.BLKW 1</td><td>Owner UIC</td></tr>
<tr><td>B.DVPR::</td><td>.BLKW 1</td><td>Volume protection code
The volume protection codes are as follows:
<table>
<tr><th>Bits</th><th>Protection</th></tr>
<tr><td>0–3</td><td>System</td></tr>
<tr><td>4–7</td><td>Owner</td></tr>
<tr><td>8–11</td><td>Group</td></tr>
<tr><td>12–15</td><td>World</td></tr>
</table>
The bit settings within each protection group are as follows:
<table>
<tr><th>Bit</th><th>Access denied (if bit set)</th></tr>
<tr><td>0</td><td>Read</td></tr>
<tr><td>1</td><td>Write</td></tr>
<tr><td>2</td><td>Create</td></tr>
<tr><td>3</td><td>Delete</td></tr>
</table>
</td></tr>
<tr><td>B.DACP::</td><td>.BLKW 2</td><td>Name of Ancillary Control Processor (ACP) for device</td></tr>
<tr><td>B.MLEN==.</td><td></td><td>Length of MOUNT transactions</td></tr>
</table>

Figure 10–14: System Time Change Transaction Block

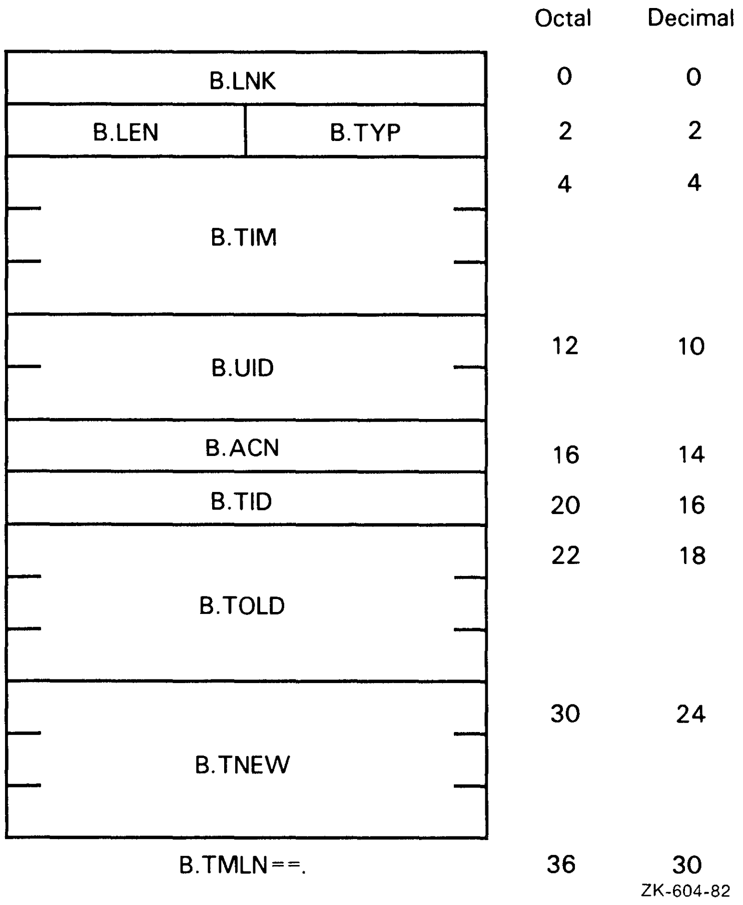


Table 10–12 defines the offsets for the System Time Change Transaction Block.

Table 10–12: System Time Change Transaction Block Offsets

Offset	Size or Value	Definition
B.TOLD::	.BLKB 6	Old time (yr, mon, day, hr, min, sec)
B.TNEW::	.BLKB 6	New time (yr, mon, day, hr, min, sec)
B.TMLN==.		Transaction length

Figure 10–15: Print Despooler Transaction Block

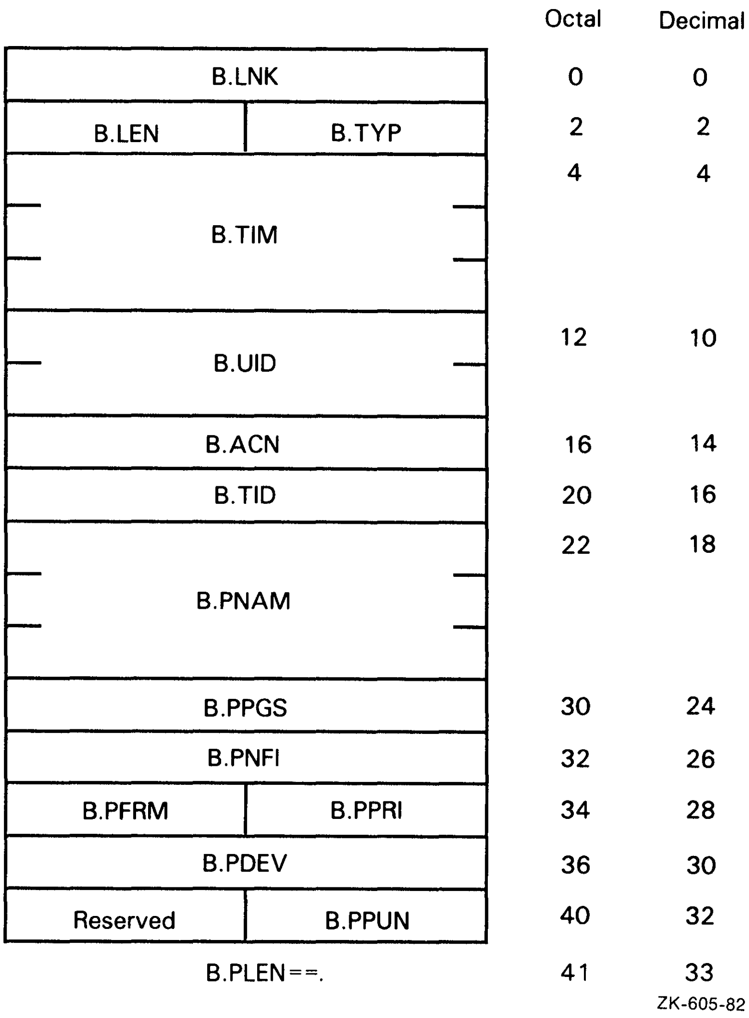


Table 10–13 defines the offsets for the Print Despooler Transaction Block.

Table 10–13: Print Despooler Transaction Block Offsets

Offset	Size or Value	Definition
B.PNAM::	.BLKW 3	Print job name (RAD50)
B.PPGS::	.BLKW 1	Page count
B.PNFI::	.BLKW 1	Number of files printed
B.PFRM::	.BLKB 1	Form number

Table 10-13 (Cont.): Print Despooler Transaction Block Offsets

Offset	Size or Value	Definition
B.PPRI::	.BLKB 1	Print priority
B.PDEV::	.BLKW 1	Print device name (ASCII)
B.PPUN::	.BLKB 1	Unit number of print device
B.PLEN==.		Transaction length

Figure 10-16: Card Reader Spooling Transaction Block

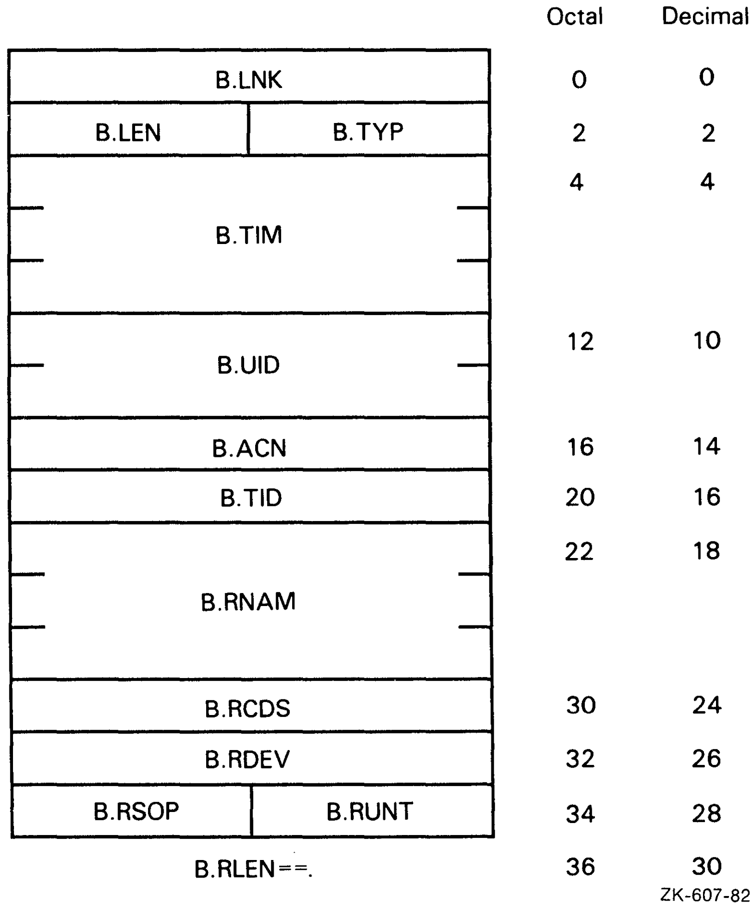


Table 10-14 defines the offsets for the Card Reader Spooling Transaction Block.

Table 10–14: Card Reader Spooling Transaction Block Offsets

Offset	Size or Value	Definition
B.RNAM::	.BLKW 3	Batch or print job name
B.RCDS::	.BLKW 1	Number of cards read
B.RDEV::	.BLKW 1	Reader device name (ASCII)
B.RUNT::	.BLKB 1	Unit number of reader device
B.RSOP::	.BLKB 1	Submit or print (0=SUBMIT, 1=PRINT)
B.RLEN==.		Transaction length

Figure 10-17: Device Statistics Block

		Octal	Decimal
B.LNK		0	0
B.LEN	B.TYP	2	2
B.TIM		4	4
B.UID		12	10
B.ACN		16	14
B.TID		20	16
X.NAME		22	18
X.IOC		26	22
X.ERSL	X.ERHL	32	26
X.ERHC	X.ERSC	34	28
X.WCNT		36	30
X.CYLC		42	34
X.CCYL		46	38
X.FLIM	X.FCUR	50	40
X.DNAM		52	42
Reserved	X.UNIT	54	44
X.LGTH==.		56	46

ZK-902-82

Table 10–15 defines the offsets for the Device Statistics Block.

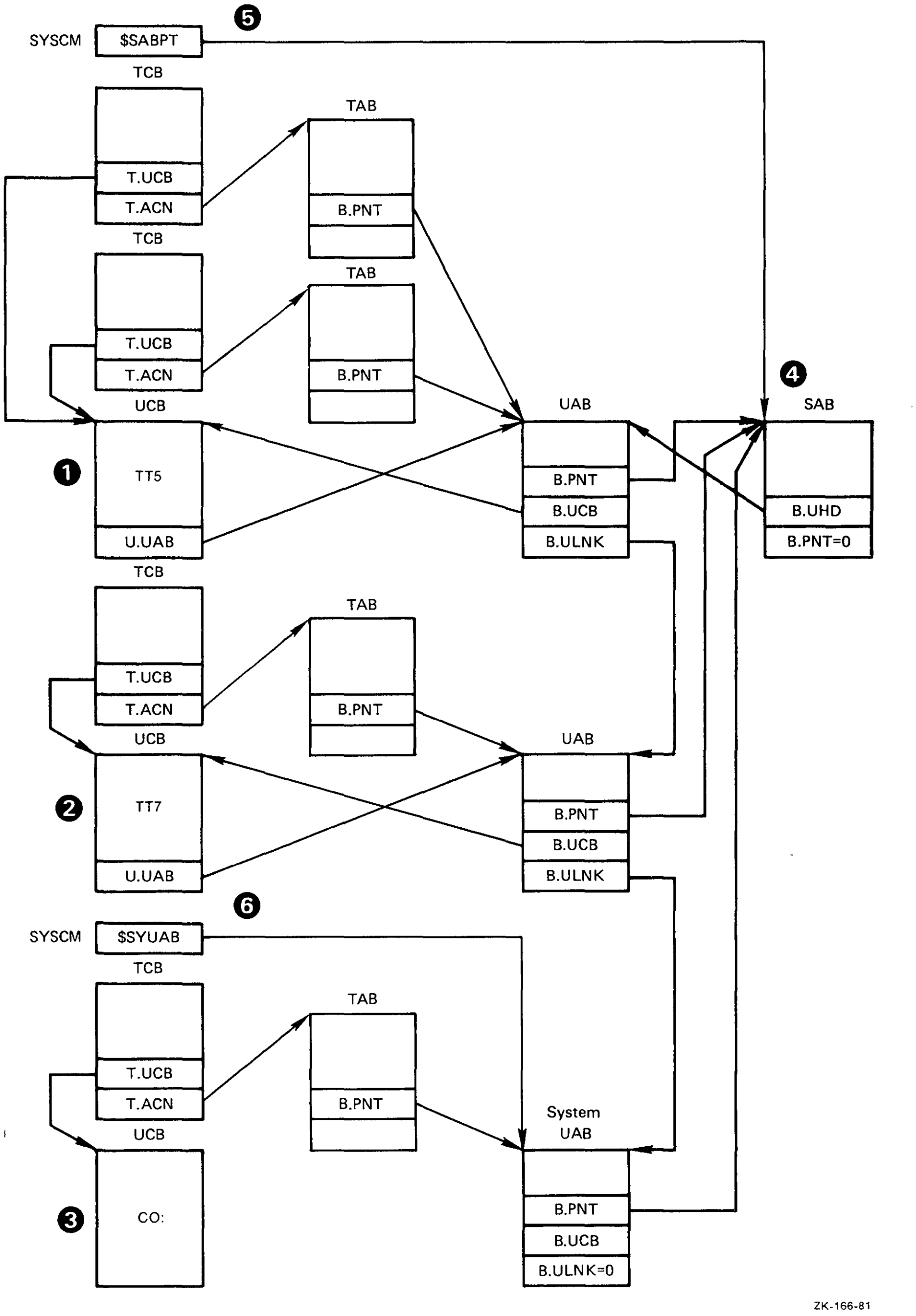
Table 10–15: Device Statistics Block Offsets

Offset	Size or Value	Definition
X.NAME::	.BLKW 2	Drive name in Radix–50
X.IOC::	.BLKW 2	I/O Count
X.ERHL::	.BLKB 1	Hard error limit
X.ERSL::	.BLKB 1	Soft error limit
X.ERSC::	.BLKB 1	Soft error count
X.ERHC::	.BLKB 1	Hard error count
X.WCNT::	.BLKW 2	Words transferred count
X.CYLC::	.BLKW 2	Cylinders crossed count
X.CCYL::	.BLKW 1	Current cylinder
X.FCUR::	.BLKB 1	Current fairness count
X.FLIM::		Fairness count limit
X.DSKD::	.BLKB 1	Disk direction (high bit 1=out)
X.DNAM::	.BLKB 1	Device name for accounting
X.UNIT::	.BLKB 1	Unit number for accounting
	.BLKB 1	Unused
X.LGTH==.		Length of the Unit Control Block Extension (UCBX)

10.6.2 Account Block Linkages

Figure 10–18 provides an overview of the linkages between the system UAB, the SAB, the TABs, and the UABs.

Figure 10–18: Account Block Linkages



❶ Terminal TT5, which is indicated by the Unit Control Block (UCB), is the terminal of the most recently logged-in user. The user of this terminal has started two tasks, which are indicated by the two Task Control Blocks (TCBs) that contain pointers to the UCB (T.UCB) and each associated Task Account Block (TAB; T.ACN). Each TAB then points to the terminal's User Account Block (UAB). The total usage of terminal TT5 is totaled in the associated UAB.

❷ Terminal TT7 is that of the next most recently logged-in user. The user has started a task indicated by the single TCB, which points to a TAB that points to a UAB. The total usage of terminal TT7 is totaled in the associated UAB.

❸ The CO: console terminal has associated with it a single TCB that indicates a system task is running. In effect, the system is also a user and has system usage accumulated at the task level in the TAB and totaled in the UAB associated with the CO: terminal.

Note

Although all active UCBs point to the appropriate UABs using U.UAB, this is not true for CO:. U.UAB does not exist for CO:; instead, it is reflected by location $SYUAB.

❹ The System Account Block (SAB) contains the usage totals for all the users including the system itself. The B.UHD pointer in the SAB points to the UAB of the most recently logged-in user. B.UHD and the B.ULNK words in each UAB are the linkages that are used when Resource Accounting periodically writes the UABs to the scan file.

❺ The pointer to the SAB ($SABPT) is in the Executive module SYSCM.

❻ The pointer to the system UAB ($SYUAB) is in the Executive module SYSCM.

10.7 System Resources Used

The following sections discuss the amount of system space that Resource Accounting uses. Secondary pool usage varies from a certain minimum to an amount determined by the number of logged-in users and event-associated transactions.

10.7.1 Executive Space Requirements

Accounting support uses approximately 532 words of Executive instruction and data space. Space in system pool is required for one active task and one open file.

10.7.2 Secondary Pool Usage

Each logged-in user requires two blocks (32 words per block) of secondary pool for the UAB; that is, two blocks for each logged-in terminal or virtual terminal. Each task running in the system requires two blocks for the TAB, if task accounting is enabled. Resource Accounting needs two blocks for the SAB and two blocks for the system UAB. Non-memory-resident events, such as device mounts, require one block for a brief period of time.

10.8 Resource Accounting Messages

Accounting messages are numbered from 1_{10} to 79_{10}. However, not all of the numbers in this range are used. The number appears at the start of the accounting message.

Format

```
hh:mm:ss aaaTnn nn. *aaaaaa* message...
```

Display Parameters

hh:mm:ss

Specifies time-stamping of the message occurrence in the format hh:mm:ss. For example, at 10 minutes and 36 seconds after 1:00 P.M., time-stamping is 13:10:36. Note that the hours are recorded in 24-hour format.

aaaTnn

Specifies the task issuing the message and the terminal executing the task. The first three letters of the task name are indicated by aaa. Tnn is the terminal number of the terminal executing the task. The word SYSLOG appears if the message originated from that task.

nn.

Specifies the message number.

aaaaaa

Specifies the message severity code, in one of four forms. This code may not be present; if it is not, the message is an informational message. The four forms of the severity code are as follows:

DIAG	Information only.
WARN	Possible error condition.
ERROR	Command did not succeed.
FATAL	Extreme error condition.

The remainder of this section lists the Resource Accounting messages by message number:

nn:nn:nn aaaTnn 1. Undefined message number nn. at PC aaaaaa

Explanation: The message number is a decimal number specified by nn. The location is specified by the address aaaaaa. This is an internal Resource Accounting error.

User Action: Submit a Software Performance Report (SPR).

nn:nn:nn aaaTnn 2. Directive error. $DSW=nnn PC is just before aaaaaa

Explanation: The directive error code is an octal number specified by nnn. The program counter (PC) contains an address just preceding the octal address aaaaaa. This message provides additional information to a message that will appear with it.

User Action: Look up the directive error code in the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual* to obtain more information.

**nn:nn:nn aaaTnn 4. FCS aaa ERROR CODE = nnn
FILE: DBn: [???,???]filename.filetype;v**

Explanation: The word "I/O" or "DSW" is specified by aaa. The error code is specified by nnn. The UIC is inaccessible and appears as question marks.

User Action: Refer to the *RSX-11M-PLUS Mini-Reference* for an explanation of the error code.

nn:nn:nn aaaTnn 5. •ERROR• Illegal filename

Explanation: The file name specified in the START/ACCOUNTING command is invalid.

User Action: Correct the file name and reenter the command.

nn:nn:nn aaaTnn 6. •FATAL• Can't open transaction file

```
                    .
                    .
                    .
                [message]
```

Explanation: The transaction file could not be created. An additional message explaining the reason follows.

User Action: Refer to the description of the second message to determine the cause of the error.

nn:nn:nn aaaTnn 7. •FATAL• Can't open transaction file (by id)

```
                    .
                    .
                    .
                [message]
```

Explanation: After it was created, the transaction file became inaccessible. An additional message explaining the reason follows this message.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command. Then, refer to the description of the second message to determine the cause of the error. Correct the error before restarting Resource Accounting.

nn:nn:nn aaaTnn 8. •FATAL• Can't write transaction file

```
                    .
                    .
                    .
                [message]
```

Explanation: The transaction file could not be written. An additional message explaining the reason follows this message.

Resource Accounting cannot recover from this error. Accounting data is lost because the file cannot be written.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command. Then, refer to the description of the second message to determine the cause of the error. Correct the error before restarting Resource Accounting.

nn:nn:nn aaaTnn 9. •ERROR• Can't close transaction file

Explanation: The transaction file could not be closed. Resource Acounting cannot recover from this error. Accounting data is lost because the file cannot be closed.

User Action: If possible, determine why the file cannot be closed. Restart Resource Accounting for a short period of time to see if the error recurs.

nn:nn:nn aaaTnn 10. •FATAL• Can't init new blocks

Explanation: The transaction file cannot be written when the file is being extended. Resource Accounting cannot recover from this error. Accounting data is lost because the file cannot be written to be extended.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command. Correct the error before restarting Resource Accounting.

nn:nn:nn aaaTnn 11. •FATAL• Can't init new blocks. I/O error = nnn

Explanation: The transaction file cannot be written when the file is being extended. Resource Accounting cannot recover from this error. Accounting data is lost because the file cannot be written to be extended.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command. Locate the I/O error code in the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual;* then, correct the error before restarting Resource Accounting.

nn:nn:nn aaaTnn 12. •FATAL• Can't write attributes. I/O error = nnn

Explanation: The transaction file cannot be written when the file is being extended. Resource Accounting cannot recover from this error. Accounting data is lost because the file cannot be written to be extended.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command. Locate the I/O error code in the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual;* then, correct the error before restarting Resource Accounting.

nn:nn:nn aaaTnn 13. •FATAL• Can't write attributes

Explanation: The transaction file cannot be written when the file is being extended. Resource Accounting cannot recover from this error. Accounting data is lost because the file cannot be written to be extended.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command. Correct the error before restarting Resource Accounting.

nn:nn:nn aaaTnn 14. •FATAL• Secondary pool allocation failure

Explanation: There is insufficient secondary pool remaining to start Resource Accounting.

User Action: Use the Virtual Monitor Console Routine (VMR; see Chapter 5) to increase the secondary pool partition size. Then, rebootstrap the system.

nn:nn:nn aaaTnn 15. •FATAL• Bad SYSLOG queue

Explanation: The SYSLOG task has outstanding I/O during accounting startup.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command and restart Resource Accounting.

nn:nn:nn aaaTnn 16. •ERROR• Accounting already active

Explanation: Accounting is already active.

User Action: No user action is required.

nn:nn:nn aaaTnn 17. •FATAL• Error spawning SYSLOG

```
                    .
                    .
                    .
               [message]
```

Explanation: The SYSLOG task could not be started. An additional message explaining the reason follows this message.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command. Then, refer to the description of the second message to determine the cause of the error. Correct the error before restarting Resource Accounting.

nn:nn:nn aaaTnn 18. Accounting system not started

```
                    .
                    .
                    .
               [message]
```

Explanation: Resource Accounting was not started. An additional message explaining the reason follows this message.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command. Then, refer to the description of the second message to determine the cause of the error. Correct the error before restarting Resource Accounting.

nn:nn:nn aaaTnn 21. SYSLOG exit with status = n.

```
                    .
                    .
                    .
               [message]
```

Explanation: The exit status of the SYSLOG task is represented by the decimal number n. See the accompanying SYSLOG message.

User Action: Correct the problem, if there is one; then, reenter the command.

nn:nn:nn aaaTnn 22. •ERROR• SYSLOG not installed

Explanation: The SYSLOG task was not installed when you entered the START/ACCOUNTING command.

User Action: Install SYSLOG and reenter the START/ACCOUNTING command.

nn:nn:nn aaaTnn 23. •ERROR• Error bit set. Cleanup before restart.

Explanation: Resource Accounting had an error and cannot be restarted until you issue the STOP/ACCOUNTING CLEAN_UP command.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command and restart Accounting.

nn:nn:nn aaaTnn 25. •ERROR• Accounting did not crash

Explanation: You entered A STOP/ACCOUNTING CLEAN_UP command that was not needed because the system did not fail.

User Action: No user action is required.

nn:nn:nn aaaTnn 27. •FATAL• System command handling error

Explanation: The Executive directive Get MCR command line (GMCR$) failed.

User Action: No user action is required. The ...ACC task executed, so no command line was queued. Access to this task is only valid with the specified accounting commands.

nn:nn:nn aaaTnn 28. •ERROR• Privilege violation

Explanation: You did not have the privilege for the specified operation.

User Action: No user action is requrired. Nonprivileged users can access the accounting data for their terminal only.

nn:nn:nn aaaTnn 29. •ERROR• Accounting not supported in system

Explanation: Resource Accounting cannot be used unless it is selected as an option during system generation.

User Action: Include support for Resource Accounting during system generation.

nn:nn:nn aaaTnn 30. •ERROR• Illegal command or parameter

Explanation: You entered a command incorrectly.

User Action: Reenter the command, using the correct syntax.

nn:nn:nn aaaTnn 31. •ERROR• Command or parameter ambiguous

Explanation: You abbreviated a parameter with too few letters to be uniquely identified.

User Action: Reenter the command, specifying more letters for the parameter name.

nn:nn:nn aaaTnn 32. •ERROR• Syntax error

Explanation: You entered a command incorrectly.

User Action: Reenter the command with the correct syntax.

nn:nn:nn aaaTnn 33. •ERROR• Terminal I/O error

Explanation: A terminal I/O error occurred during a prompt or a read to the terminal.

User Action: Reenter the command. If the error persists, submit an SPR to DIGITAL.

nn:nn:nn aaaTnn 34. •ERROR• Repeated or contradictory parameter

Explanation: You entered an invalid or duplicate parameter combination.

User Action: Enter the correct command parameter or parameters.

nn:nn:nn aaaTnn 35. •ERROR• Required parameter missing

Explanation: A parameter is missing in the command line.

User Action: Reenter the command with the required parameter.

nn:nn:nn aaaTnn 36. •ERROR• Value out of range

Explanation: You entered a value that is either too large or too small.

User Action: Reenter the command with the correct value.

nn:nn:nn aaaTnn 38. •ERROR• No accounting data present

Explanation: You specified a terminal in a SHOW ACCOUNTING command that either does not have any accounting data or is not logged in.

User Action: Determine the correct terminal to use and reenter the command.

nn:nn:nn aaaTnn 39. •ERROR• Device specification error

Explanation: You entered a device specification incorrectly.

User Action: Determine the correct device and reenter the command.

nn:nn:nn aaaTnn 40. •FATAL• No startup transaction found

Explanation: The START/ACCOUNTING command and the SYSLOG task are closely related and depend on each other for correct operation. Any attempt to start the SYSLOG accounting task by any method other than the use of START/ACCOUNTING results in this message.

User Action: If START/ACCOUNTING and SYSLOG were used as intended and this error resulted, submit an SPR to DIGITAL.

nn:nn:nn aaaTnn 41. •WARN• Bad transaction length. Block(s) of secondary pool may be lost

Explanation: A transaction may be too large or too small. You may have designed the transaction incorrectly.

User Action: Correct the generation of the transaction. See B.MINL or B.MAXL in Section 10.6.1.4.

nn:nn:nn aaaTnn 42. •ERROR• UAB Listhead not zero on shutdown

Explanation: This message indicates an internal consistency error.

User Action: Submit an SPR to DIGITAL.

nn:nn:nn aaaTnn 43. •ERROR• Transactions follow SAB

Explanation: This message indicates an internal consistency error.

User Action: Submit an SPR to DIGITAL.

nn:nn:nn aaaTnn 44. •FATAL• UAB Not found in list

Explanation: This message indicates an internal consistency error.

User Action: Submit an SPR to DIGITAL.

nn:nn:nn aaaTnn 45. •FATAL• Exiting due to error

Explanation: The SYSLOG task exited for the specified reason. An additional message explaining the reason follows this message.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command. Then, refer to the description of the second message to determine the cause of the error. Correct the error before restarting Resource Accounting.

nn:nn:nn aaaTnn 46. •FATAL• Transaction not in secondary pool

Explanation: This message indicates an internal consistency error.

User Action: Submit an SPR to DIGITAL.

nn:nn:nn aaaTnn 47. Exiting

Explanation: This message indicates a normal SYSLOG exit due to a STOP/ACCOUNTING command.

User Action: No user action is requrired. This is an informational message.

nn:nn:nn aaaTnn 50. Accounting not active

Explanation: You entered a command that requires Resource Accounting to be running.

User Action: Enter the START/ACCOUNTING command and reenter the command.

nn:nn:nn aaaTnn 51. •ERROR• Accounting had a fatal error, must be cleaned up

Explanation: Resource Accounting failed.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command.

nn:nn:nn aaaTnn 52. •ERROR• SYSLOG Not active

Explanation: The SYSLOG task has exited unexpectedly.

User Action: Enter the STOP/ACCOUNTING CLEAN_UP command and restart Resource Accounting.

nn:nn:nn aaaTnn 53. •ERROR• Cannot write crash transaction

.
.
.

[message]

Explanation: The crash transaction could not be written to the new transaction file. The crash data is lost.

An additional message explaining the reason follows this message.

User Action: No user action is required.

nn:nn:nn aaaTnn 54. •ERROR• Error during scan file rename

Explanation: The specified error was encountered when dealing with the scan file.

User Action: Determine the cause of the problem and correct it.

nn:nn:nn aaaTnn 55. •ERROR• On scan file PUT$

Explanation: The specified error was encountered when dealing with the scan file.

User Action: Determine the cause of the problem and correct it.

nn:nn:nn aaaTnn 57. Enter reason for the crash (Maximum 60. characters)

Explanation: This is the prompt for the user-supplied crash reason because "yes" was answered as the CRASH_REASON parameter on the START/ACCOUNTING command.

User Action: Enter a crash reason.

nn:nn:nn aaaTnn 58. •ERROR• Terminal I/O Error

Explanation: An I/O error occurred from the terminal during the input of the crash information. The crash information is lost.

User Action: No user action is required.

nn:nn:nn aaaTnn 60. •ERROR• I/O Error

Explanation: The specified I/O error occurred during the SHOW ACCOUNTING TRANSACTION command.

User Action: Correct the I/O condition and try the command again.

nn:nn:nn aaaTnn 64. •ERROR• Illegal file name

Explanation: There is an incorrect file specification for the SHOW ACCOUNTING TRANSACTION command.

User Action: Correct the file specification and try the command again.

nn:nn:nn aaaTnn 65. •ERROR• Open error

Explanation: The specified error occurred during a file open procedure of the SHOW ACCOUNTING/TRANSACTION_FILE command.

User Action: Determine the cause of the error, correct it, and try the operation again.

nn:nn:nn aaaTnn 70. •ERROR• On scan file open

Explanation: The specified error was encountered when Resource Accounting was processing the scan file.

User Action: Determine the cause of the problem and correct it.

nn:nn:nn aaaTnn 71. •ERROR• On scan file delete

Explanation: The specified error was encountered when Resource Accounting was processing the scan file.

User Action: Determine the cause of the problem and correct it.

nn:nn:nn aaaTnn 72. •ERROR• On scan file close

Explanation: The specified error was encountered when Resource Accounting was processing the scan file.

User Action: Determine the cause of the problem and correct it.

nn:nn:nn aaaTnn 73. •ERROR• On scan file open

Explanation: The specified error was encountered when Resource Accounting was processing the scan file.

User Action: Determine the cause of the problem and correct it.

nn:nn:nn aaaTnn 74. •ERROR• Copying scan file to trans file

Explanation: The specified error was encountered when processing the scan file. The crash information that would have appeared in the transaction file is lost.

User Action: No user action is required.

nn:nn:nn aaaTnn 75. •ERROR• On scan file close

Explanation: The specified error was encountered when processing the scan file.

User Action: Determine the cause of the problem and correct it.

nn:nn:nn aaaTnn 76. •ERROR• On scan file delete

Explanation: The specified error was encountered when processing the scan file.

User Action: Determine the cause of the problem and correct it.

nn:nn:nn aaaTnn 79. •ERROR• Task not in memory or not active

Explanation: The task specified with the SHOW ACCOUNTING/INFORMATION command parameter TASK=taskname was not in memory or was not active. Therefore, there is no task accounting date for this task.

User Action: No user action is required.

Part III: System Problems and Performance

Chapter 11
Handling System Problems

This chapter introduces the basic tools and concepts that are necessary for understanding the problems you may encounter while managing your system In addition, it includes a brief discussion of the Diagnostics Loader task (DLD), which is available for multiprocessor systems only.

11.1 System Faults

System faults cause an RSX-11M-PLUS or Micro/RSX operating system to stop processing an application properly. A *fault* could be a user error, a system software failure, an application software error, or a hardware fault. You can recognize these types of errors as follows:

- *User errors* (for example, accidentally removing a disk from the system) are obvious because the fault usually occurs immediately after the user action.
- *System software failures* generally occur if a system problem is encountered while you are debugging the application software.
- *Application software errors* are application-specific and are not discussed in this manual.
- *Hardware faults* occur when a device malfunctions and causes errors.

The remainder of this chapter describes how to detect, isolate, and recover from hardware faults.

11.2 Hardware Faults

Hardware faults can be grouped in the following categories:

- Media failures
- Transient failures
- Intermittent failures
- Soft errors
- Hard errors

The following subsections describe each type of fault in detail.

11.2.1 Media Failures

Media failures occur when I/O operations fail because portions of the medium (a disk or tape) are bad. For both disks and tapes, when a media failure is detected, the system makes repeated attempts to perform read operations.

Before you initialize a disk, check the disk for faulty sectors with the Bad Block Locator Utility (BAD). BAD marks the bad sectors, which prevents the file system from using them. (See the *RSX-11M-PLUS Utilities Manual* for more information.)

Occasionally, sectors on the media become unusable, causing I/O operations to those sectors to fail. The Error Logging system can be used to detect these failures. (See the *RSX-11M-PLUS and Micro/RSX Error Logging Manual* for more information.) If your media contains critical data records, or if a large number of errors have occurred on them, you should probably replace the media.

On devices supported by the DU-device driver (DUDRV), bad blocks may be replaced by the Bad Block Replacement Control Task (RCT) or the Mass Storage Control Protocol (MSCP) controller. See Chapter 13 for more information on RCT.

11.2.2 Transient Failures

Transient failures cannot be repeated and are caused by electrical factors, such as static from clothing or carpeting. If the failure occurs on an I/O device, the I/O operation is retried automatically and the application software does not detect the failure.

If the failure occurs in memory or in a central processing unit (CPU), the system may fail or "crash" and may have to be rebootstrapped.

The Error Logger records transient failures. For more information, see the *RSX-11M-PLUS and Micro/RSX Error Logging Manual.*

11.2.3 Intermittent Failures

Intermittent failures are caused by irregular conditions within the system and usually happen during periods of high system or device activity. The failures occur when recorded data is too corrupted to be read. The data becomes corrupted when the device on which it is recorded is not working properly, usually because of a temporary power shortage.

An example of an intermittent failure is a disk device detecting a power failure, which causes the disk's volume valid bit to be reset. Dismounting and remounting the disk clears this condition and may allow I/O operations to proceed.

If the sources of intermittent failures are not obvious or directly reproducible, re-create the conditions that caused the failure by using the I/O Exerciser (IOX). IOX simulates high system activity, which could cause the failure to recur. (See Chapter 12 for information on IOX).

The Error Logger records intermittent failures that result in device errors.

11.2.4 Soft Errors

Soft errors are device errors that do not prevent an I/O operation from completing successfully. The device driver retries the operation and, depending on the capabilities of the device, executes the error correction code (ECC) or disk-head offset positioning. The driver may also attempt to perform multiple read requests of a smaller size.

Soft errors usually indicate poor media or a misaligned device. Disk heads that are not aligned properly cause disk errors, which can be corrected by head offset positioning.

If several soft errors occur on a device, locate the source of the errors and contact DIGITAL Field Service Maintenance.

The Error Logger records soft errors.

11.2.5 Hard Errors

Hard errors are solid device failures that prevent a device from working properly. The errors make the device unusable, but they are also the easiest to diagnose and correct.

The Error Logger records failed operations on disks and tapes.

11.3 Recoverable and Nonrecoverable Errors

Soft errors and many media errors are recoverable errors. They are recorded by the Error Logger but do not impact system operation.

Hard errors and some media errors are nonrecoverable errors. They usually result in recurring device errors and failure of the application software. (The application can survive one failure on each device type before it will stop processing completely.) However, the operating system itself often recovers from the failure because it requires only the system disk, the central processing unit (CPU), and memory to continue running.

11.4 Fault Isolation

Once the software is stable, the availability of a system depends on the reliability of individual devices and the time required to restore processing. RSX-11M-PLUS and Micro/RSX systems can, in some cases, work around equipment failures in individual devices; however, you must eventually isolate a faulty device, correct the reason for the failure, and restore the device to the system. Depending on the severity and location of the error, you may have to restart the application and the system.

Failures often exhibit symptoms that allow you to detect a failing device before it disrupts system operation. Because of this, you should regularly monitor the condition of all peripheral devices and their ability to respond correctly to I/O requests. Periodically, examine the Error Logging reports on all devices and run the I/O Exerciser (IOX) on infrequently used devices. The Error Logger gathers information concerning all types of errors as they occur and generates comprehensive error reports. IOX exercises combinations of devices under high I/O activity. (For more information on IOX, see Chapter 12. For information on the Error Logger, see the *RSX-11M-PLUS and Micro/RSX Error Logging Manual.*)

The Error Logger also provides error reports that contain a description of device fault conditions. Use the reports to determine whether the device should remain in operation. If the fault conditions are transient (see Section 11.2.2) and the device is capable of correctly performing its I/O functions, the device can remain in operation. However, you should closely monitor the device and schedule it for future maintenance as required.

11.5 System Recovery

The procedure you use to diagnose and recover from a system fault depends on the primary use of your system. If it is mainly used for application development, then use the Error Logger or IOX to determine the cause of a failure and to prevent future failures. If your system is used for process control and is running an application that requires enhanced availability, utilities such as the Executive Debugging Tool (XDT) can be used to isolate and correct failed components. (For more information on XDT, see the *RSX-11M-PLUS and Micro/RSX XDT Reference Manual.*)

If you are using an RSX-11M-PLUS system and you want to restart it and your application, first bring up the baseline system (that is, the minimum requirements for running the RSX-11M-PLUS operating system). Then, gradually bring devices on line until you have the minimum system required for running the application. If any of the devices are faulty, remove them from the system and bring others on line until you have the required configuration.

The method for recovering from system faults also depends on the type of fault. For example:

- Transient errors could cause the system to fail, which would require bootstrapping the system again.
- Hard errors prevent you from using the device. The device may need to be reconfigured out of the system so that diagnostics can be run for it.
- Intermittent errors do not have an obvious source. You may need to begin with a minimum configuration and then gradually add devices until you find the one that caused the error.

  If the minimum configuration still has the problem, switch to a different configuration. Another possible way to recover from intermittent failures is to turn the error into a hard error by using IOX, which can impose an appropriate system load. By re-creating the error, you can discover its source; then, you can correct the error.

The following sections describe how to recover from different kinds of hardware failures.

11.5.1 Device Unit and Controller Failures

Most failures in an operating system result from disk, tape, or other such device failures. Either the units themselves fail or the respective controller fails.

The time needed to reconfigure around these failures depends entirely on the application. If the application can notify an operator of the failure, thus allowing the operator to remove and inspect the media, mount it on another drive, and reintegrate it into the system; then, the time to reconfigure is short. If, on the other hand, the operator must find a way to halt the application, to move the media, and to restart the system, then the time to reconfigure increases.

The Task Termination Notification program (TKTN) prints a message at the console terminal whenever an nonrecoverable hardware error occurs. For example:

```
*** DB1: -- Nonrecoverable hardware device error
```

These messages can alert an operator to an impending failure and allow time to issue the commands to remove the failing unit from the system.

11.5.2 Processor Failures

Although it occurs rarely in comparison with device unit and controller failures, a processor failure also halts system operation. If a processor fails, the system usually exhibits one of the following symptoms:

- The system traps to XDT or fails.
- The sanity timer expires for one or more processors (multiprocessor systems only).
- The console lights on nonremote diagnostic consoles show that the system is halted in the Executive.
- The system halts with no printouts or visual indicators.

Note

> On multiprocessor systems, a crash, odd address trap, or other failure may occur on a processor other than the faulty one.

On multiprocessor systems, regardless of the cause of a processor failure, the recovery technique is straightforward and direct: you must isolate and exclude the disabled processor and reboot the system. The memory configuration need not change unless you wish to perform standalone diagnostics (see Section 11.6) on the failed processor.

11.5.3 Memory Failures

The two most common causes of memory failures are uncorrectable double-bit errors (parity errors) and loss of memory power.

If a user-state task encounters an uncorrectable memory parity error, the system aborts the task and locks that portion of memory. The Fixer task (FXR) detects the parity errors and creates a distinct segment in the task's region by using the memory in which the errors were detected. This segment is separate from the rest of the region and is not used by other tasks. FXR is completely transparent to both users and tasks.

If the Executive or a system-state task encounters an uncorrectable parity error, the system prints the following message and then halts:

```
***EXEC PARITY ERROR STOP
```

Also, the sanity timers on multiprocessing systems may expire.

On RSX–11M–PLUS multiprocessor operating systems with MK11 memory, regardless of system state (user or kernel), a double-bit error causes the MK11 memory box hardware to light the UNCOR ERROR light or lights on the respective memory box port controllers. The lights indicate that an uncorrectable error has halted the system.

Before rebooting the system, you must manually exclude the MK11 memory box, as follows:

1. Switch all ports of the faulty memory box off line. If necessary, use the *thumbwheel switches* to properly address the remaining memory box or boxes. (The *thumbwheel switches* are used when the failed box is box 0, which is the boot box, or when you want to disable a memory box by changing the starting addresses of the remaining boxes.)

2. Halt the processor, initialize it, and rebootstrap the system.

When loss of power causes the memory failure, the system halts. The system exhibits one or more of the following symptoms:

- No lights or indicators are lit. This usually indicates that the outside power supply has failed.
- No MEM POWER READY lights are lit on the memory box. This indicates that the memory box's power supply has failed or has been powered down. Check the power switch on the Port A controller to the box.
- No MEM POWER READY lights are lit on one port of all memory boxes. This indicates that the central processing unit's (CPU's) power supply has failed or that the CPU has been powered down. Note that the power supply failure could have come through an expander box or through the far side of a switched bus currently switched to the CPU.

In all of the above cases, if a CPU or a memory box has been switched off but then switched back on before the memory batteries discharge, the system usually recovers without rebootstrapping. Otherwise, follow the same two steps used when an uncorrectable error halts the system.

11.6 Standalone Maintenance for Multiprocessor Systems

On multiprocessor systems, when a device fails, it is often necessary to run diagnostic tasks before servicing the device. However, the diagnostic tasks require a processor for execution. Thus, to service most devices, you must remove a processor from the system.

The following list gives the requirements for testing and servicing peripheral devices:

- A processor must be available to execute the diagnostics for the peripheral.
- Memory must be available for the diagnostics.
- A mechanism to load the diagnostic task into memory must be available.

The processor that executes the diagnostics must be connected to the UNIBUS on which the device to be tested resides. This implies that other peripherals on that bus run will not be available to the remainder of the system. Therefore, when determining the initial system configuration, you must ensure that duplicate devices are located on distinct bus runs. If the device to be tested is on a switched bus run, the switched bus can be connected to any processor to which it has access.

To make memory available for diagnostic tasks, the usual procedure is to logically remove a memory box from the running system and to use the box for diagnostics only. This memory box is then switched manually so that the processor to be used for diagnostics views the box with a starting address of 0.

For loading a diagnostic into memory, the RSX–11M–PLUS multiprocessor system provides the Diagnostic Loader task (DLD). DLD loads a diagnostic program into a partition in the memory box of the diagnostic processor (the processor to which the failed device is attached). Diagnostic programs are system image files that you use to test the failed devices. The partition is created and specified by you and is mapped by Program Logical Address Space (PLAS) directives. By default, DLD clears the partition to the end after loading the program. The diagnostic is then run stand alone on the failed processor.

The steps for using DLD are as follows:

1. Use the File Transfer Utility program (FLX) to transfer the diagnostic program from the diagnostic tape or disk to a Files–11 device attached to the diagnostic processor. (See the *RSX-11M-PLUS Utilities Manual* for information on FLX.) Because the tape or disk is in DOS–11 format, use the FLX switch Image Mode (/IM) to convert the file (the diagnostic program) to Files–11 format. The file is in formatted binary format and has a file type of BIC or BIN. (Note that binary formatted files cannot be patched.)

2. Reconfigure your system so that you have a system that is logically connected to two CPUs; each CPU should have at least one memory box (each box has 256K words of memory). An example of a typical configuration is as follows:

   ```
   BOX 0    PORT A  ONLINE, START=000
            PORT B

   BOX 1    PORT A  OFFLINE
            PORT B  ONLINE, START=000

   IIST     Off line both sides.
   ```

3. Bootstrap RSX–11M–PLUS on CPA.

4. Set BOX 0 PORT A to ONLINE, START=010

5. Bring Box 1 on line. This memory box will contain the diagnostic partition. For example:

   ```
   CON>
   ```

6. Use the CON DISPLAY command to determine the base address and size of the memory box. For example:

   ```
   CON>
   ```

   The base address of the partition will be the base address of the diagnostic program.

7. Create the diagnostic partition in the memory box by using the command SET /PAR and by specifying the type DIAG. This partition will be devoted to diagnostic functions and should begin at the location in memory that the diagnostic processor will see as location 0 or the bottom of memory. For example:

   ```
   >
   ```

   This command creates a partition named JIMK with a base address of 2000000_8, which consists of 1777_{10} 32-word blocks. The other processors see the memory box used for diagnostics as starting at location 2000000_8, but the processor running the diagnostic will see the box as starting at location 0.

   Other partition types used with the SET /PAR command ensure that the requested memory is within the system image area (less than $SYSIZ), but DIAG ensures only that the memory physically exists. For the other processor to run concurrently with the diagnostic processor, the partition must be larger than $SYSIZ.

8. Use the DLD task to convert the binary formatted file into a bootable system image file (patchable with the Task File Patch Program called ZAP; see the *RSX-11M-PLUS Utilities Manual*) and then to load it into the partition. The DLD command line has the following format:

   ```
   DLD>[partition][/HE]=filename[.typ][/qualifier]
   ```

partition

Specifies the partition created in the diagnostic processor's memory box into which the diagnostic program will be loaded. Unless you specify the /NL qualifier, you must specify a partition name. Otherwise, DLD displays a "Partition not found" error message. If you specify both a partition name and the /NL qualifier, the qualifier overrides the name.

/HE

Provides a brief description of DLD's use and format.

typ

Specifies the file type for the diagnostic program. The file type can be BIC, BIN, or SYS.

If you specify the file type as BIC or BIN, DLD converts the file into a system image file. Also, if DLD cannot find the file, it searches for a file with the SYS file type. If you do not specify any file type, DLD searches for a file with the BIC file type and then for one with the BIN file type.

If you specify the file type as SYS, DLD does not have to convert the file; it is ready to be loaded into the memory box.

/qualifier

Specify one of the following qualifiers:

/NL Inhibits loading of the diagnostic program (DLD creates the system image file only).

/NC Inhibits clearing (zeroing) of any remaining memory in the partition after the diagnostic program has been loaded. The default is to clear the remaining memory.

/OFF:n Offsets the starting location of the load by n times 32_{10} words. The area between the beginning of the partition and the beginning of the diagnostic is unaltered.

/LI Lists information about the partition and the file: the partition name (if you have specified one), the base and size of the partition, and the new name of the system image file (if it is converted).

Specify this qualifier to display the transfer address of the diagnostic program (usually 200_8 or 1000_8). You will need the transfer address to initiate the diagnostic.

/LW Accounts for the DOS–11 link word that may exist in bytes 0 and 1 of every input block.

The following example converts the diagnostic file TEST.BIN (in binary format) into the system image file TEST.SYS:

```
DLD>
PARTITION NAME: JIMK
PARTITION BASE: 20000    SIZE: 1777 (32. WORD BLOCKS)
FILE NAME: TEST.SYS;1
TRANSFER ADDRESS: 200
```

In this example, DLD lists the transfer address of the file and other information about the file and the partition.

9. After you enter the DLD command line, DLD loads the program into a region that it creates in the diagnostic partition. The region is called DIAG and is the same size as the partition. After the program is loaded, DLD clears any remaining memory from the partition (unless otherwise specified).

10. Halt the processor (in this case, CPB) and start the diagnostic at its transfer address. Push the START button to begin the diagnostics.

11. Use the following DIGITAL Command Language (DCL) or Monitor Console Routine (MCR) command line to remove the diagnostic partition from the memory box:

    ```
    DCL>
    ```

    ```
    MCR>
    ```

Chapter 12
The I/O Exerciser

The I/O Exerciser (IOX) detects I/O problems on the disk, terminal, and tape units in your hardware configuration. IOX tests the hardware (and accompanying software) by performing repeated operations to the same unit. This is commonly referred to as "exercising" the device.

IOX exercises devices on two kinds of volumes: non-file-structured (NFS) and file-structured (Files–11). In the context of IOX, the meaning of these terms differs from other manuals within the RSX–11M–PLUS and Micro/RSX documentation sets. They are defined here as follows:

- All tapes and terminals are NFS volumes. NFS describes devices and the testing of devices that contain non-file-structured volumes.
- Disks can be either NFS or Files–11 volumes.
- Files–11 volumes are disks that you initialize with the Monitor Console Routine (MCR) command INITIALIZE. They have a home block and a Files–11 structure.

You can use IOX to test Files–11 disks, NFS disks, magnetic tapes, DECtapes, and cassettes. Using the IOX Command Language, you can execute, control, and monitor I/O exercises for all the device units in your system.

This chapter describes how to use IOX. It also describes each IOX command in alphabetical order and provides examples on how to set exercising parameters.

12.1 Introduction to IOX

Before you run an I/O exercise, you need to consider the following aspects of testing:

- The type of devices you want to exercise
- The type of exercises you want to run for each device
- The parameters you want to set for each exercise

To show how these aspects affect the way you use IOX, this section provides an overview of the procedures IOX uses to exercise a device.

Using IOX, you can perform an exercise operation on any mounted disk volume, magnetic tape unit, or terminal in your hardware configuration (as long as it has not been allocated to another user). By default, IOX exercises up to 28 units simultaneously. However, by modifying certain parameters, you can use IOX to test as many as 250 units simultaneously.

Note

> To change the initial default values for IOX parameters as well as other default values for the IOX task, IOX.TSK, edit the IOX task-build file IOXBLD.BLD. On RSX–11M–PLUS systems, this file is located on your system disk in directory [1,24].
>
> Pregenerated RSX–11M–PLUS (RL02 distribution kits) and Micro/RSX operating systems do not allow you to modify the default values for IOX; thus, they do not include the file IOXBLD.BLD.

When IOX exercises a device, it follows a general procedure consisting of four basic steps:

1. Writing bytes of data (made up of a specific data pattern) to a mounted volume
2. Reading the data just written
3. Checking that the data read matches the data written
4. Reporting mismatches of data

IOX repeats this process until the exercise has completed.

More specifically, however, IOX uses procedures that depend upon the type of device being exercised (disk, magnetic tape, cassette, DECtape, or terminal). The following list provides a more detailed description of how IOX exercises each device type:

Disks

IOX writes a buffer of data to a disk and then reads that data from the disk. By setting exercise parameters, you determine the data pattern to be written, the range of blocks to be tested, and the access method to be used (sequential or random).

Magnetic Tapes and Cassettes

To exercise magnetic tapes and cassettes, IOX uses a procedure similar to the one used for disks. IOX rewinds the tape, writes one record, backspaces one record, and then reads the record that was just written. This procedure is repeated until the number of records that you specified for testing are read. At the end of the exercise, IOX writes two tape marks and rewinds the tape. If you test more than one unit, IOX continues testing on those units while the tape is rewinding.

By setting exercise parameters, you determine the data pattern to be written, the size of the data buffer, and the number of records to be exercised.

Terminals

IOX exercises terminals by continuously writing out a portion of the American Standard Code for Information Interchange (ASCII) character set to a terminal that is not logged in. If you specify the parameter LOOPBACK NO (that is, if you accept the default), IOX does not perform any additional functions. However, if you specify LOOPBACK, IOX reads the characters back (after writing them), compares the two sets of characters, and reports on the comparison between the two sets.

Note

> LOOPBACK mode requires that a loopback connector be connected in place of the terminal to direct the characters written back to IOX. IOX depends on the presence of a type-ahead buffer to store the characters between writing and reading them.

For more information on the LOOPBACK parameter command, see Section 12.6.16.

DECtapes

IOX exercises DECtapes by writing forward and by reading forward, and then by writing reverse and by reading reverse until the end of the tape is reached. IOX exercises DECtapes by using a data buffer of two blocks (1024_{10} bytes).

By setting a test parameter, you determine the data pattern that IOX writes to the DECtape.

12.2 Running an I/O Exercise

There are four steps involved in running an I/O exercise:

1. Invoking IOX
2. Including the device or devices to be tested in your test configuration
3. Selecting the device units for exercising (using the SELECT, VERIFY, or FILES11 command)
4. Issuing IOX commands

Using IOX commands, you can start and stop the I/O exercise, set and modify the exercise parameters, and monitor IOX reports. The sections that follow provide detailed descriptions of each procedure. Section 12.6 provides a description of all the IOX commands.

12.2.1 Invoking IOX

IOX is task built as a nonprivileged, checkpointable task that runs at a priority of 50 in the GEN partition. If IOX is not installed, you can invoke it by entering the DIGITAL Command Language (DCL) or MCR command RUN. You can also install the IOX task by using the DCL or MCR command INSTALL; then, you can invoke IOX by typing the task name (IOX, by default).

You can invoke IOX from a nonprivileged or a privileged terminal, but you must be at a privileged terminal to install it.

Format 1: Invoking an Uninstalled Version of IOX

To invoke an uninstalled version of IOX, enter the MCR or DCL command RUN, as follows:

```
>
```

If you invoke IOX with the RUN command, IOX uses your terminal number to name the task and prompts you with this task name. For example, if you type the RUN command at terminal TT10, IOX displays the following prompt:

```
TT10>
```

You can enter IOX commands at the prompt.

Format 2: Invoking an Installed Version of IOX

To install IOX, enter the MCR or DCL command INSTALL from a privileged terminal, as follows:

```
>
```

After IOX is installed, invoke it with the following command:

```
>
```

IOX responds by prompting you with the name of the task that you assigned to IOX when you installed it. If you did not specify a task name when you installed IOX, its task name is ...IOX and the prompt is as follows:

```
IOX>
```

You can enter IOX commands at the prompt.

12.2.2 Setting Up a Test Configuration—The CONFIGURE Command

Before setting up your test configuration, allocate and mount all units that you want to exercise. To mount volumes that are not in Files–11 format, use the MOUNT command with the /FOREIGN switch. (For information on the MCR commands ALL and MOU, see the *RSX–11M–PLUS MCR Operations Manual.* For information on the DCL commands ALLOCATE and MOUNT, see the *RSX–11M–PLUS Command Language Manual* or the *Micro/RSX User's Guide, Volume* 2.)

After you allocate and mount the units, include them in your test configuration. When you include a unit in your test configuration, IOX adds the unit to its list of units available for exercising and creates an empty data structure for the unit.

In the default test configuration, IOX includes one of each device type supported by DIGITAL (as unit number 0). To add units to the test configuration, use the CONFIGURE command. To display the units that are included in the current test configuration, enter the CONFIGURE command without specifying any of the parameters.

Format

```
CONFIGURE [ddnn:[=type]]
```

Parameters

dd

Specifies the device mnemonic (for example, DU).

nn

Specifies the device unit number.

type

Specifies the device type: DISK, TERM (for terminal), or TAPE.

You can also assign logical device names to units in the test configuration with the DCL command ASSIGN (MCR command ASN). Logical names make it possible for you to direct multiple IOX exercises to the same device. (For more information on the ASSIGN command and logical assignments, see the *RSX-11M-PLUS Command Language Manual* or the *Micro/RSX User's Guide, Volume 2*. For information on the ASN command, see the *RSX-11M-PLUS MCR Operations Manual*.)

Examples

```
IOX>

Disks:
DB0   DD0   DF0   DK0   DL0   DM0   DP0   DR0   DS0   DU0
DX0   DY0   EM0   SY0

Tapes:
CT0   DT0   MM0   MS0   MT0   MU0

Terminals:
TT0
```

Displays the current test configuration.

```
IOX>
```

Adds DK1 to the configuration as a disk unit (because DK0 is already configured as a disk) and MM1 to the configuration as a tape unit (because MM0 is already configured as a magnetic tape).

```
IOX>
```

Adds TT12 to the configuration as a terminal unit (because TT0 is already configured as a terminal.)

12.2.3 Selecting Devices for Exercising

After configuring all the units you want to exercise, select the units for exercising. When you select a unit, IOX fills in the data structure created when you configured the unit. This data structure contains all the information IOX needs to exercise the specified unit. To select units for exercising, use the FILES11 command for Files–11 structured volumes, and use the SELECT command or the VERIFY command for non-file-structured (NFS) volumes. Briefly, these commands are described as follows:

- The FILES11 command selects a disk unit with a mounted Files–11 volume for an I/O exercise. Exercising with FILES11 does not affect the contents of the disk being tested; your data is preserved.
- The SELECT command selects a unit with an NFS volume for an I/O exercise. Exercising with the SELECT command destroys the contents of the volume being exercised.

  The SELECT command is also used to select a terminal for exercising.
- The VERIFY command selects a disk unit with a mounted NFS volume for an I/O exercise. Exercising with VERIFY does not affect the contents of the volume because VERIFY only reads from the volume; it does not write to the volume being exercised. Your data is preserved.

Before exercising a unit, IOX checks the home block to see if it is file-structured. If you try to select a Files–11 volume with the SELECT or VERIFY commands, IOX returns an error message.

Note

> If you specify the IOX command VOLUMECHECK=NO parameter, you can prevent IOX from checking the file structure of the volume. Use the VOLUMECHECK command only if you are selecting a Files–11 disk that you no longer want to use as Files–11.

Section 12.6 contains information on the use and syntax of these commands.

12.2.4 Using the IOX Command Language

The IOX Command Language has the following four categories of commands:

Function commands — Select devices for testing, deselect devices from testing, and initiate tasks to run in parallel with IOX.

Control commands — Start and stop IOX processing, switch IOX modes of operation, and exit from IOX to the operating system monitor.

Display commands — Display current default parameters, get help text for IOX commands, and display status and activity reports during an I/O exercise.

Parameter commands — Set default parameters that affect how IOX tests the devices you select for testing.

Table 12–1 summarizes the IOX commands by category.

Table 12–1: IOX Command Summary

Command	Description
	Function Commands
DESELECT	Removes a device from the list of devices to be tested.
EXECUTE	Directs a command line to the operating system for execution.
FILES11	Selects a device with a mounted Files–11 volume for an I/O exercise that preserves the contents and structure of the volume.
SELECT	Selects a unit with a mounted NFS scratch volume for an I/O exercise that destroys the contents of the volume.
VERIFY	Selects a mounted disk for an NFS exercise that reads buffers of data without writing on the volume or performing data comparisons. The contents of the volume are preserved.
	Control Commands
ABORT	Terminates execution of an I/O exercise.
CONTROL_C	Allows you to enter IOX commands while an exercise is in progress.
EXIT	Terminates IOX processing and exits from IOX.

Table 12–1 (Cont.): IOX Command Summary

Command	Description
	Control Commands
PROCEED	Exits Interactive Mode, enters Execution Mode, and checks for IOX activity.
RESTART	Reinvokes IOX without exiting to the operating system.
START	Starts exercising units that you have selected for testing.
	Display Commands
BADBLOCKS	Enters bad blocks into the bad blocks list and omits them from testing.
CONFIGURE	Displays the current configuration.
DENSITY	Displays the current density setting on a magnetic tape.
HELP	Displays a summary of IOX commands at your terminal.
PARAMETERLIST	Displays current default parameters, IOX buffer space statistics, and device-dependent parameters for devices you have selected for testing.
PATTERN	Displays the pattern that IOX writes and reads during an exercise.
PRINTSUMMARY	Displays summary reports that summarize IOX activity between the last interval report and the time you enter the command.
RANGE	Displays the minimum and maximum block numbers for NFS testing on the specified disk.
SPY	Displays status information on devices being tested.
	Parameter Commands
BADBLOCKS	Enters bad blocks into the bad blocks list and omits them from testing.
BUFFERSIZE	Sets the amount of data that IOX transfers for every read or write request when testing NFS disks, magnetic tapes, and cassettes.
COMPAREDATA	Enables or disables checking of the data pattern.
CONFIGURE	Adds devices to your test configuration.
CONTROL_C	Enables or disables the use of CTRL/C for interrupting an I/O exercise.
DENSITY	Sets the density and characteristic word of the specified magnetic tape or displays the current density and characteristic word for the specified magnetic tape unit.

Table 12-1 (Cont.): IOX Command Summary

Command	Description
	Parameter Commands
ERRORLIMIT	Sets the maximum number of errors that IOX tolerates between interval reports before terminating testing on a device.
INTERLEAVE	Sets the number of blocks to be skipped before IOX writes the next buffer of data for disk testing (applies only to sequential testing, which you determine with the RANDOM command).
LOGFILE	Determines whether IOX directs activity and error reports to your terminal or to a log file.
LOOPBACK	Writes ASCII characters to a logged out terminal and then reads them back.
PATTERN	Sets the pattern that IOX writes and reads during an exercise.
RANDOM	Directs IOX to select either random or sequential blocks for disk testing.
RANGE	Sets the minimum and maximum block numbers for NFS testing on the specified disk.
RECORDS	Specifies the number of records to be exercised on magnetic tapes and cassettes.
REPORTERRORS	Enables or disables error reports.
RETRIES	Determines (for magnetic tapes and NFS disks) whether an I/O driver repeats an I/O operation after the driver receives errors during the operation.
RUNTIME	Sets the length of time (in minutes) that IOX exercises the unit or units you have selected for testing.
SUMMARYTIME	Determines how often (in minutes) IOX will output interval reports.
TEMPORARYFILE	Sets the size of the temporary file that IOX uses for testing Files-11 disks with the FILES11 command.
VOLUMECHECK	Enables or disables checking of disks, which are mounted as NFS volumes, for a home block and Files-11 structure.
WAIT	Determines whether or not IOX waits for an event flag while an exercise is in progress.
WRITECHECK	Determines whether or not I/O drivers check write requests to NFS disks.

For a detailed description of each command, see Section 12.6.

12.3 IOX Operating Modes

IOX operates in three modes: Command, Execution, and Interactive. Some of the IOX commands execute in only one of the three modes. This section discusses the functions of these modes, how you enter and exit each mode, and what IOX commands are valid in each mode.

12.3.1 Command Mode

IOX is in Command Mode after you have invoked it and have received the first IOX prompt. In Command Mode, no devices are being exercised, and tasks initiated by the EXECUTE command are queued but are not active. (For more information on the EXECUTE command, see Section 12.6.10.) Use Command Mode to set parameters, to select devices for testing, and to initiate tasks to be executed.

When you press the RETURN key, you receive an IOX prompt. When you press CTRL/C, you exit to MCR for one command line.

12.3.2 Execution Mode

When you begin an I/O exercise by entering the START command, IOX exits Command Mode and enters Execution Mode. In Execution Mode, the following events occur:

- Exercises begin on selected devices
- Tasks initiated with the EXECUTE command begin executing
- The IOX clock is turned on
- Activity reporting begins (See Section 12.5.1)
- Error reporting begins (See Section 12.5.2)

Set the CONTROL_C parameter in Command Mode to allow you to enter IOX commands while an exercise is in progress. By default, CTRL/C is enabled (CONTROL_C YES is set).

If you disable CTRL/C (that is, if you set CONTROL_C NO), then pressing CTRL/C in Execution Mode gives control to MCR. You receive the MCR prompt (>) every time you press the RETURN key and the explicit prompt (MCR>) every time you press CTRL/C. This remains in effect until the I/O exercise ends. With CTRL/C disabled, you cannot interrupt the exercise, but you can terminate it with the MCR command ABORT. (For more information on the ABORT command, see the *RSX-11M-PLUS MCR Operations Manual.*)

If you enable CTRL/C (that is, if you set CONTROL_C YES), IOX does not accept any input from your terminal except CTRL/C. If you try to enter something else, you receive the following error message for each key you press:

```
IOX -- Only ^C Allowed
```

When you press CTRL/C, you receive the IOX prompt (instead of the MCR prompt) and IOX exits Execution Mode and enters Interactive Mode.

(For more information on CONTROL_C, see Section 12.6.6.)

12.3.3 Interactive Mode

You have access to Interactive Mode only if CONTROL_C is enabled. When you press CTRL/C in Execution Mode, IOX exits Execution Mode and enters Interactive Mode. In Interactive Mode, the RETURN key and CTRL/C have the same functions that they have in Command Mode.

In Interactive Mode, the exercise and tasks initiated by the EXECUTE command are running. Activity reports and error reports directed to your terminal are suspended if you have set LOGFILE=NO. If you have set LOGFILE=YES, activity and error reports continue as in Execution Mode.

The following IOX commands are *not* valid in Interactive Mode:

- BUFFERSIZE

  You can use the BUFFERSIZE command within the SELECT and VERIFY command lines to override the current defaults for the device being selected, but the BUFFERSIZE command cannot be used by itself to reset the default.

- PATTERN

  Although you cannot use the PATTERN command to choose a different pattern to be written to the volumes being tested, you can use PATTERN with no argument to display all the patterns.

- RUNTIME
- START
- RESTART

If you enter one of these commands, IOX prints the following error message:

```
IOX -- Command valid only in command mode
```

The following IOX commands are valid *only* in Interactive Mode:

- ABORT

  When you use the ABORT command, IOX exits Interactive Mode and enters Command Mode. The ABORT command with no qualifiers terminates all exercising and all tasks initiated by the EXECUTE command. This form of ABORT is valid only in Interactive Mode because in Command Mode there can be no exercise or active tasks to abort.

- PRINTSUMMARY
- PROCEED

  The PROCEED command exits Interactive Mode and enters Execution Mode unless no activity remains when you issue PROCEED, in which case IOX aborts the exercise.

- SPY

Although you can use PRINTSUMMARY and SPY in Command Mode without generating an error message, these commands display meaningful information only when you use them in Interactive Mode.

12.4 Using Indirect Command Files with IOX

You can use indirect command files as input to IOX. IOX accepts one level of indirect command files. You can use indirect command files to set up your test configuration, to choose devices for testing, to set exercise parameters, and to start and exit the exercise. The default file type is CMD.

For example, use an editor to create a file named TEST.CMD that contains the following IOX command sequence:

```
CONFIGURE DB2:   ; Adds DB2 to the test configuration.

FILES11 DB2:     ; Selects DB2 for an exercise that preserves
                 ; the contents and structure of the disk.

RUNTIME 10       ; Sets an exercise parameter that runs
                 ; for 10 minutes.

START            ; Starts the exercise.

EXIT             ; Exits from IOX to the operating system
                 ; monitor when the exercise completes.
```

After you invoke IOX, execute the command file by typing an at sign (@) followed by the file name. For example:

```
IOX>
```

Note that you cannot interrupt an I/O exercise to enter commands; you must execute the command file before you start an exercise.

In this example, TEST.CMD instructs IOX to exercise DB2 (preserving its contents and structure) for 10 minutes and then to exit to the operating system monitor.

12.5 IOX Output

IOX generates two types of output: activity reports and error reports. *Activity reports* summarize IOX processing information, such as how long an exercise runs and how many asynchronous system traps (ASTs) are executed. There are two kinds of error reports: data compare error reports and I/O error reports. *Data compare reports* show where IOX finds errors on the volume being tested; *I/O reports* indicate failed QIOs from a driver.

The LOGFILE command directs activity and error reports to your terminal or to a log file. The REPORTERRORS command enables or disables error reports. (For more information on the LOGFILE and REPORTERRORS commands, see Section 12.6.15 and 12.6.24.)

12.5.1 Activity Reports

There are three types of activity reports, as follows:

Accumulated Totals — Summarize the activity for the entire exercise.

Interval Reports — Summarize IOX activity in the time intervals set by the SUMMARYTIME command.

Summary Reports — Summarize activity between the last Interval Report and the time you enter the PRINTSUMMARY command, which requests a summary report.

Sections 12.6.31 and 12.6.19 describe the SUMMARYTIME and PRINTSUMMARY commands, respectively.

The Format of Activity Reports

With the exception of the first line, the formats for the different types of activity reports are identical. Interval and summary reports begin with the time and date of the report.

For example:

```
IOX -- Summary of exerciser activity at 20-JUL-87 11:16:26
```

The accumulated totals report does not give the time or date of the report. Instead, it begins as follows:

```
A C C U M U L A T E D   T O T A L S :
- - - - - - - - - - -   - - - - - - -
```

The Content of Activity Reports

The content of all activity reports is identical. The following is an interval report of an exercise that tests the device SY:.

```
IOX -- Summary of exerciser activity at 31-DEC-87 11:18:29
TIME -- Run: 1 min.    Elapsed: 2 min.    Remaining: 3 min.
SY0: -- 1,874. R/W requests totaling 1,874. blocks with no errors
ASTs executed: 1,875.
Idle-loop iterations: 1,803.
```

The activity report contains the following information:

- Timing information
  - Time between reports as set in the SUMMARYTIME command
  - Time that the exercise has been in progress to the nearest minute
  - Time remaining to the nearest minute before the exercise will finish
- Unit information for each unit being tested
  - Number of I/O requests to the unit
  - Number of blocks exercised for disks and tapes
  - Number of bytes exercised for terminals
  - Number of error encounters on the unit

- Number of ASTs executed
- Number of iterations through the idle loop

Use the following commands to control the output of activity reports:

LOGFILE — Directs activity and error reports either to your terminal or to a log file in your default disk area. (See Section 12.6.15.)

PRINTSUMMARY — Generates reports that summarize IOX activity between the last interval report and the time at which you enter the command. (See Section 12.6.19.)

SPY — Displays information on the device or devices being tested. (See Section 12.6.29.)

SUMMARYTIME — Specifies the frequency at which IOX generates interval reports. (See Section 12.6.31.)

12.5.2 Error Reports

There are two kinds of error reports: *data compare error reports* and *I/O error reports.*

Data Compare Error Reports

A *data compare error* indicates that the driver returned a success I/O code to IOX but that the data written to the volume and the data read from the volume did not match. The following is an error report generated by a data compare error:

```
IOX -- DR1: data compare error at block 33,937(decimal) 102221(octal) ❶
Good Data: 005767   Bad Data: 155555    ❷
Word Position: 0 (decimal) 0 (octal)    ❸
Buffer contained 255. additional errors ❹
```

❶ Identifies the device on which IOX encountered an error. It reports the starting block (in both decimal and octal) in which the error was found.

❷ Shows the valid ("good") data that IOX wrote to the volume and the invalid ("bad") data that IOX read from the volume. Only the first error encountered in the test buffer is reported.

❸ Shows the position of the first word in the test buffer that generated a mismatch during the current operation. Only the first error encountered in the test buffer is reported.

❹ Shows how many other errors were found in the test buffer.

I/O Error Reports

If the Executive directive Queue I/O (QIO$) fails, the driver returns an I/O failure code to IOX. IOX generates a specific *I/O error report* as a result of the following I/O errors:

I/O Code	Description
IE.ABO	Operation aborted
IE.BBE	Bad block
IE.DAO	Data overrun

I/O Code	Description
IE.DNR	Device not ready
IE.EOT	End-of-tape detected
IE.FHE	Fatal hardware error
IE.OFL	Device off line
IE.PRI	Privilege violation
IE.VER	Unrecoverable error
IE.WCK	Write-check error
IE.WLK	Write-lock error

These error codes are shown in the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual.* Section 12.8 of this chapter presents the format and content of the error messages that IOX generates because of these error codes.

12.6 IOX Command Descriptions

This section provides complete descriptions of the commands in the IOX Command Language. The commands are presented alphabetically, and each command description includes the following information:

- Type of command: Function, Display, Parameter, and Control
- Command format or formats
- Descriptions of one or more qualifiers
- Operating mode restrictions. (If no mode restrictions are described, the command is valid in Command Mode and in Interactive Mode.)
- Types of devices and testing to which the command applies.
- Initial defaults, if applicable.

The IOX Command Language observes the following conventions:

- Commands are not case sensitive. You can enter the commands in uppercase, lowercase, or both. In the text, all commands are represented by uppercase letters.
- You can abbreviate commands by using any combination of characters that appear unique to the IOX Command Language. In the text, all commands are presented in their unabbreviated form.
- In a command line, lowercase words and letters indicate that you substitute a value.
- All numerical qualifiers are decimal.
- Square brackets ([ ]) indicate optional elements in the command line.

12.6.1 ABORT

The ABORT command is an IOX control command. It terminates the execution of tasks initiated with the IOX command EXECUTE.

There are three ABORT command formats; each format performs the same basic function. However, the format you use determines which tasks are aborted.

Note

> You cannot use the ABORT command to abort a task that is attached to your terminal, such as the Peripheral Interchange Program (PIP).

Format 1: Terminating IOX and Aborting All Tasks

```
ABORT
```

Format 1 terminates the execution of an I/O exercise and tasks initiated by the EXECUTE command. This format is valid only in Interactive Mode (testing in progress) because in Command Mode (no testing in progress) there can be no running exercise or active tasks to abort.

Format 2: Aborting All Active Tasks

```
ABORT/ALL
```

In Interactive Mode, Format 2 terminates the execution of all tasks initiated by the EXECUTE command and removes the tasks from the list of tasks to be executed. In Command Mode, this format removes all tasks from the list of tasks to be executed.

Format 3: Aborting a Specified Task

```
ABORT taskname
```

In Interactive Mode, Format 3 terminates the execution of a task initiated by the EXECUTE command and removes the task from the list of tasks to be executed. In Command Mode, this form of ABORT removes the task from the list of tasks to be executed.

The taskname parameter specifies the name of the task to be aborted.

12.6.2 BADBLOCKS

The BADBLOCKS command is a device-dependent parameter command and a display command. It applies only to non-file-structured (NFS) disk testing. To use the BADBLOCKS command, specify a disk that has already been selected for NFS operations.

There are two formats for using BADBLOCKS as a parameter command. Both formats enter bad blocks into the bad blocks list. After you enter the bad blocks for your disk into the list, the block numbers remain on the list until you deselect the disk.

Format 1: Specifying Individual Blocks

```
BADBLOCKS ddnn:blocknumber[,blocknumber1][,blocknumber2][,...]
```

Parameters

ddnn:

Specifies the device mnemonic (dd) and unit number (nn) for the device to be tested. You can also specify a logical name that has been assigned to the device.

blocknumber[, . . .]

Specifies the number of the block or blocks to be entered in the bad block list. If you specify more than one block number, separate the numbers with commas.

Format 2: Specifying a Range of Blocks

```
BADBLOCKS ddnn:beg:num
```

Parameters

ddnn:

Specifies the device mnemonic (dd) and unit number (nn) for the device to be tested. You can also specify a logical name that has been assigned to the device.

beg

Specifies the beginning block number.

num

Specifies the number of sequential blocks.

You can also use the two formats together on the same command line.

By using BADBLOCKS as a parameter command, you can omit the bad blocks on your disk from IOX testing. BADBLOCKS does not find bad blocks on your disk; instead, it allows you to enter blocks reported as bad (by the BAD utility) into a bad blocks list for the disk. When you exercise the disk with the SELECT or VERIFY commands, IOX does not exercise the blocks you enter in the list. Note that IOX omits only those blocks that you list in the BADBLOCKS command line; IOX assumes any blocks that are not on the list are good (even if the BAD utility has listed them as bad blocks.)

To use BADBLOCKS as a display command, specify BADBLOCKS with the device specification (ddnn), but do not include any block numbers.

Examples

Enters block 3 in the bad blocks list for the DM1 device.

Enters blocks 3, 4, 5, 11, 12, 13, 99, and 151 into the bad blocks list for DM1.

Enters blocks 3 to 5, 10 to 13, and 100 to 149 into the bad blocks list for DM1. You can extend the list of numbers for the length of a single command line. (IOX does not accept continuation lines.)

Enters blocks 3 to 7, 10 to 13, 99, 100 to 149, and 151 into the bad blocks list for DM1.

```
000003:005
000010:004
000099:051
000151:001
```

Displays the bad blocks list for DM1.

The number of sequential blocks was specified for block number 100, but not for block number 99. However, when IOX displays the bad blocks list, it does not separate the two blocks because they are consecutive. It strings consecutive blocks together, even if you did not enter all the block numbers.

Also, IOX does not display two lines (one for block 99 and one for blocks 100 to 149). Instead, it displays the bad blocks list in one line (line 3).

Note that you can name the blocks explicitly. In the previous example, if you entered 51 blocks beginning with block 99 in the bad blocks list (that is, 99:51), IOX would produce the same display.

12.6.3 BUFFERSIZE

The BUFFERSIZE parameter command is device dependent. Use BUFFERSIZE to determine the amount of data that IOX transfers for every read/write operation request when exercising NFS disks, terminals, magnetic tapes, and cassettes. IOX stores the specified size in bytes and rounds this number up to the next 4-byte boundary.

Format

```
BU[FFERSIZE] buffersize
```

If you use the random data pattern (PATTERN 0), the buffer size should not exceed 3400_{10} bytes. IOX responds to larger buffer sizes by generating unpredictable data compare errors. (For more information on the available data patterns, see the description of the PATTERN command.)

Although read/write buffers are used for testing Files–11 disks and NFS DECtapes, the buffer size for these devices is fixed. A 512_{10}-byte buffer is used for all Files–11 testing. A 1024_{10}-byte buffer is used for all DECtape testing.

The BUFFERSIZE command is valid only when used by itself in Command Mode. In addition, it is valid only if units are not currently selected; that is, you must set your default buffer size before you select any units for testing. However, you can use the BUFFERSIZE command in the SELECT and VERIFY command lines to override the default buffer size for the selected device as long as you set the buffer size smaller than the default.

The initial default is 1024_{10} bytes. The default buffer size for terminals is the size of the type-ahead buffer, if type-ahead is enabled. Otherwise, the default is 96_{10} bytes. (For more information on type-ahead buffers, see the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual.*)

12.6.4 COMPAREDATA

The COMPAREDATA command is a device-dependent parameter command for the IOX commands FILES11 and SELECT. It enables or disables the checking of the data pattern.

Format

```
COMPAREDATA option
```

Options

YES

Enables data pattern checking.

NO

Disables data pattern checking.

When you direct IOX to compare data patterns, it compares the data that it writes to the unit with the data it reads from the unit. If the data read does not match the data written, IOX prints a data compare error report. Section 12.5.2 discusses data compare error reports in detail.

You can use the COMPAREDATA command within the FILES11 and SELECT command lines to override the current COMPAREDATA setting for the device being tested.

The initial default is COMPAREDATA YES.

12.6.5 CONFIGURE

The CONFIGURE command is a parameter command and a display command. To add devices to your test configuration, use CONFIGURE as a parameter command. To display the current configuration, use CONFIGURE as a display command.

Format

```
CONF[IGURE] [ddnn:[=type]]
```

Parameters

ddnn:

Specifies the device mnemonic (dd) and unit number (nn) for the device to be included in the test configuration. You can also specify a logical name that has been assigned to the device.

If you specify more than one unit, separate the unit specifications with spaces or tabs.

type

Specifies the device type. By default, IOX configures one of each supported device type as unit 0. Allowable device types are: DISK, TERM (terminal), or TAPE (magnetic tape or cassette).

If you do not specify a device type when you configure a device, the device type default values are as follows:

- IOX maintains consistency in configuring device units. If one unit of a device type has already been configured as a disk and you add a unit with the same device specification, IOX also configures that unit as a disk. For example, if DK0 and DB0 are configured as disks in the default configuration, IOX also configures DK1, DB1, and DB2 as disks. By the same logic, if MM0 is configured as a tape unit by default, IOX also configures MM5 as a tape.

  Likewise, you cannot configure DK1, DB1, or DB2 as tapes, and you cannot configure MM5 as a disk. If the device type you specify is not consistent with the device unit, IOX displays the following error message:

  ```
  IOX---dduu: specified device type is inconsistent
  ```

- For devices with mnemonics that you have not configured previously, the default type is a disk.

You can add any device to the test configuration and assign it a logical device name by using the MCR (or DCL) command ASSIGN (ASN). When you configure a device, IOX adds the device to its list of devices available for testing and creates an empty data structure for the device.

If a device is not already included in the test configuration, you must configure it before you can select it for testing. IOX does not remove the devices from the configuration at the end of an exercise. Once you have included a device in the test configuration, only the IOX command RESTART can remove it.

If you specify CONFIGURE without any qualifiers, it displays the current test configuration. To find out if a device is already configured, use CONFIGURE as a display command:

```
IOX>
```

IOX responds as follows:

```
Disks:
DB0  DD0  DF0  DK0  DL0  DM0  DP0  DR0  DS0  DU0
DX0  DY0  EM0  SY0

Tapes:
CT0  DT0  MM0  MS0  MT0  MU0

Terminals:
TT0
```

Examples

```
IOX>
```

Configures both XX and YY as disks, and MQ1 as a tape. IOX also displays the following messages:

```
     IOX -- XX0: device type not specified - defaulting to 'DISK'
     IOX -- YY0: device type not specified - defaulting to 'DISK'
```

```
IOX>
```

Configures devices DK1, DB1, DB2, XX0, and YY0 as disk units. IOX configures devices MM5 and MQ1 as tape units.

```
IOX>
Disks:
DB0  DB1  DB2  DD0  DF0  DK0  DK1  DL0  DM0  DP0
DR0  DS0  DU0  DX0  DY0  EM0  SY0  XX0  YY0
Tapes:
CT0  DT0  MM0  MM5  MQ1  MS0  MT0  MU0
Terminals:
TT0
```

Displays the current device assignments (including the units added in the previous example.)

12.6.6 CONTROL_C

The CONTROL_C command is a general parameter command. Use CONTROL_C to enable or disable the use of CTRL/C for interrupting an I/O exercise.

If you are in Command Mode or Interactive Mode, the CONTROL_C command has no effect on the CTRL/C sequence. When you are in either of these two modes, pressing CTRL/C always returns you to the MCR. However, in Execution Mode, the function of CTRL/C depends upon the option you select with the CONTROL_C command. By default, CTRL/C allows you to exit Execution Mode and enter Interactive Mode.

Section 12.3 describes the three IOX operation modes in detail.

Format

```
CONTROL_C option
```

Options

YES

Enables CTRL/C and prevents IOX from accepting any characters other than CTRL/C as input. If you try to enter a command, you receive the following error message for each key you press:

```
IOX -- Only ^C allowed
```

When you press CTRL/C, you receive the IOX prompt instead of the MCR prompt. The prompt indicates that IOX has left Execution Mode; you are now in Interactive Mode. Note that this is the only way to access Interactive Mode from Execution Mode.

After you enter Interactive Mode, you can enter other IOX commands.

The initial default is CONTROL_C YES.

NO

Disables CTRL/C and allows IOX to accept CTRL/C as a command for returning control to MCR. You receive the MCR prompt (>) each time you press the RETURN key, and you receive the explicit prompt (MCR>) every time you press CTRL/C. This remains in effect until the exercise ends.

With CTRL/C disabled, you cannot interrupt the exercise to enter IOX commands. However, you can end the exercise with the MCR command ABORT. You can also issue other MCR commands while an exercise is in progress.

12.6.7 DENSITY

The DENSITY command is a device-dependent parameter command and a display command. It applies only to magnetic tape testing.

Format

```
DEN[SITY] ddnn: [n]
```

Parameters

ddnn:

Specifies the device mnemonic (dd) and unit number (nn) for the magnetic tape to be tested. You can also specify a logical name that has been assigned to the device.

n

Specifies the magnetic tape density in bits per inch (bpi).

The following tape densities are valid:

MM	MS	MT	MU
800	1600	200	1600
1600		556	6250
		800	

To use DENSITY as a device-dependent parameter command, issue the DENSITY command before you issue the SELECT command, or use the DENSITY command within the SELECT command line. The SELECT command selects the tape for non-file-structured testing, while the DENSITY command sets the tape density. IOX records the new density setting for the unit in the second characteristic word (U.CW2) of the Unit Control Block (UCB).

To use DENSITY as a display command, omit the argument (n). IOX displays the current density of the specified magnetic tape unit and the current bit settings for U.CW2. (For a description of the individual bit settings, see the *RSX-11M-PLUS and Micro/RSX I/O Drivers Reference Manual.*)

The DENSITY command is the only command that determines the current DENSITY setting on a magnetic tape (DENSITY is not displayed by the PARAMETERLIST command).

Examples

```
IOX>
Density= 1600 BPI Characteristic Word= 004000 (octal)
```

Displays the current density and characteristic word (U.CW2) for the tape MM0. (For an MM device, the density is 1600_{10} bpi when bit 11 of U.CW2 is set.)

```
IOX>
```

Changes the tape density to 800_{10} bpi.

DENSITY

```
IOX>
Density= 800 BPI Characteristic Word= 000000 (octal)
```

Displays the new tape density and characteristic word (U.CW2) for MM0. (For an MM device, the density is 800_{10} bpi when bit 11 of U.CW2 is cleared.)

```
IOX>
Density= ???? BPI     Characteristic Word= 004004 (octal)
```

Displays the current default setting for U.CW2. Because you have not set the density for MM, IOX does not display the current density. (However, for an MM device, the density is 1600_{10} bpi when bit 11 of U.CW2 is set.)

```
IOX>
IOX>
Density= 800 BPI      Characteristic Word= 001440 (octal)
```

Sets the tape density to 800_{10} bpi; then, displays the new settings for the density and the second characteristic word, U.CW2. (For an MM device, the density is 800_{10} bpi when bit 11 of U.CW2 is cleared.)

Except for MT devices, the initial default density for magnetic tape devices is 1600 bpi. The initial default for MT devices is 800 bpi.

12.6.8 DESELECT

The DESELECT command is an IOX function command. Use DESELECT to omit, or "deselect," devices that you previously selected for testing with the commands SELECT, VERIFY, or FILES11. Note that the DESELECT command *does not* remove a device from the test configuration.

Format

```
DES[ELECT] ddnn:
```

Parameter

ddnn:

Specifies the device mnemonic (dd) and unit number (nn) for the device to be tested. You can also specify a logical name that has been assigned to the device.

If you issue the command in Command Mode (while no testing is in progress), DESELECT removes the device from the list of devices to be tested. If you issue the DESELECT command in Interactive Mode (while testing of the specified device is in progress), IOX stops testing the device and removes the device from the list of devices to be tested.

For devices that you have selected for non-file-structured testing, DESELECT does the following:

- Terminates I/O operations (if they are in progress)
- Deletes the bad blocks list associated with the device (disks only)
- Deallocates all buffer space associated with the device
- Detaches IOX from the device

For devices that you have selected for FILES11 testing, DESELECT does the following:

- Terminates I/O operations (if they are in progress)
- Closes and deletes the temporary file
- Deallocates all buffer space associated with the device

12.6.9 ERRORLIMIT

The ERRORLIMIT command is a device-dependent parameter command that sets the maximum number of errors IOX tolerates between interval reports before IOX deselects the device.

Format

```
ER[RORLIMIT] n
```

Parameter

n

Specifies the maximum number of errors between interval reports before IOX stops testing the device. After the maximum number is reached, IOX removes the device from the list of devices to be tested.

The initial default is 10 errors.

To determine how often IOX generates interval reports, use the ERRORLIMIT command with the SUMMARYTIME command. (See Section 12.6.31 for more information.) To override the default error limit for the device being selected for testing, use ERRORLIMIT within the FILES11, SELECT, and VERIFY command lines.

Example

```
IOX>
IOX>
```

Requests interval reports every 2 minutes and sets the error limit at 10 errors. As a result of these settings, IOX does not allow more than 10 errors on a device in a 2-minute interval before deselecting the device from testing.

12.6.10 EXECUTE

The EXECUTE command is an IOX function command. Use the EXECUTE command to direct a command line to the operating system for execution.

To remove a task from the list of tasks to be executed and to terminate execution of the task (if the task is in progress), use the IOX or MCR command ABORT.

Format

```
EXE[CUTE] taskname commandstring
```

Parameters

taskname

Specifies the name of the installed task.

commandstring

Specifies the command line for the task you want to execute.

When you use the EXECUTE command in Interactive Mode (testing in progress), the task you initiate with EXECUTE runs immediately. When you use the EXECUTE command in Command Mode (no testing in progress), IOX queues the task for execution. The task begins executing when you start an exercise with the START command.

Tasks you initiate with EXECUTE run in parallel with I/O exercises. Therefore, you can use EXECUTE to run a task that does system testing while IOX is doing I/O testing. For example, if you have written and installed a central processing unit (CPU) exerciser task, you can run this task in parallel with IOX.

Likewise, you can use the EXECUTE command to run the Bad Block Locator Utility (BAD). BAD lists the bad blocks on a non-file-structured (NFS) disk at your terminal. To prevent IOX from performing I/O operations on the blocks that are listed as bad, enter this information into the IOX bad blocks list. (To enter information into the bad blocks list, use the BADBLOCKS command. For more information, see Section 12.6.2.)

Restrictions

The following restrictions apply to the EXECUTE command:

- You cannot use EXECUTE to run uninstalled tasks.
- EXECUTE only operates on systems that support task spawning.
- You must provide a number sign (#) for tasks whose command line processing requires a leading space (such as the BAD task). IOX converts the number sign to a space. (You do not need to include a number sign in the command line for tasks that do not require a leading space.)
- The rules for using the spawn (SPWN$) directive and entering command lines apply to the EXECUTE command arguments (taskname and commandstring). (For more information, see the *RSX-11M-PLUS and Micro/RSX Executive Reference Manual.*)

EXECUTE

Examples

```
IOX>
```

Runs the Bad Block Locator Utility (BAD) on the NFS disk DL0. The BAD command switch /LI tells BAD to list all the bad blocks.

```
IOX>
IOX>
```

Runs a CPU exerciser task that has been installed with the task name ...QWK. Also, IOX passes the command line /CPU=1144 to the task.

12.6.11 EXIT

The EXIT command is a control command. Use the EXIT command to terminate IOX processing and to exit from IOX to the operating system monitor.

Format

```
EXI[T]
```

12.6.12 FILES11

The FILES11 command is a function command for selecting a Files–11 disk for an I/O exercise. When you test a disk with the FILES11 command, you protect the contents and structure of your disk. IOX writes data into a temporary file, which it removes from the disk at the end of the test.

Before you can select a disk for testing, you must add the disk to the test configuration (see Section 12.2.2).

Format

```
F[iles11] ddnn: [command=qualifier] ...
```

Parameters

ddnn:

Specifies the device mnemonic (dd) and unit number (nn) for the device to be tested. You can also specify a logical name that has been assigned to the device.

command

Specifies a valid IOX command. To specify more than one command, separate the commands with spaces or tabs.

qualifier

Specifies the exercise parameter being set by the IOX command.

When you select a disk for testing with the FILES11 command, you can override the default parameters by using the following device-dependent parameter commands in the command line:

COMPAREDATA — Enables or disables checking of the data pattern (see Section 12.6.4).

ERRORLIMIT — Sets the maximum number of errors that IOX tolerates between interval reports before terminating testing on a device (see Section 12.6.9).

INTERLEAVE — Sets the number of blocks to be skipped before IOX writes the next buffer of data for disk testing (see Section 12.6.14).

RANDOM — Directs IOX to select either random or sequential blocks for testing (see Section 12.6.21).

TEMPORARYFILE — Sets the size of the temporary file that IOX uses for testing (see Section 12.6.32).

You can use these commands to set the default parameters before you use the FILES11 command to select a unit. You can also use these commands within the FILES11 command line to override the default parameters for the unit being selected. Howver, once you have selected a device, the default parameters become fixed for that device. You cannot reset them until after you deselect the device.

If you do not specify any commands within the FILES11 command line, the device is exercised with the default parameters currently in effect.

If you issue the FILES11 command in Interactive Mode (while an exercise is in progress), IOX begins testing the specified device immediately. If you issue the FILES11 command in Command Mode (while no exercise is in progress), IOX includes the specified device in its list of devices to exercise. To begin the exercise, use the START command.

Examples

```
IOX>
```

Selects the device DB0 for I/O exercising. IOX compares data, sequentially accesses the blocks in the file, and opens a 600-block temporary file.

Example 12–2 in Section 12.7 illustrates and describes how you test an RL02 disk with the FILES11 command.

12.6.13 HELP

The HELP command is a display command. To display a summary of IOX commands at your terminal, use the HELP command.

Format

```
H[ELP]
```

12.6.14 INTERLEAVE

The INTERLEAVE command is a parameter command that sets a device-dependent parameter for disk testing. The INTERLEAVE command sets the number of blocks to be skipped before IOX writes the next buffer of data.

Format

```
I[NTERLEAVE] number
```

Parameter

number

Specifeis the logical blocks numbers (LBNs) that IOX skips before it writes the next buffer of data. The initial default is four blocks.

You can use the INTERLEAVE command within the FILES11, SELECT, and VERIFY command lines to override the current default for the disk being selected.

Restriction

The INTERLEAVE factor applies only to sequential testing. To enable sequential testing, specify the RANDOM command with the NO option. (The initial default for the RANDOM command is YES; IOX tests blocks on the disk randomly and ignores the INTERLEAVE command.)

12.6.15 LOGFILE

The LOGFILE command is a general parameter command. Use the LOGFILE command to direct activity and error reports either to your terminal or to a log file in your default disk area. Sections 12.5.1 and 12.5.2 discuss the content and format of activity and error reports.

Format

```
LOGFILE option
```

Options

NO

Directs activity and error reports to your terminal. This is the initial default.

YES

Directs activity and error reports to the log file.

Summary reports, which IOX generates when you issue the PRINTSUMMARY command, are displayed at your terminal regardless of how you have set LOGFILE.

IOX does not open a new log file each time you run an exercise with LOGFILE enabled. Instead, IOX creates the file when you first run an exercise, and then it appends new data to the created file. The file is located in your User File Directory (UFD) and is named as follows:

```
yyyyyy.LOG
```

The prefix yyyyyy is the name of the task currently running IOX on your terminal (IOX, another task name, or the name of your terminal).

In addition to activity and error reports, IOX inserts the following date and time information into the log file:

- When the exercise begins, followed by the output of the PARAMETERLIST command
- When the exercise ends
- When IOX starts a log file
- When IOX ends log file entries
- When an exercise is aborted

For example, all exercises begin with an event line such as the following:

```
IOX -- 20-JUL-87 11:15:24 ******** EXERCISER STARTED
********
```

In addition, all exercises end with an event line similar to the following one:

```
IOX -- 20-JUL-87 11:20:34 ******** EXERCISER TERMINATED
********
```

12.6.16 LOOPBACK

The LOOPBACK command is a parameter command for exercising terminals. When LOOPBACK is enabled, IOX writes ASCII characters to a logged out terminal and then reads them back.

Format

```
L[OOPBACK] option
```

Options

YES

Enables LOOPBACK testing of a terminal line and interface.

NO

Disables LOOPBACK testing. IOX tests a terminal by displaying a portion of the ASCII character set.

This is the default.

LOOPBACK YES tests a terminal line and terminal interface. During the I/O exercise, characters are written by the interface to the terminal line. Then, the characters are reversed by the loopback connector, which sends them back up the line. After they are received by the interface, IOX reads them back into a buffer for comparison. If characters are not transmitted successfully throughout the exercise, IOX reports an error.

Restriction

The terminal characteristics SLAVE, PASSTHRU, NOECHO, LOWERCASE, and NOWRAP must be set *before* the loopback connector is installed.

12.6.17 PARAMETERLIST

The PARAMETERLIST command is a display command. PARAMETERLIST displays the following information:

- Current execution default parameters at the time you entered the PARAMETERLIST command
- IOX buffer space statistics

  IOX displays the statistics in the following format:

  ```
  totalmemory:freeblock:freebytes:fragments
  ```

Field	Description
totalmemory	Displays the size of IOX buffer space in bytes.
freeblock	Displays the size of the largest free block of IOX buffer space in bytes.
freebytes	Displays the total amount of free bytes available in IOX buffer space.
fragments	Displays the number of memory fragments in IOX buffer space.

- List of tasks initiated by the EXECUTE command (if applicable)
- Devices selected for testing with the FILES11, SELECT, and VERIFY commands

Examples

```
IOX>

BUFFERSIZE= 1024.          COMPAREDATA= YES
ERRORLIMIT= 10.            CONTROL_C= YES
INTERLEAVE= 4.             LOGFILE= NO
PATTERN= 0.                LOOPBACK= NO
RECORDS= 1024.             RANDOM= YES
RUNTIME= 5.                REPORTERRORS= YES
SUMMARYTIME= 1.            RETRIES= YES
TEMPORARYFILE= 500.        VOLUMECHECK= YES
                           WAIT= YES
                           WRITECHECK= NO

Buffer space usage= 3290.:1924.:1924.:1.
```

Displays the current default parameters and the IOX buffer space statistics.

Example 12–1 describes the PARAMETERLIST display in more detail.

Example 12–1: PARAMETERLIST Display Shown in Four Parts

```
IOX>EXECUTE ...BAD #DLO:/LI                              ❶
IOX>FILES11 SY: TEMPORARYFILE=800 ERRORLIMIT=50          ❷
IOX>SELECT MMO: RECORDS=350 BUFFERSIZE=700 RETRIES=NO
IOX>VERIFY DMO: RANGE 200:1000 RANDOM=NO INTERLEAVE=8    ❸
IOX>PARAMETERLIST

Device  Mode  Buffersize  Filesize/Records/Range  Parameters ❹
------  ----  ----------  ----------------------  ----------

DMO:    VFY      1024.      200. ->1,000.        RET INT=8. ERR=10.  ❺
MMO:    SEL       700.                350.         COM ERR=10.
SYO:    F11       512.                800.         COM RET RAN ERR=50. ❻

Task    Status    Command line
----    ------    ------------                ❼
...BAD  QUEUED    DLO:/LI

BUFFERSIZE= 1024.       COMPAREDATA= YES
ERRORLIMIT= 10.         CONTROL_C= YES
INTERLEAVE= 4.          LOGFILE= NO
PATTERN= 0.             LOOPBACK= NO
RECORDS= 1024.          RANDOM= YES              ❽
RUNTIME= 5.             REPORTERRORS= YES
SUMMARYTIME= 1.         RETRIES= YES
TEMPORARYFILE= 500.     VOLUMECHECK= YES
                        WAIT= YES
                        WRITECHECK= NO

Buffer space usage= 5338.:1020.:1588.:2.    ❾
```

❶ While IOX executes, the EXECUTE command runs the Bad Block Locator Utility (BAD). (For more information, see Section 12.6.10.)

❷ Selects SY: for FILES11 testing, overriding the defaults for the TEMPORARYFILE and ERRORLIMIT commands. IOX tests SY: as a Files–11 device, using the following parameters:

- A 512-byte data buffer (fixed)
- A TEMPORARYFILE size of 800 blocks
- COMPAREDATA enabled (by default)
- RETRIES enabled (by default)
- RANDOM YES (by default)
- An error limit of 50 errors

❸ Selects DM0 with the VERIFY command. IOX tests DM0, using the following parameters:

- A default buffer size of 1024_{10} bytes
- A range of blocks, 200 to 1000
- RETRIES enabled (by default)

- RANDOM disabled
- The INTERLEAVE factor set to 8
- A default error limit of 10

❹ Displays the selected devices (DM, MM, and SY:) and their device-dependent parameters. Unless the default parameters were overridden in the previous command lines, IOX displays the defaults.

This part of the display has five fields that are labeled by function. They are as follows:

Display Field	Description
device	Specifies the device mnemonic of the selected device.
mode	Indicates the testing mode selected: VERIFY, SELECT, or FILES11.
buffersize	Specifies the buffer size used for reading data from the device and writing data to the device. You can alter the buffer size for magnetic tapes, non-file-structured (NFS) disks, and cassettes. Files–11 disks have a fixed buffer size of 512_{10} bytes. DECtapes have a fixed buffer size of 1024_{10} bytes.
filesize records range	Indicates one of the following: • Temporary file size in blocks for disks selected with FILES11 • Number of records to test for magnetic tapes and cassettes selected with the SELECT command • Range of blocks to be tested on NFS disks selected with the SELECT or VERIFY command
parameter	Specifies device-dependent parameter commands. Parameter commands specify a value (such as INTERLEAVE=8. or ERRORLIMIT=10.) or a YES/NO setting. In a display, RAN indicates that RANDOM is set to YES; the absence of RAN indicates that RANDOM is set to NO.

❺ Display line corresponding to the VERIFY command line (see number 3).

❻ Display line corresponding to the FILES11 command line.

❼ List of task or tasks initiated by the EXECUTE command. In this example, only one task (BAD) was initiated.

This part of the display has the following fields:

Task Installed — Specifies the taskname (for example, ...BAD).

Status Execution — Specifies one of the following status fields:

- QUEUED
- ACTIVE
- SUCCESS
- WARNING
- ERROR
- SEVEREERROR

Command Line — Specifies the command line passed to the task.

❽ Current defaults at the time the PARAMETERLIST command was entered.

(In Command Mode, you can press CTRL/O to suppress the output of PARAMETERLIST. In Interactive Mode, pressing CTRL/O has no effect on the output.)

❾ IOX buffer space statistics.

In the examples, these statistics have changed to reflect the buffer allocations for the three devices that you selected for testing.

12.6.18 PATTERN

The PATTERN command is a parameter command and a display command. As a parameter command, PATTERN sets the data pattern that IOX writes and reads during I/O testing. As a display command, PATTERN displays the 13 available patterns and indicates the current default with an asterisk (*).

Format

```
PAT[TERN] [patternnumber]
```

Parameter

patternnumber

Specifies a number corresponding to the pattern that you want IOX to use during I/O testing. To use PATTERN as a display command, omit this argument.

The numbers and corresponding patterns are as follows:

Pattern Number	Data Pattern
0	Random data
1	000000,000000
2	177777,177777
3	125252,052525
4	165555,133333
5	163126,163126
6	055555,155555
7	022222,122222
8	111111,111111
9	007417,007417
10	021042,021042
11	104210,104210
12	052652,052652

Pattern 0 is a random data pattern from pure portions of IOX. It differs from block to block.

Restrictions

The following restrictions apply to the PATTERN command:

- When you use pattern 0, do not exceed a buffer size of 3400$_{10}$ bytes. When a buffer is larger than 3400$_{10}$ bytes, IOX generates unpredictable data compare errors. Note that pattern 0 is the initial default pattern.
- If you use logical assignments to direct multiple IOX exercises to one device, you cannot use pattern 0.

- To change the default pattern for the PATTERN command, you must be in Command Mode. The form of the PATTERN command that changes the pattern default is not valid in Interactive or Execution Mode.

Examples

```
IOX>

0  = Random Data   *
1  = 000000,000000
2  = 177777,177777
3  = 125252,052525
4  = 165555,133333
5  = 163126,163126
6  = 055555,155555
7  = 022222,122222
8  = 111111,111111
9  = 007417,007417
10 = 021042,021042
11 = 104210,104210
12 = 052652,052652
```

Displays the default pattern (pattern 0, random data).

```
IOX>
```

Changes the default pattern from random data to pattern 3 (125252,052525).

```
IOX>

0  = Random Data
1  = 000000,000000
2  = 177777,177777
3  = 125252,052525 *
4  = 165555,133333
5  = 163126,163126
6  = 055555,155555
7  = 022222,122222
8  = 111111,111111
9  = 007417,007417
10 = 021042,021042
11 = 104210,104210
12 = 052652,052652
```

Displays the new default pattern (the initial default was pattern 0). The asterisk indicates that the default is now pattern 3.

12.6.19 PRINTSUMMARY

The PRINTSUMMARY command is a display command that is valid only in Interactive Mode. When you enter the PRINTSUMMARY command, IOX generates summary reports that summarize IOX activity between the last interval report and the time you enter the command.

Summary reports and interval reports are different types of activity reports produced by IOX. Activity reports summarize IOX processing information, such as how long an exercise will run and how many asynchronous system traps (ASTs) have been executed. Summary reports and interval reports have the same content and format. However, IOX generates interval reports at time intervals set with the IOX command SUMMARYTIME. IOX generates summary reports when you enter the PRINTSUMMARY command in Interactive Mode.

IOX displays the summary reports at your terminal (even if LOGFILE is enabled).

For a complete description of activity reports, see Section 12.5.1.

12.6.20 PROCEED

The PROCEED command is a control command that is valid only in Interactive Mode. When you enter the PROCEED command, IOX exits Interactive Mode, enters Execution Mode, and checks for IOX activity. If no exercises are in progress and no tasks initiated by IOX are executing, then IOX terminates the exercise.

If there is IOX activity and you have disabled LOGFILE (set LOGFILE NO), IOX resumes the display of activity and error reports at your terminal. (When LOGFILE is disabled in Interactive Mode, IOX suspends the display of activity and error reports at your terminal.) If there is IOX activity and you have enabled LOGFILE (set LOGFILE YES), IOX continues recording activity and error reports in the log file.

Use the PROCEED command for one of the following reasons:

- If you want IOX to end an exercise as soon as IOX activity ends
- If LOGFILE is disabled and you want IOX to resume display of activity and error reports at your terminal.

For more information, see the description of the IOX command LOGFILE.

12.6.21 RANDOM

The RANDOM command is a device-dependent parameter command that applies to disk testing. It directs IOX to select either random or sequential blocks for disk testing.

Format

```
RAND[OM] option
```

Options

YES

Directs IOX to use its random number generator to choose the blocks on the selected disk for exercising.

RANDOM YES is the initial default.

NO

Directs IOX to test blocks on the disk sequentially.

If you set RANDOM NO, IOX skips the number of blocks specified with the INTERLEAVE command before testing the next block. (For more information, see the description of the IOX command INTERLEAVE.)

You can use the RANDOM command within the FILES11, SELECT, and VERIFY command lines to override the current default parameter, when selecting blocks for testing.

Examples

```
IOX>
IOX>
```

Tests every eighth block on the disk (that is, block 0, 7, 15 and so on).

12.6.22 RANGE

The RANGE command is a device-dependent parameter command and a display command. It applies to disks that you select for non-file-structured (NFS) testing with the IOX commands SELECT or VERIFY. (For more information on the SELECT and VERIFY commands, see the individual command descriptions. For an example of NFS testing, see Section 12.7.)

Format

```
RANG[E] ddnn: [minimum:maximum]
```

Parameters

ddnn:

Specifies the device mnemonic (dd) and unit number (nn) for the disk to be tested. You can also specify a logical name that has been assigned to the device.

minimum

Indicates the beginning block number for the range of blocks to be tested.

maximum

Indicates the ending block number for the range of blocks to be tested.

As a parameter command, RANGE sets the minimum and maximum block numbers for NFS testing on the specified disk unit. You can also use RANGE as a parameter command within the SELECT or VERIFY command lines.

As a display command, RANGE is valid only for devices you have previously selected for IOX testing. (For information on selecting a device, see Section 12.2.3.) RANGE displays the minimum and maximum block numbers for the specified disk unit. The display form of the RANGE command cannot be used within the SELECT or VERIFY command lines.

Examples

```
IOX>
```

Tests blocks in the range 300 to 500 on the device DL0.

```
IOX>
300. ->500.
```

Displays the current range of blocks to be tested. This display verifies that the range specified in the previous example is now in effect.

```
IOX>
0. ->20,420.
```

Displays the current range of blocks to be tested on the device DL0. In this example, the range of blocks is from 20 to 420. (This is the default for RL02 devices.)

The default range for all disks that have been selected is between zero and the maximum block number on the disk minus 60 (0:maximum –60).

12.6.23 RECORDS

The RECORDS command is a device-dependent parameter command. The RECORDS command applies to magnetic tapes and cassettes that you have selected for testing with the IOX command SELECT. Use the RECORDS command to specify the number of records on the magnetic tape or cassette to be exercised.

Format

```
REC[ORDS] recordcount
```

Parameter

recordcount

Specifies the number of records on the magnetic tape or cassette to be exercised. If you set the record count to 0, IOX tests the records on magnetic tapes and cassettes until it detects the end of the tape. At the end of the tape, IOX displays a message and deselects the tape unit.

You can use RECORDS in the SELECT command line to override the default record count for the magnetic tape or cassette being selected.

Examples

```
IOX>
```

Directs IOX to exercise 200 records on all magnetic tapes and cassettes.

The initial default is 1024_{10} records.

12.6.24 REPORTERRORS

The REPORTERRORS command is a general parameter command that enables or disables IOX error reports.

Format

```
REP[ORTERRORS] option
```

Options

YES

Enables IOX error reports. This is the initial default.

NO

Disables IOX error reports.

There are two kinds of error reports: *data compare error reports* and *I/O error reports*.

IOX reports a *data compare error* when the following two conditions are met:

1. A success code is returned from the device driver (meaning that a QIO succeeded).
2. The data that IOX wrote does not match the data that IOX read.

Section 12.5.2 illustrates and describes data compare error reports.

I/O error reports contain IOX error messages. I/O error codes are returned from the device drivers (meaning that a QIO failed). These error codes generate one of the error messages discussed in Section 12.8.

Restrictions

IOX directs error reports to your terminal, but it does so with the following restrictions:

- Error reports never interrupt activity reports.
- If LOGFILE is enabled, IOX enters error reports in the log file instead of displaying them at your terminal.
- In Interactive Mode, IOX suppresses the printing of error reports.

12.6.25 RESTART

The RESTART command is a control command that is valid only in Command Mode. When you restart IOX, you reinvoke IOX without exiting to the operating system monitor.

The RESTART command performs the following functions:

- Deselects devices that are selected for testing.
- Resets all parameters to their initial defaults.
- Removes all tasks queued for execution by the EXECUTE command from the list of tasks to be executed.
- Removes all devices from the default test configuration.

12.6.26 RETRIES

The RETRIES command is a device-dependent parameter command that applies to the testing of magnetic tapes and non-file-structured (NFS) disks.

Format

```
RET[RIES] option
```

Options

YES

Directs a device driver to use its standard recovery techniques to attempt to recover from an error.

This is the default.

NO

Inhibits an I/O driver from using its standard recovery techniques when it encounters an error.

You can use the RETRIES command within the SELECT and VERIFY command lines to override the current default for the device being selected.

12.6.27 RUNTIME

The RUNTIME command is a general parameter command that is valid only in Command Mode (that is, when no exercising is in progress).

Format

```
RUN[TIME] minutes
```

Parameter

minutes

Specifies the number of minutes that IOX tests the device or devices that you select for testing. If you specify RUNTIME 0, IOX continues testing the selected devices until you abort the exercise with either the IOX or MCR command ABORT.

The initial default is 5 minutes.

12.6.28 SELECT

The SELECT command is a function command. It selects a device with a non-file-structured (NFS) volume mounted for an I/O exercise that destroys the contents and structure of the volume. Before you can use SELECT, you must include the device in the test configuration.

Format

```
SE[LECT] dduu: [command=qualifier] ...
```

Parameters

ddnn:

Specifies the device mnemonic (dd) and unit number (nn) for the device to be tested. You can also specify a logical name that has been assigned to the device.

command

Specifies one of the commands listed in Table 12–2.

qualifier

Specifies the value for the parameter you are setting with the corresponding command.

Exercise parameters determine how IOX exercises units. These parameters set various aspects of testing, such as: how long an exercise is to run, how many records are to be tested on magnetic tapes, what range of blocks are to be tested on disks, and whether output is to be directed to your terminal or to a log file.

Use the following commands in conjunction with the SELECT command to set default parameters for exercising a terminal or a unit with an NFS volume:

BADBLOCKS
BUFFERSIZE
COMPAREDATA
DENSITY
ERRORLIMIT
INTERLEAVE
LOOPBACK
RANDOM
RANGE
RECORDS
RETRIES
VOLUMECHECK
WRITECHECK

With the exception of the BADBLOCKS command, you can use these commands within the SELECT command line to override the current default settings. For more information on each command, see the individual command descriptions.

There are two kinds of exercise parameters: general and device dependent. Table 12–2 lists the device-dependent parameters for exercising different types of devices.

Table 12–2: Parameter Commands by Device Type for the SELECT Command

				Magnetic	
Parameter Command	**NFS Disks**	**Cassettes**	**DECtapes**	**Tapes**	**Terminals**
BADBLOCKS[1]	X				
BUFFERSIZE	X	X		X	X
COMPAREDATA	X	X	X		X
DENSITY				X	
ERRORLIMIT	X	X	X	X	X
INTERLEAVE	X				
LOOPBACK					X
RANDOM	X				
RANGE	X				
RECORDS		X		X	
RETRIES	X			X	
VOLUMECHECK	X				
WRITECHECK	X				

[1] Cannot be used within SELECT command line.

Once you have selected a device, the default parameters become fixed for that device. You cannot reset them without first "deselecting" the device.

When you issue the SELECT command in Interactive Mode (while an exercise is in progress), IOX begins testing the specified device immediately. When you issue the SELECT command in Command Mode (while no exercise is in progress), IOX includes the specified device in its list of devices to exercise but does not begin the exercise until you start the exercise with the START command.

If no further parameter commands are specified within the SELECT command line, the device is exercised with the parameters currently in effect.

Note that commands can be separated only by spaces or tabs.

Example

```
IOX>
```

This command line does the following:

- Selects DL0 for I/O exercising
- Tests only blocks 100 to 4000
- Allows the driver to retry I/O operations if it encounters errors
- Accesses the blocks on the volume sequentially testing every eighth block
- Uses a data buffer of 2000 bytes for data transfer

Example 12–3 in Section 12.7 illustrates and describes how to test a magnetic tape unit with the SELECT command.

12.6.29 SPY

The SPY command is a display command. To display status information on one or more devices being tested, use the SPY command in Interactive Mode.

Format

```
SP[Y] [ddnn:]
```

Parameter

ddnn:

Specifies the device mnemonic (dd) and unit number (nn) for the device. You can also specify a logical name that has been assigned to the device.

If you do not specify a device, IOX displays status information on all devices being tested. Otherwise, SPY displays status information only for the specified device.

Example

```
>
DB0: -- TYPE=F11   VBN=305.   I/O=AST   queued   FUNCTION=WRITE
```

This display indicates the following:

- Device being exercised (DB0)
- Type of testing being performed on the device: FILES11, SELECT, or VERIFY
- Virtual block number (VBN) being tested (For non-file-structured [NFS] volumes, IOX displays the logical block number [LBN])
- I/O request status: asynchronous system trap (AST) queued, in progress, or executing an idle loop
- Type of I/O function being performed: READ or WRITE

SPY always outputs to your terminal, even if LOGFILE is enabled.

12.6.30 START

The START command is a control command that is valid only in Command Mode. When you use the START command, IOX exits Command Mode and enters Execution Mode. Section 12.3 discusses the three IOX operating modes in detail.

After you have set your test configuration and have selected the devices you want to test, use the START command to begin exercising the selected units. START also turns on the IOX clock, begins activity and error reporting (if you enabled error reporting with REPORTERRORS YES), and begins executing tasks queued by the EXECUTE command.

12.6.31 SUMMARYTIME

The SUMMARYTIME command is a general parameter command that specifies how often IOX generates interval reports. (For a complete discussion of interval reports, see Section 12.5.1).

Format

```
SU[MMARYTIME] minutes
```

Parameter

minutes

Specifies the time interval (in minutes) between interval reports. To suppress interval reports, set SUMMARYTIME to 0.

The initial default is 1 minute between interval reports.

12.6.32 TEMPORARYFILE

The TEMPORARYFILE command is a device-dependent parameter command that applies to the testing of Files–11 disks with the FILES11 command. The FILES11 command protects the contents of the disk by writing them to a temporary file, which IOX deletes at the end of the exercise.

Format

```
T[EMPORARYFILE] filesize
```

Parameter

filesize

Specifies the size of the temporary file in blocks.

Use the TEMPORARYFILE command in the FILES11 command line to override the current default for the disk being selected.

The initial default is 500 blocks.

VERIFY

12.6.33 VERIFY

The VERIFY command is a function command you use to select a mounted non-file-structured (NFS) disk for an exercise. It reads buffers of data without writing to the disk, performing data comparisons or destroying the contents of the disk.

Format

```
VE[RIFY] ddnn: [command=qualifier] ...
```

Parameters

ddnn:

Specifies the device mnemonic (dd) and unit number (nn) for the device to be tested. You can also specify a logical name that has been assigned to the device.

command

Specifies one of the commands listed in Table 12–2.

qualifier

Specifies the value for the parameter you are setting with the corresponding command.

VERIFY accepts the following device-dependent parameter commands within its command line:

BUFFERSIZE — Sets the amount of data that IOX transfers for every read or write request (see Section 12.6.3).

ERRORLIMIT — Sets the maximum number of errors that IOX tolerates between interval reports before terminating testing on a device (see Section 12.6.9).

INTERLEAVE — Sets the number of blocks to be skipped before IOX writes the next buffer of data for disk testing (see Section 12.6.14).

RANDOM — Directs IOX to select either random or sequential blocks for testing (see Section 12.6.21).

RANGE — Sets the minumum and maximum block numbers for NFS testing on the specified disk (see Section 12.6.22).

RETRIES — Determines whether or not the device driver repeats an I/O operation when it receives errors during the operation (see Section 12.6.26).

Once you have selected a device, the default parameters for that device are fixed. You cannot reset them without first deselecting the device. If you do not specify any parameter commands within the VERIFY command line, the disk is exercised with the parameters currently in effect.

When you use the VERIFY command in Interactive Mode (while an exercise is in progress), IOX begins testing the specified device immediately. When you issue the VERIFY command in Command Mode (while no exercise is in progress), IOX includes the specified device in its list of devices to exercise but does not begin the exercise until you start the exercise with the START command.

Note that the parameter commands can be separated only by spaces or tabs.

Example

```
IOX>
```

Illustrates how to use the ERRORLIMIT and RANGE commands within the VERIFY command line to override the current defaults for testing the device DM0. This command line selects DM0 for read-only testing of blocks 1000 to 3000 (inclusive) with a maximum of 1000 errors before IOX deselects the disk.

Example 12–4 in Section 12.7 illustrates and explains how to test an NFS disk with VERIFY.

12.6.34 VOLUMECHECK

The VOLUMECHECK command is a device-dependent parameter command that applies to NFS disk testing with the IOX command SELECT. VOLUMECHECK enables or disables checking of disks mounted as non-file-structured (NFS) volumes for a home block and a Files–11 structure.

Format

```
VO[LUMECHECK] option
```

Options

YES

Enables VOLUMECHECK. Prevents you from using the SELECT command to select a volume with a Files–11 structure. You should use VOLUMECHECK YES unless you are intentionally writing over a Files–11 disk.

NO

Disables VOLUMECHECK. IOX allows you to use the SELECT command to select the disk without warning you that the test destroys the contents of the disk. Do not use VOLUMECHECK NO unless you are intentionally writing over a Files–11 disk.

You can use the VOLUMECHECK command within the SELECT command line to override the current default values for the disk being selected.

The initial default is VOLUMECHECK YES.

12.6.35 WAIT

The WAIT command is a general parameter command that determines whether or not IOX waits for an event flag while an exercise is in progress. The WAIT command has no effect on tasks initiated with the EXECUTE command.

Format

```
WA[IT] option
```

Options

YES

Requires IOX to wait for an event flag to signal that an I/O operation has completed. During this time, the Executive may be able to execute another task.

NO

Does not require IOX to wait for an event flag; IOX executes an idle loop during I/O operations. Setting WAIT NO occupies system resources; therefore, you should not specify this option unless you intentionally want to monopolize resources.

The initial default is WAIT YES.

12.6.36 WRITECHECK

The WRITECHECK command is a device-dependent parameter command that applies only to testing non-file-structured (NFS) disks with the IOX command SELECT. Use WRITECHECK to determine whether or not I/O drivers check write requests to NFS disks.

If you enable WRITECHECK, IOX writes data to the disks by issuing the I/O request Write Logical Block With Check (IO.WLC) to the driver instead of Write Logical Block (IO.WLB). The disk controller reads, word by word, a copy of the data from the data buffer, while simultaneously reading the same data from the disk. Then, the disk controller compares the two copies of the data. If the copies do not match, the driver repeats the original write operation and performs another write-check. This process continues until the WRITECHECK function succeeds or the retry count for the driver is exceeded.

If RETRIES is disabled, or if RETRIES is enabled but the retry count for the driver is exceeded, the driver returns an error code (IE.WCK) and IOX generates an error message. (For more information on the IOX command RETRIES, see Section 12.6.26.)

You can use the WRITECHECK command within the SELECT command line to override the current default for the device being selected.

Format

```
WR[ITECHECK] option
```

Options

YES

Enables WRITECHECK. This is the default.

NO

Disables WRITECHECK.

12.7 Examples

This section provides examples of setting parameters for different types of devices by using the commands FILES-11, SELECT, and VERIFY.

12.7.1 Setting Parameters for Exercising Files-11 Volumes

This section shows you how to use IOX to exercise an RL02 with a Files–11 structure. Example 12–2 uses the following IOX commands:

- PARAMETERLIST
- RUNTIME
- FILES11
- TEMPORARYFILE
- START

For a brief description of each command, see Table 12–1. For more detailed information, see the individual command descriptions.

Example 12–2: Exercising an RL02 with the FILES11 Command

```
>                        ❶
>                        ❷
>                        ❸

IOX>PARAMETERLIST
BUFFERSIZE= 1024.            COMPAREDATA= YES
ERRORLIMIT= 10.              CONTROL_C= YES
INTERLEAVE= 4.               LOGFILE= NO
PATTERN= 0.                  LOOPBACK= NO
RECORDS= 1024.               RANDOM= YES
RUNTIME= 5. ❼               REPORTERRORS= YES          ❹
SUMMARYTIME= 1.              RETRIES= YES
TEMPORARYFILE= 500. ❽       VOLUMECHECK= YES
                             WAIT= YES
                             WRITECHECK= NO

Buffer space usage= 3290.:1924.:1924.:1.

IOX>                           ❺
IOX>                     ❻
IOX>

Device  Mode  Buffersize  Filesize/Records/Range  Parameters
------  ----  ----------  ----------------------  ----------        ❾
DL0:    F11      512.            800.     ❿     COM RET RAN ERR=10.

BUFFERSIZE= 1024.            COMPAREDATA= YES
ERRORLIMIT= 10.              CONTROL_C= YES
INTERLEAVE= 4.               LOGFILE= NO
PATTERN= 0.                  LOOPBACK= NO
RECORDS= 1024.               RANDOM= YES
RUNTIME= 3.            ❼    REPORTERRORS= YES
SUMMARYTIME= 1.              RETRIES= YES
TEMPORARYFILE=500.     ❽    VOLUMECHECK= YES
                             WAIT= YES
                             WRITECHECK= NO

Buffer space usage= 3290.:1308.:1308.:1.

IOX>                   ⓫
```

The command sequence in Example 12–2 does the following:

❶ Allocates drive DL0.

❷ Mounts the disk with label "LOUIE" in DL0.

❸ Invokes an installed version of IOX with default task name, ...IOX.

❹ Displays the initial default parameters with PARAMETERLIST.

❺ Changes the default RUNTIME to 3 minutes.

❻ Selects DL0 for FILES11 testing with a temporary file size of 800 blocks.

❼ Note the difference between the defaults displayed by the PARAMETERLIST command before and after you changed RUNTIME and selected DL0. Only the RUNTIME default has changed.

❽ The default for TEMPORARYFILE has not changed.

❾ After you selected DL0 for testing, PARAMETERLIST contains an extra line at the beginning that displays all device-dependent parameters for the selected device. Each selected device has its own display line. There are five fields of parameters.

❿ Field four shows that the temporary file size used for FILES11 testing for DL0 is 800_{10} blocks instead of the 500_{10}-block initial default. During FILES11 testing, this field refers only to temporary file size.

⓫ The START command begins FILES11 testing on DL0 for a RUNTIME of 3 minutes and uses a temporary file size of 800_{10} blocks.

12.7.2 Setting Parameters for Exercising NFS Volumes

This section shows examples for the command sequences that test a magnetic tape volume using the SELECT command and that test a non-file-structured (NFS) disk using the VERIFY command. Both examples are explained in detail.

Testing a Magnetic Tape with the SELECT Command

The command sequence in Example 12–3 tests a magnetic tape by using a buffer of 8 Kb (which contains the data pattern 052652) to transfer data.

Example 12–3 illustrates the following commands:

- PARAMETERLIST
- PATTERN
- BUFFERSIZE
- DENSITY
- SELECT
- RETRIES
- ERRORLIMIT
- RECORDS
- START
- EXIT

For a brief description of each command, see Table 12–1. For more detailed information, see the individual command descriptions.

Example 12–3: Exercising a Magnetic Tape with the SELECT Command

```
> [illegible]
> [illegible]                                 ❶
> [illegible]
IOX> [illegible]

BUFFERSIZE= 1024.        ❽     COMPAREDATA= YES
ERRORLIMIT= 10.                 CONTROL_C= YES
INTERLEAVE= 4.                  LOGFILE= NO
PATTERN= O.              ❾     LOOPBACK= NO
RECORDS= 1024.                  RANDOM= YES                    ❷
RUNTIME= 5.                     REPORTERRORS= YES
SUMMARYTIME= 1.                 RETRIES= YES
TEMPORARYFILE= 500.             VOLUMECHECK= YES
                                WAIT= YES
                                WRITECHECK= NO

Buffer space usage= 3290.:1924.:1924.:1.

IOX> [illegible]        ❸
IOX> [illegible]        ❹
IOX> [illegible]
  Density= 1600 BPI    Characteristic Word= 004004 (octal)    ❺
IOX> [illegible]                                               ❻
IOX> [illegible]

Device  Mode  Buffersize  Filesize/Records/Range  Parameters
------  ----  ----------  ----------------------  ----------
MMO:    NFS      8192. ⓫      20000. ⓬       COM ERR=10. ❿

BUFFERSIZE= 8192.        ❽     COMPAREDATA= YES
ERRORLIMIT= 10.                 CONTROL_C= YES
INTERLEAVE= 4.                  LOGFILE= NO
PATTERN= 12.             ❾     LOOPBACK= NO
RECORDS= 1024.                  RANDOM= YES                       ❼
RUNTIME= 5.                     REPORTERRORS= YES
SUMMARYTIME= 1.                 RETRIES= YES
TEMPORARYFILE= 500.             VOLUMECHECK= YES
                                WAIT= YES
                                WRITECHECK= NO

Buffer space usage= 36058.:8188.:18292.:3.

IOX> [illegible]
   Density= 800 BPI    Characteristic Word= 000000 (octal)
IOX> [illegible]      ⓭
IOX -- MMO: bad block, 112 (decimal) 160 (octal)
```

(Continued on next page)

Example 12–3 (Cont.): Exercising a Magnetic Tape with the SELECT Command

```
IOX -- Summary of exerciser activity at 20-JUL-87 16:50:40
  TIME -- Run: 1 min.   Elapsed: 1 min.   Remaining: 4 min.
  MMO: -- 230. R/W requests totaling 230. records with 1 errors
  AST's executed: 346.
  Idle-loop iterations: 344.
      .
      .
      .

IOX -- MMO: bad block, 486 (decimal) 746 (octal)
IOX -- MMO: bad block, 551 (decimal) 1047 (octal)

IOX -- Summary of exerciser activity at 20-JUL-87 16:54:50
  TIME -- Run: 1 min.   Elapsed: 5 min.   Remaining: 0 min.
  MMO: -- 217. R/W requests totaling 217. records with 2 errors
  AST's executed: 325.
  Idle-loop iterations: 321.

 A C C U M U L A T E D   T O T A L S :
 - - - - - - - - - - -   - - - - - - -
  TIME -- Run: 5 min.   Elapsed: 5 min.   Remaining: 0 min.
  MMO: -- 1,121. R/W requests totaling 1,121. records with 3 errors
  AST's executed: 1,686.
  Idle-loop iterations: 1,670.

IOX>
>
```

The command sequence in Example 12–3 does the following:

❶ Allocates drive MM0, mounts the magnetic tape, and invokes an installed version of IOX. Note that the tape is mounted with the /FOREIGN switch.

❷ Displays the initial defaults.

❸ Selects pattern number 12 to be written to the tape.

❹ Changes the default BUFFERSIZE from 1024_{10} bytes to 8192_{10} bytes (8 Kb).

❺ Displays the default density and characteristic word on the magnetic tape mounted in drive MM0.

❻ Selects MM0 for NFS testing with SELECT, overriding the defaults for RETRIES, ERRORLIMIT and RECORDS, and sets the tape density at 800_{10} bpi.

❼ Checks the parameters with PARAMETERLIST and DENSITY.

❽ Changes the BUFFERSIZE default.

❾ Changes the PATTERN default.

❿ Displays the device display line for MM0.

⓫ Shows the current BUFFERSIZE of MM0.

⓬ Shows that the number or records to be tested is 20,000.

⓭ Starts NFS testing with SELECT on MM0.

The remainder of Example 12-3 shows interval reports, error reports, and accumulated totals generated at your terminal during this exercise. These reports are discussed fully in Section 12.5. When the exercise has finished, exit to the operating system monitor (unless you want to do more testing).

Exercising with the VERIFY Command

The following commands affect the exercise parameters of NFS disks that you select with the VERIFY command:

BADBLOCKS
BUFFERSIZE
ERRORLIMIT
INTERLEAVE
RANDOM
RANGE
RETRIES

Use these commands with the VERIFY command to set default parameters for testing a unit with an NFS disk. You can use all but the BADBLOCKS command within the VERIFY command line.

The command sequence in Example 12-4 performs oscillating seeks between blocks 1 and 1000 to align disk heads. It reads only 4 bytes per data transfer, which is the minimum. No error or summary reporting is required. The exercise runs until you abort it. Example 12-4 illustrates the following commands:

- PARAMETERLIST
- VERIFY
- RANGE
- BUFFERSIZE
- RANDOM
- INTERLEAVE
- ERRORLIMIT
- REPORTERRORS
- SUMMARYTIME
- RUNTIME
- LOGFILE
- START
- CTRL/C
- ABORT

For a brief description of each command, see Table 12-1. For more detailed information, see the individual command descriptions.

Example 12–4: Exercising an NFS RK07 with the VERIFY Command

```
>
>                       ❶
>
>

BUFFERSIZE= 1024.            COMPAREDATA= YES
ERRORLIMIT= 10.              CONTROL_C= YES
INTERLEAVE= 4.               LOGFILE= NO
PATTERN= 0.                  LOOPBACK= NO
RECORDS= 1024.               RANDOM= YES
RUNTIME= 5.                  REPORTERRORS= YES          ❷
SUMMARYTIME= 1.              RETRIES= YES
TEMPORARYFILE= 500.          VOLUMECHECK= YES
                             WAIT= YES
                             WRITECHECK= NO

Buffer space usage= 3290.:1924.:1924.:1.

IOX>                                                        ❸
IOX>
IOX>
IOX>                             ❹
IOX>
IOX>

Device  Mode  Buffersize  Filesize/Records/Range  Parameters
------  ----  ----------  ----------------------  ----------
 DMO:   VFY       4.      1. ->1,000.            RET INT=999. ERR=50000.

BUFFERSIZE= 1024.            COMPAREDATA= YES
ERRORLIMIT= 10.              CONTROL_C= YES
INTERLEAVE= 4.               LOGFILE= YES                           ❺
PATTERN= 0.                  LOOPBACK= NO
RECORDS= 1024.               RANDOM= YES
RUNTIME= 0.                  REPORTERRORS= NO
SUMMARYTIME= 0.              VOLUMECHECK= YES
TEMPORARYFILE= 500.          WAIT= YES
                             WRITECHECK= NO

Buffer space usage= 3290.:1912.:1912.:1.

IOX>
IOX>                     ❻
IOX>
```

The command sequence in Example 12–4 does the following:

❶ Allocates drive DM0, mounts the RK07 as foreign (MOU /FOR) for NFS testing, and invokes IOX.

❷ Displays the initial default parameter.

❸ Selects DM0 for read-only NFS testing with VERIFY, overriding the following defaults:

RANGE — Set for blocks 1 to 1000.

BUFFERSIZE — Set at 4 bytes (minimum).

RANDOM — Disables block testing and sets INTERLEAVE at 999. IOX tests blocks 1 and 1000 only.

ERRORLIMIT — Set to $50{,}000_{10}$ (extremely high) so that IOX will not deselect DM0, regardless of how many errors are encountered during the exercise.

❹ Turns off REPORTERRORS so that no error reports are generated. Turns SUMMARYTIME off so that no interval reports will be generated. Sets RUNTIME to 0 so that IOX runs the exercise until you abort it. Sets LOGFILE YES to direct all output to the log file, IOX.LOG, instead of to the terminal.

❺ Shows that the RUNTIME, SUMMARYTIME, and LOGFILE defaults have been reset and that DM0 is being selected with the VERIFY command. The exercise conditions are as follows:

- A buffer size of 4_{10}
- A range of 1000_{10} blocks
- RETRIES enabled
- INTERLEAVE set at 999
- ERRORLIMIT set at 50,000

❻ Shows how to start the exercise with the START command and how to enter Interactive mode by pressing CTRL/C. To abort the exercise, enter the ABORT command.

The following display illustrates the contents of the log file for Example 12–4:

```
IOX -- 20-JUL-87 18:24:12 ******** EXERCISER STARTED ********

Device  Mode  Buffersize  Filesize/Records/Range  Parameters
------  ----  ----------  ----------------------  ----------
DMO:    VFY      4.         1. ->1,000.          RET INT=999. ERR=50000.

BUFFERSIZE= 1024.             COMPAREDATA= YES
ERRORLIMIT= 10.               CONTROL_C= YES
INTERLEAVE= 4.                LOGFILE= YES
PATTERN= 0.                   LOOPBACK= NO
RECORDS= 1024.                RANDOM= YES
RUNTIME= 0.                   REPORTERRORS= NO
SUMMARYTIME= 0.               RETRIES= YES
TEMPORARYFILE= 500.           VOLUMECHECK= YES
                              WAIT= YES
                              WRITECHECK= NO

Buffer space usage= 3290.:1812.:1812.:1.

IOX -- Summary of exerciser activity at 20-JUL-87 18:27:28
  TIME -- Run: 3 min.   Elapsed: 3 min.   Remaining: 65533 min.
  DMO: -- 3,075. R/W requests totaling 3,075. blocks with no errors
  AST's executed: 3,081.
  Idle-loop iterations: 2,982.
```

```
 A C C U M U L A T E D   T O T A L S :
 - - - - - - - - - - -   - - - - - - -
  TIME -- Run: 0 min.   Elapsed: 3 min.   Remaining: 65533 min.
  DMO:A -- 3,076. R/W requests totaling 3,076. blocks with no errors
  AST's executed: 3,082.
  Idle-loop iterations: 2,982.

IOX -- 20-JUL-87 18:27:31 ******** EXERCISER ABORTED
********
```

The log file output terminates when you abort IOX. The log file begins with the time and date that the exercise began and the current default parameters as displayed by the PARAMETERLIST command.

12.8 IOX Error Messages

IOX generates two types of error messages: error messages with device specifications and error messages without device specifications. IOX displays device specifications as appropriate.

The format for an error message with a device specification is as follows:

```
taskname -- ddnn: message
```

Parameters

taskname

Specifies the name of the IOX task that is currently running. In the error messages that follow, taskname is always IOX.

ddnn

Represents the device specification text of the message.

message

Represents information specific to this message.

The format for an error message without a device specification is as follows:

```
taskname--message
```

The taskname and message parameters are the same as those parameters defined in the previous format.

All error messages use the letter "x" to indicate that IOX substitutes the appropriate command, task name, or value for "x" within the message.

12.8.1 Error Messages with Device Specifications

This section lists IOX error messages in alphabetical order, according to the first letter that follows the device specification.

IOX—dduu: bad block, xxx (decimal) xxx (octal)

Explanation: IE.BBE error code returned to IOX (bad block error).

User Action: Use the BADBLOCKS command (only for NFS disks) to enter the block number into the bad blocks list.

IOX—dduu: block number out of range

Explanation: You used the BADBLOCKS command to enter a block number in the bad blocks list that is outside the range of blocks that you specified with the RANGE command.

User Action: Reenter the command with the correct block number.

IOX—TTuu: buffer size being reduced to typeahead buffer size

Explanation: The buffer size specified or used by default is larger than the size of the type-ahead buffer.

User Action: No user action is required. IOX will reduce the buffer size to equal the size of the type-ahead buffer.

IOX—dduu: buffer size is greater than default

Explanation: The buffer size that you specified in the SELECT or VERIFY command line is greater than the current default set with the BUFFERSIZE command by itself.

User Action: Either raise the default buffer size (using BUFFERSIZE by itself) or lower the buffer size override for the device (using BUFFERSIZE within the SELECT or VERIFY command line).

IOX—dduu: data compare error at block xxx (decimal) xxx (octal)

```
Good data: xxxxx bad data: xxxxx
Word position: xxx (decimal) xxx (octal)
Buffer contained xx. additional errors
```

Explanation: IOX found an error when comparing the data read with the data written. This error message shows the first block number in which the error occurred during the transfer, the good and bad data, the word position of the first error found in the buffer, and the number of additional errors found in the buffer. Section 12.5.2 discusses data compare error reports in detail.

User Action: No user action is required. If the error persists, enter the block in the bad blocks list by using the BADBLOCKS command (for NFS disks only).

IOX—dduu: data overrun at block xxx (decimal) xxx (octal)

Explanation: IE.DAO error code returned to IOX (data overrun). The record size given was greater than the record size read. This error occurs on tape devices only.

User Action: No user action is required.

IOX—dduu: device type not specified—defaulting to 'DISK'

Explanation: This message tells you that you are adding a device to your test configuration with the CONFIGURE command and that IOX is configuring that device as a disk by default.

User Action: No user action is required.

IOX—dduu: did not respond—unit now deselected

Explanation: The unit did not respond to the IOX command within 1 minute. The device may have been accidentally put off line or a tape drive may have lost its column vacuum.

User Action: Check the device.

IOX—TTuu: does not support the LOOPBACK command

Explanation: LOOPBACK cannot be used with the selected terminal line. This error occurs because type-ahead is disabled.

User Action: Use the DCL command SET TERMINAL /TYPE_AHEAD (MCR command SET /TYPEAHEAD) to enable type-ahead for the selected terminal.

IOX—dduu: does not support the x command

Explanation: The x command is incompatible with the device specification dduu.

User Action: For a list of the device-dependent parameter commands supported by different devices, see Table 12–2. Enter parameter commands that are compatible with the type of device you are selecting.

IOX—dduu: duplicate block number

Explanation: The block number you entered with the BADBLOCKS command is already in the bad blocks list for the device.

User Action: No user action is required.

IOX—dduu: end of tape detected—unit now deselected

Explanation: IE.EOT error code returned to IOX (end-of-tape detected). IOX attempted to write a record past the end-of-tape marker.

User Action: This is an informational message only.

IOX—dduu: error threshold exceeded—unit now deselected

Explanation: The number of errors set with the ERRORLIMIT command has been exceeded.

User Action: Repeat the exercise with a higher value for the ERRORLIMIT parameter. If you have already specified the maximum value for ERRORLIMIT and this message is displayed, the device you selected may be defective.

IOX—dduu: error while reading home block

Explanation: When VOLUMECHECK is enabled, this message occurs if IOX detects an error while attempting to read the home block of a Files–11 device.

User Action: The disk you selected may be defective. If you are testing a scratch disk, select the disk again with VOLUMECHECK NO.

IOX—dduu: failed to attach

Explanation: IOX could not attach the unit.

User Action: The device may be mounted or attached by someone else. Make sure you have allocated the device and have mounted it with the /FOREIGN qualifier (/FOR keyword).

IOX—dduu: failed to detach

Explanation: An error occurred while IOX was attempting to detach a unit that it had previously attached.

User Action: No user action is required. However, if this problem persists, submit a Software Performance Report (SPR) to DIGITAL.

IOX—dduu: failed to open temporary file

Explanation: IOX could not find enough free blocks on the disk to open a temporary file as large as the one you specified with the TEMPORARYFILE command.

User Action: Specify a smaller temporary file size either by resetting the default with the TEMPORARYFILE command or by using TEMPORARYFILE within the FILES11 command line to override the default TEMPORARYFILE size for each device.

If you are using the default (500 blocks) when IOX generates this message, you need to provide additional space on the disk.

IOX—dduu: fatal hardware error—unit now deselected

Explanation: The system returned the error code IE.FHE to IOX. This indicates a fatal hardware error.

User Action: Check the hardware.

IOX—dduu: illegal density

Explanation: You specified an invalid magnetic tape density with the DENSITY command.

User Action: Reenter the command with a valid density (see Section 12.6.7).

IOX—dduu: illegal range

Explanation: The range of blocks that you specified for the device to be tested with the RANGE command is not within the limits for the device.

User Action: To display the default range for the device, enter the RANGE command without specifying any parameters; then, specify a range within these limits.

IOX—dduu: invalid block number

Explanation: The block number that you entered with the BADBLOCKS command was not a valid block number for the device.

User Action: No user action is required.

IOX—dduu: I/O is unsatisfied at timeout—function: xxxx

Explanation: When IOX finishes processing, it waits 5 seconds for I/O to stop. If a long tape has not finished rewinding, this message may occur. The function that timed out is xxxx.

User Action: No user action is required.

IOX—dduu: I/O request aborted—unit now deselected

Explanation: The driver returned the error code IE.ABO to IOX. The operation aborted.

User Action: No user action is required.

IOX—dduu: is already configured

Explanation: You tried to configure a unit (with the CONFIGURE command) that is already in the test configuration.

User Action: Use CONFIGURE as a display command to display the devices in the current test configuration.

IOX—dduu: is already selected

Explanation: You tried to select a unit that has already been selected.

User Action: Use the PARAMETERLIST command to display information on all units that are selected.

IOX—TTuu: is logged in

Explanation: A user is logged in on the terminal selected.

User Action: Select a terminal that is not in use.

IOX—dduu: is not configured

Explanation: You issued a command that is valid only for configured devices.

User Action: Use the CONFIGURE command to configure the device.

IOX—dduu: is not mounted or allocated for NFS operations

Explanation: You have selected a device for testing with SELECT or VERIFY that is not properly allocated and mounted for NFS testing.

User Action: Allocate and mount the device with the /FOREIGN switch.

IOX—dduu: is not ready—unit now deselected

Explanation: IE.DNR error code returned to IOX (device not ready).

User Action: Select the device again.

IOX—dduu: is not selected

Explanation: You issued a device-dependent parameter command (BADBLOCKS or RANGE) that is valid only after you have selected the device.

User Action: Select the device; then, reenter the command.

IOX—dduu: is offline

Explanation: The driver returned the error code IE.OFL to IOX, which indicates that the device is off line. Either the device was taken off line while testing was in progress, or you tried to select a device that was off line.

If testing was in progress on the device when it was taken off line, IOX appends the following string to the error message:

```
- unit now deselected
```

User Action: Ensure that the device is on line.

IOX—dduu: is write locked—unit now deselected

Explanation: The driver returned the error code IE.WLK IOX, which indicates that the device is write-locked.

User Action: Check the write-lock/write-enable switch located on the device drive. Also, check the write-lock status for the device.

IOX—dduu: LUN assignment failure

Explanation: Either the device ddnn is not known to the system or you have exceeded the maximum number of devices that IOX supports. The default is 28 devices; the maximum number of devices is 246.

User Action: Modify the system configuration to include the device in your system (see Chapter 4) or edit the Task Builder (TKB) command file, IOXBLD.CMD to allow IOX to support more devices (see Section 12.1).

IOX—dduu: privilege violation—unit now deselected

Explanation: The dirver returned the error code IE.PRI to IOX, which indicates a privilege violation. Either the device was not allocated or it was mounted by someone else.

User Action: Be sure you have allocated and mounted the device before you begin the I/O exercise.

IOX—dduu: specified device type is inconsistent

Explanation: You tried to configure device dduu as a tape when a dd-type device had already been configured as a disk. This error also occurs if you configure dd as a tape and then attempt to configure another dd-type device as a disk.

User Action: Use the CONFIGURE command as a display command to check on the devices in your test configuration. Configure the unit with a different device mnemonic.

IOX—dduu: unexpected error #xx

Explanation: IOX received an I/O error code that it could not process. The error code is the octal number xx.

User Action: No user action is required.

IOX—dduu: unexpected request to queue I/O—request aborted

Explanation: IOX received an asynchronous system trap (AST) from a device after it deselected the device.

User Action: Check the device.

IOX—dduu: unrecoverable error at block xxx (decimal) xxx (octal)

Explanation: The driver returned the error code IE.VER to IOX, which indicates an unrecoverable error.

User Action: Check the device drive and the media.

IOX—dduu: VOLUMECHECK failure. Unit contains a home block

Explanation: The unit is file-structured because the device contains a home block. This error message is generated by the SELECT command if you try to select a Files–11 volume while VOLUMECHECK is enabled.

User Action: IOX destroys information on devices that you select for exercising with the SELECT command. If you want to destroy the contents of the disk, disable VOLUMECHECK (VOLUMECHECK NO) and select the device again.

IOX—dduu: write check error at block xxx (decimal) xxx (octal)

Explanation: The driver returned the error code IE.WCK IOX, which indicates a write-check error. The write-check operation failed at block xxx.

User Action: No user action is required.

12.8.2 Error Messages Without Device Specifications

This section lists IOX error messages in alphabetical order, according to the first letter that follows the IOX task name.

IOX—Indirect command file syntax error

Explanation: There is an error in the indirect command file.

User Action: Check the indirect command file.

IOX—Indirect command file nesting level exceeded

Explanation: IOX accepts indirect command files with only one level of nesting.

User Action: Rewrite the indirect command file so that it does not have more than one level of nesting.

IOX—Aborting due to user command

Explanation: You aborted IOX by using the MCR command ABORT.

User Action: To continue testing, reinvoke IOX.

IOX—EXECUTE command is not supported

Explanation: You attempted to use the EXECUTE command in a system that does not support task spawning.

User Action: Include support for task spawning during system generation.

IOX—Command input I/O error

Explanation: IOX detected an error when it tried to read the command that you typed at the terminal.

User Action: Reenter the command.

IOX—Command valid only in Command Mode

Explanation: You tried to enter a command that cannot be entered while an I/O exercise is in progress.

User Action: Either abort the exercise and enter the command or wait until the exercise has finished to enter the command.

IOX—Command valid only in Interactive Mode

Explanation: You tried to enter a command that cannot be entered unless IOX is running.

User Action: Enter the START command to run IOX; then, reenter the command.

IOX—Command not unique

Explanation: The command abbreviation that you entered was not unique.

User Action: Reenter the command with enough characters to distinguish it from other IOX commands.

IOX—Failed to close log file. F.ERR= -x

Explanation: IOX could not close the log file in your directory. The File Control Services (FCS) error returned was -x.

User Action: For a description of the I/O error code, see the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual.* Then, take the appropriate action.

IOX—Failed to open indirect command file

Explanation: The FCS routine GCML failed to open the indirect command file.

User Action: Check to see that the file exists and that you specified the file correctly; then, resubmit the file to IOX.

IOX—Failed to open log file. F.ERR= -x

Explanation: IOX could not open the log file. The FCS error returned was -x.

User Action: The disk may be write-protected or may not have enough blocks. For a description of the I/O error code, see the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual.* Then, take the appropriate action.

IOX—Failed to open existing log file. F.ERR= -x

Explanation: IOX could not open a log file that already exists in your directory. The FCS error returned was -x.

User Action: For a description of the I/O error code, see the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual.* Then, take the appropriate action.

IOX—Failed to truncate/close log file. F.ERR= -x

Explanation: IOX could not truncate or close the log file. The FCS error returned was -x.

User Action: For a description of the I/O error code, see the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual.* Then, take the appropriate action.

IOX—Fatal file system error. F.ERR= xxx PC+2= xxx

Explanation: IOX exited because an error occurred while IOX was trying to issue a read or write. F.ERR is the octal error code in the File Descriptor Block (FDB) indicating the file error type. PC+2 is the octal location of the failure in the IOX code.

User Action: For a description of the I/O error code, see the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual.* Then, take the appropriate action.

IOX—Fatal system error. DSW= xxx PC+2= xxx

Explanation: IOX immediately exits because a directive that IOX issued failed. DSW is the Directive Status Word in octal. PC+2 is the octal location of the failure in the IOX code.

User Action: For a description of the I/O error code, see the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual.* Then, take the appropriate action.

IOX—Invalid pattern number

Explanation: You tried to set a data pattern by using a pattern number greater than 12. The valid arguments for PATTERN are 0 to 12 (inclusive).

User Action: Enter a valid pattern number.

IOX—Invalid task name

Explanation: You specified an invalid task name with the EXECUTE command. Either the task name is incorrect or the task does not exist.

User Action: Reenter the task name correctly or determine if the task exists.

IOX—No activity remains—aborting

Explanation: In Execution Mode, IOX determines whether or not any exercises are in progress or any tasks are executing at 1-minute intervals. If IOX finds no activity, it aborts the exercise and generates this informational message.

In Interactive Mode, IOX determines whether or not any exercises are in progress or whether any tasks are executing only when you enter the PROCEED command. If there is no activity when you enter PROCEED, IOX aborts the exercise and issues this informational message.

User Action: No user action is required.

IOX—No activity to start

Explanation: When you issued the START command, no devices were selected for testing and no tasks were initiated for execution.

User Action: Select devices that you want to test, and use the EXECUTE command to initiate tasks that you want to execute.

IOX—No buffer space available

Explanation: IOX does not have adequate buffer space to accommodate all the devices you selected.

User Action: Either deselect some devices from testing or reinstall or reinvoke IOX with a greater increment.

IOX—No such command—type H for help

Explanation: You entered a command that IOX did not recognize.

User Action: Type H (abbreviation for the HELP command), which displays a summary all IOX commands.

IOX—No tasks to abort

Explanation: You tried to abort tasks by using the ABORT command when no tasks had been initiated by the EXECUTE command.

User Action: No user action is required.

IOX—Only CTRL/C allowed

Explanation: You set CONTROL_C YES to enable you to enter IOX commands while exercises are in progress. Therefore, IOX only accepts CTRL/C in Execution Mode.

User Action: Press CTRL/C to exit Execution Mode; then, IOX is in Interactive Mode and you can enter commands.

IOX—Syntax error

Explanation: The command line that you entered contained a syntax error.

User Action: If necessary, check the syntax of the command line that you entered. See Section 12.6 or type H (for HELP).

IOX—Task never executed

Explanation: The task that you tried to abort is not on the list of tasks initiated by the EXECUTE command.

User Action: Use the PARAMETERLIST command to check the list of tasks to be executed.

IOX—Task "xxxxxx" is currently active

Explanation: You tried to execute a task that was already active. The name of the task is xxxxxx.

User Action: No user action is required.

IOX—Task "xxxxxx" is completed. Status= nnnnnnn

Explanation: The task has completed execution. The name of the task is xxxxxx. The task status represented by nnnnnnn is one of the following: SUCCESS, WARNING, ERROR, and SEVEREERROR.

User Action: No user action is required.

IOX—Task "xxxxxx" is not installed

Explanation: Task "xxxxxx" is not installed and cannot be initiated by using the EXECUTE command.

User Action: Install the task from a privileged terminal.

IOX—Task "xxxxxx" is unable to execute

Explanation: Task "xxxxxx" could not execute.

User Action: Try to execute the task again.

IOX—Units are selected—command ignored

Explanation: The BUFFERSIZE command is valid only if no units are selected for testing.

User Action: You can use the BUFFERSIZE command within the SELECT and VERIFY command lines to override the default for each device. However, if you require a larger buffer size for all devices, use the DESELECT command to deselect all units; then, reissue the BUFFERSIZE command.

Chapter 13

The Bad Block Replacement Control Task

This chapter describes bad block handling and the Bad Block Replacement Control task (RCT). It begins with an introduction to RCT, which is followed by information on installing RCT and loading the driver for DU-type devices (DUDRV). The chapter continues with a description of how the RCT task performs bad block replacment. It concludes with a brief summary of the interaction between RCT and the Error Logger.

13.1 Introduction to RCT

The Bad Block Replacement Control Task (RCT) handles bad block replacement and recovery on Mass Storage Control Protocol (MSCP) disks such as the RA80. However, RCT does not replace the Bad Block Locator utility (BAD); you must run BAD on MSCP devices to create a bad block descriptor file before you initialize the disk. BAD locates bad blocks and lists them in a bad block descriptor file. To include information in the bad block descriptor file about blocks to be detected by the device controller and to be replaced by RCT, specify the BAD switch /RETRY. (For more information on BAD, see the *RSX-11M-PLUS Utilities Manual.*)

Note

> Utilities such as BAD, the Backup and Restore Utility (BRU), and the INI task, treat MSCP devices as non-last-track devices. For more information, see the *RSX-11M-PLUS Utilities Manual.*

13.2 Installing RCT and Loading DUDRV

If you are using a conventional RSX-11M-PLUS system (that is, a system that has not been pregenerated) and you select a DU-type device during system generation, the system generation procedure builds the RCT task and the DUDRV driver for you. Then, when you create the system image with the Virtual Monitor Console Routine (VMR), the command procedure SYSVMR.CMD installs RCT and loads DUDRV. If you have a pregenerated RSX-11M-PLUS or Micro/RSX operating system, the system startup procedure includes support for RCT by default.

However, you may need to load or install and then load RCT and DUDRV under the following conditions:

- You have removed RCT, unloaded DUDRV, or both.
- You have edited the file SYSVMR.CMD to exclude RCT, DUDRV, or both.

If you plan to load DUDRV manually, you must install RCT first. If RCT is not installed, you will receive an error message at the console terminal.

To install RCT, use the DIGITAL Command Language (DCL) or Monitor Console Routine (MCR) command INSTALL, as follows:

```
DCL>
```

```
MCR>
```

To load DUDRV, use the LOAD command, as follows:

```
>
```

13.3 Bad Block Handling

Bad block handling is performed by either the RCT task or the disk controller, depending on the type of device being used. Table 13-1 summarizes the relationship between device type, controller type, and the method of bad block handling.

Table 13-1: Bad Block Replacement Method for Different Device Types

Devices	Controllers	Method
RA80/81/60	UDA50 KDA50	RCT replaces bad blocks.
RC25	KLESI-U KLESI-Q	RCT replaces bad blocks.
RD51/52/53/54	RQDX1 RQDX2 RQDX3	Controller replaces bad blocks.
RX50	RQDX1 RQDX2 RQDX3 RUX50	No replacement.
RX33	RQDX3	No replacement.

The following subsections describe the methods of bad block handling in more detail.

RCT Performs Bad Block Replacement

If the RCT task performs the bad block replacement, bad block handling involves the following sequence of steps:

- Detecting the bad block
- Notifying the device driver DUDRV
- Replacing the bad block (using RCT)
- Revectoring I/O from the bad block to the replacement block

The disk controllers handle the detection of bad blocks and the notification of the driver (DUDRV). After the driver is notified, it activates RCT, which performs all the bad block replacement functions. After the bad blocks have been replaced, the controller dynamically revectors (that is, redirects) I/O from the bad block to the replacement block.

RCT also performs replacement and recovery on disks that went off line during bad block replacement or before the contents of a write-back cache were copied to the disk (see Section 13.3.3).

Controller Performs Replacement

If the controller (RQDX-series) performs the bad block replacement, bad block handling involves the following sequence of steps:

1. Detecting the bad block
2. Replacing the bad block (using the controller)
3. Revectoring I/O from the bad block to the replacement block
4. Notifying the device driver DUDRV

For RD-series devices, the controller handles the detection and replacement of bad blocks. After the bad blocks have been replaced, the controller dynamically revectors (that is, redirects) I/O from the bad block to the replacement block. When the revectoring is complete, the controller notifies DUDRV.

The following subsections describe each aspect of bad block handling in more detail.

13.3.1 Detecting Bad Blocks—The MSCP Disk Configuration

To perform bad block replacement functions, RCT uses the following address spaces on MSCP disks:

- Logical block numbers (LBNs)
- Replacement block numbers (RBNs)
- Revector control tables

The subsections that follow describe these addresses.

Logical Block Numbers

LBNs are accessible to user tasks for transferring data between the disk and the operating system. When the controller encounters an invalid LBN and notifies DUDRV, DUDRV activates RCT to replace the LBN.

Replacement Block Numbers

RBNs are blocks reserved throughout the disk that RCT allocates as substitutes for bad LBNs. When a disk is formatted, a forced error bit is placed in the header of each replacement block. This prevents a bad block from accidentally being revectored to a good block that is already in use.

RBNs cannot be directly accessed by user tasks during normal data transfers.

Revector Control Tables

RCT and the device controllers use the DUDRV driver to access the revector control tables. Each table entry corresponds to an RBN; the address of the RBN depends upon the location of the entry in the table. An entry contains the LBN being replaced and a code describing the status of the RBN. The status codes are as follows:

Status Code	Definition
ALLOCATED	Indicates that the RBN is currently being used.
UNALLOCATED	Indicates that the RBN is not being used.
UNUSABLE	Indicates that the RBN cannot be used.
PRIMARY	Identifies the first RBN read when a bad block is replaced.
SECONDARY	Identifies the RBN read after the primary blocks are read.

For a given LBN, a primary RBN is in a predetermined location; all other RBNs are secondary. The revector control tables store information in a flag word that enables RCT to determine if recovery techniques are required when a disk is brought on line.

13.3.2 Notifying DUDRV

The controller may detect a bad LBN while processing an I/O request from the driver. The controller notifies DUDRV by listing the bad LBN in an end packet. DUDRV then stores this information in pool and activates RCT.

DUDRV provides RCT with the bad LBN, the I/O packet address, and the Unit Control Block (UCB) address.

13.3.3 Replacing Bad Blocks

The RCT task performs the bad block replacement. It stores data from the bad block, allocates and initializes the replacement block, and then updates data structures on the disk.

After receiving information from DUDRV, RCT performs the following steps in bad block replacement:

1. Makes the disk device appear as though it's attached, which inhibits all I/O requests to the device (except the I/O necessary for bad block replacement).
2. Reads the data from the bad LBN into a buffer in its data space.
3. Determines which RBN on the disk to use by applying a predetermined algorithm, which ensures that the block was revectored correctly.
4. Issues QIO$ requests to DUDRV that update the revector control tables (see Section 13.3.4).
5. Updates the header and data of the bad LBN by issuing the I/O function code IO.RPL to DUDRV. The IO.RPL code directs the controller to indicate that the LBN has been replaced either by a primary or secondary RBN.
6. Initializes the RBN by writing back the data read from the LBN. Because the bad LBN has been replaced by the RBN, the data is written transparently to the RBN by the controller.

   If an RBN is bad, RCT treats it as a bad LBN. RCT finds another RBN, updates the revector control tables, marks the old RBN as unusable, and initializes the new RBN by writing the stored data back to the bad LBN.
7. Detaches the unit. Normal I/O to the disk resumes.

During bad block replacement, RCT uses a recursion counter. If the counter identifies three consecutive bad disk blocks, it aborts the search operation and will not replace the block. Subsequently, if the problem causing the errors is in the data line (thus affecting all I/O), the recursion counter prevents RCT from marking all replacement blocks as bad.

RCT also checks MSCP disks that have just come on line for two conditions: incomplete bad block replacement, and write-back cache corruption. However, because RSX–11M–PLUS and Micro/RSX operating systems do not support disks with write-back caching, write-back cache corruption is extremely rare. RCT performs this function only under the following circumstances:

- A disk with a write-back cache failed to copy the data from the write-back cache to the disk before going off line, while on another operating system (which supports write-back caching), and was then moved to an RSX–11M–PLUS operating system.
- The revector control tables have been altered such that the write-back cache-in-use flag is set.

RCT handles these conditions in the following ways:

- If RCT determines that bad block replacement was partially completed when the disk went off line, RCT completes the bad block replacement process.
- If RCT determines that the write-back cache was not copied to the disk before the disk went off line, RCT software write-locks the disk so that the contents of the write-back cache are preserved.

The following subsections describe these procedures in more detail.

Completing Partial Bad Block Replacements

When a disk unit is brought on line, RCT checks the revector control table to be sure that no bad block replacement was in progress when the disk went off line. If replacement needs to be completed, RCT completes the bad block replacement before allowing the disk to be brought on line.

Preventing Loss of Write-Back Cache Data

RCT prevents loss of write-back cache data.

A disk with the write-back-cache-in-use flag set will be write-locked each time it is brought on line. To write-enable such a disk, use the /WRITE qualifier with the DCL command MOUNT (MCR command MOU) each time you mount the disk. RCT checks the write-back-cache-in-use flag of the revector control tables, software write-locks the disk if the flag is set, and writes the following Task Termination Notification program (TKTN) message to the console output device (CO:):

```
ddnn: -- Write back caching data lost. Unit write locked
```

The device specification ddnn is the disk that RCT has write-locked.

13.3.4 Bad Block Revectoring

Revectoring is performed dynamically by the device controller. When the controller first accesses an LBN that has been replaced by RCT, the controller revectors (redirects) the I/O to the RBN that RCT designated as the replacement for the LBN. Thereafter, I/O to the bad LBN is revectored transparently to the RBN; DUDRV does not need to activate RCT.

When the controller detects a bad LBN that has been replaced by a primary RBN, the controller accesses the RBN directly.

When the controller detects a bad LBN that has been replaced by a secondary RBN, the controller must search for the address of the secondary RBN before it can access the primary RBN. The controller locates the address in one of the following two ways:

- Checking the data field of the bad LBN
- Checking the revector control tables for the entry containing the bad LBN

After the controller locates the address of the secondary RBN, it accesses the primary RBN and performs the revectoring.

13.4 RCT and the Error Logger

DUDRV notifies the Error Logger with an error log packet whenever an error has been detected that requires bad block replacement. RCT also generates an error log packet indicating that a bad LBN has or has not been successfully replaced by an RBN. RCT logs the packet under the name of the task that issued the Executive directive Queue I/O request (QIO$), which resulted in the detection of the bad block by the controller.

For more information on Error Logging, see the *RSX–11M–PLUS and Micro/RSX Error Logging Manual.*

Chapter 14

I/O Queue Optimization

I/O queue optimization increases the number of I/O requests processed by the system in a given amount of time. I/O processing is improved by using the list of I/O requests in a queue more effectively. The I/O queue for a particular disk device is scanned by I/O queue optimization. Then, after each request is examined, the best request is removed from the queue and passed to the I/O driver for processing. By selecting from different methods of I/O queue optimization, you decide how I/O requests are selected.

This chapter describes I/O queue optimization and how you can initialize, display, and select different methods for using it. The chapter also includes a list of associated error messages.

14.1 Overview

RSX-11M-PLUS and Micro/RSX operating systems support the following methods of I/O queue optimization:

- Nearest Cylinder
- Elevator
- Cylinder Scan

The method that is best for your application depends on your processing environment. (For more information on each method, see Section 14.3.)

You can also choose not to use I/O queue optimization. I/O queue optimization works by optimizing requests in the queue; if the load on your system does not require frequent use of the queue, little optimization is achieved. If you do not use I/O queue optimization, the operating system groups the I/O requests in the queue by priority. Then, requests are processed on a first-in/first-out (FIFO) basis. The highest priority requests appear first in the queue and are processed in sequence.

If you use I/O queue optimization, the I/O requests within priority groups are examined. The request having the "appropriate" disk address is chosen as the next I/O operation. The optimization method you select determines what address is appropriate. The highest priority requests are still serviced first; however, I/O requests are processed more rapidly because requests are reordered within a priority.

I/O service time depends upon three components: seek-time, latency, and data transfer time. *Seek-time* is the amount of time the disk device spends searching for an area on a disk. *Latency* refers to the delay between seek-time and the actual *data transfer time.* All three methods of I/O queue optimization attempt to minimize the seek-time component of the data I/O service time. In some cases, I/O queue optimization may provide smoother disk operation by reducing erratic head movement.

14.2 Initiating Optimization

The Monitor Console Routine (MCR) command SET /OPT is used to initiate I/O queue optimization. The parameters in this command are used to specify the disk device, the method of optimization, and a fairness count limit for I/O requests that are passed over. These parameters are explained in detail in later sections.

The SET command is for privileged users. However, nonprivileged users can use this command to display the optimization status of devices in the system.

Format:

```
SET /[NO]OPT[=ddnn:opttype:fairnesscount.]
```

Parameters

ddnn

Specifies the device mnemonic (dd) and unit number (nn) for the disk device requiring optimization. The following device types support I/O queue optimization:

Mnemonic	Device
DBn	RP04, RP05, and RP06 pack disks
DRn	RM02, RM03, RM05, RM80, and RP07 pack disks
DMn	RK06 and RK07 cartridge disks
DLn	RL01 and RL02 cartridge disks
DUn	MSCP disks

opttype

Specifies one of the following optimization methods:

NEAR[EST] Initiates the Nearest Cylinder method; the next request processed is the one closest to the current request.

ELEV[ATOR] Initiates the Elevator method; requests are processed first in one direction along the disk and then in the reverse direction.

CSCAN Initiates the Cylinder Scan method; requests are processed in only one direction along the disk (lowest to highest).

The default is NEAREST.

fairnesscount

Specifies a limit on the number of times the most unsatisfactory I/O request is passed over. The maximum fairness count allowed is 128_{10}.

For more information on setting the fairness count, see Section 14.3.1.

The default is 10_{10}.

Qualifiers

/NOOPT[=ddnn:]

Disables disk I/O queue optimization for the specified device. When you omit =ddnn:, the system displays all the devices that do not use I/O queue optimization.

/OPT[=ddnn:opttype:fairnesscount.]

Enables disk I/O queue optimization for the specified device. When you omit =ddnn:, the system displays all the devices that do not use I/O queue optimization. (See Section 14.2.1.)

14.2.1 Displaying Optimization Status

You can display the optimization status for all devices in the system that are currently using I/O queue optimization by entering the MCR command SET /OPT, as follows:

```
>
```

The SET /OPT command displays status in the following format:

```
OPT=ddnn:opttype:fairnesscount
```

You can also display the devices that are not currently using I/O queue optimization by entering the command SET /NOOPT:

```
>
```

The SET /NOOPT command displays status in the following format:

```
NOOPT=ddnn:
```

Both nonprivileged and privileged users can display the I/O queue optimization status. For more information on the SET /OPT and SET /NOOPT commands, see the *RSX-11M-PLUS MCR Operations Manual.*

Examples

```
OPT=DB0:NEAR:10.
OPT=DB1:ELEV:5.
OPT=DB2:CSCAN:8.
```

Displays the optimization status for the devices DB0, DB1, and DB2.

```
NOOPT=DR1:
```

Displays the device DR1. DR1 is not using I/O queue optimization.

14.3 Selecting the Type of Optimization

Selecting the correct optimization method is dependent upon the I/O processing environment in which your application is running, the physical location of data on the disk drive, and how often tasks access certain data areas.

A general guideline is to try the default first. (This initiates the Nearest Cylinder method with a fairness count limit of 10.) Then, compare the difference in the rate of I/O processing or *throughput* by recording the amount of time it takes for your application to run. For a closer look at disk activity and throughput, use Resource Accounting (see Chapter 10). Resource Accounting provides this information under the heading DEVICE STATISTICS in the output from the Resource Accounting transaction file.

The default (NEAREST, and a fairness count of 10) is an average setting and should improve I/O throughput in most cases. As you become more familiar with the I/O load and data organization on the disk in your application, you can make adjustments by changing the fairness count limit or by using one of the other methods of I/O queue optimization.

14.3.1 Adjusting the Fairness Count Limit

When you use I/O queue optimization, you may notice increased waiting time. I/O requests to some data areas may have slower than normal access times because I/O queue optimization categorizes them as unsatisfactory. When this happens, the optimization method passes over these requests for a better choice, keeping the unsatisfactory requests in the queue longer. This may cause a decrease in performance for some tasks (even though overall system performance improves).

To counteract increased waiting time and to acquire a better balance, you can adjust the fairness count limit. This places a limit on how many times an I/O request can be passed over. Optimization increments the fairness count for the first I/O request in the queue whenever that request is passed over. When the fairness count for an I/O request reaches the limit indicated with the SET command, the request is removed from the queue and the fairness count is cleared.

Decreasing the fairness count limit reduces waiting time but decreases the overall throughput initially gained with the higher count. As an alternative to decreasing the fairness count limit, use another optimization method. It may be more applicable to your situation.

14.3.2 Nearest Cylinder Method

The Nearest Cylinder optimization method processes the I/O request that is closest to the current cylinder. In general, this method is effective for both long and short I/O queues. However, there may be an increase in waiting time, which may be unacceptable in an interactive environment. To remedy this, reduce the fairness count limit (which reduces the throughput) or try one of the other optimization methods. (The Elevator method may be the next best choice.)

14.3.3 Elevator Method

The Elevator optimization method processes I/O requests like an elevator. It processes requests as it moves in one direction along the disk until it reaches the last request in that direction. The Elevator method then changes direction and processes requests in that direction. This method is effective if there are many I/O requests outstanding. Also, waiting time tends to be less with the Elevator method than with the Nearest Cylinder method.

The Elevator method favors the center of the disk. Requests are done as they are passed, and the center of the disk tends to be passed over twice as often as the periphery. Therefore, a task that has a number of requests located in the center of the disk tends to have faster I/O than a task with requests at the periphery.

14.3.4 Cylinder Scan Method

The Cylinder Scan optimization method operates like the Elevator method, but it processes I/O requests only while passing in one direction along the disk. The direction is from the lowest cylinder number to the highest cylinder number. This method is most effective if each task tends to do I/O to a localized area on the disk. It results in a more even distribution of I/O among tasks.

14.4 Error Messages

SET—Privileged command

Explanation: If you are a nonprivileged user, you cannot set I/O queue optimization for a device. Nonprivileged users can only use the SET /OPT command to display the optimization status.

User Action: For more information on displaying the optimization status for a device, see Section 14.2.1.

SET—Feature not supported

Explanation: You tried to initiate optimization for a disk device that does not support I/O queue optimization.

User Action: For a list of devices that support I/O queue optimization, see Section 14.2.

SET—Device not in system

Explanation: You tried to initiate optimization for a disk device that is not mounted in the system.

User Action: Use the DCL command SHOW DEVICES (or the MCR command DEV) to obtain a list of mounted devices. Select a disk that is already mounted or mount the disk; then, initiate optimization.

SET—Illegal keyword value

Explanation: You specified an invalid value for the fairness count. The maximum fairness count allowed is 128_{10}.

User Action: Specify a fairness count less than 128_{10}.

SET—Invalid keyword

Explanation: You specified an invalid keyword.

User Action: For a list of valid keywords for the SET command, see the *RSX-11M-PLUS MCR Operations Manual* or the *RSX-11M-PLUS Command Language Manual.*

SET—Device offline or privileged diagnostic

Explanation: You tried to initiate I/O queue optimization for an offline device. You may have specified the wrong device.

User Action: Be sure you are using the correct device specification. If you have an RSX-11M-PLUS system, use the Reconfiguration Services to bring the device on line (see Chapter 4). If necessary, mount the device. Then, reenter the SET /OPT command.

Chapter 15
Disk Data Caching

Disk data caching enhances I/O operations by reducing the number of physical I/O requests directed to a disk.

The Data Cache Manager (DCM) monitors disk data cache operations by acting as an intelligent preprocessor to the disk drivers. In addition to monitoring all I/O operations performed on disk devices, the DCM decreases the number of physical I/O operations taking place on "cached" disk drives by using a designated storage area (or "cache") in memory. Copies of disk data are placed in the cache region, making the data available for memory-to-memory transfers instead of disk-to-memory transfers during an I/O request.

DCM is enabled by default on Micro/RSX systems and on pregenerated RSX–11M–PLUS systems. You can also enable DCM by selecting data caching as a system generation option. If you select the Full-functionality Executive, DCM is enabled by default.

The system provides the following Virtual Monitor Console Routine (VMR) commands in SYSVMR.CMD to permanently include disk data caching as a directive common in the bootstrapped system:

```
INS DCM11M/RON=YES
FIX DCM11M/DIR
```

15.1 Associating Data Caching with a Disk

Although data caching is enabled during system generation, it is not associated with any disk volumes. To activate data caching, you must associate it with the volumes that you want to be cached.

15.1.1 Using the MOUNT Command

The DIGITAL Command Language (DCL) command MOUNT and Monitor Console Routine (MCR) command MOU makes a disk volume available for processing. To associate caching with a selected volume, use the /CACHE qualifier with the MOUNT (MOU) command.

Format

```
DCL>MOUNT ddnn:volumelabel /[NO]CACHE[:(option,option,...)]
```

```
MCR>MOU ddnn:volumelabel /[NO]CACHE[:(option,option,...)]
```

If you specify one option only, the parentheses are optional.

Parameters

ddnn

Specifies the device on which you are mounting the volume. Note that the device must be a mass-storage device (DV.MSD); DCM does not execute with tape devices (such as DV.SQD).

volumelabel

Specifies the name that is associated with the volume.

Qualifiers

/NOCACHE

Specifies that caching is not associated with the device being mounted. /NOCACHE is the system default.

/CACHE

Associates caching with the specified disk and allows you to override the disk data caching defaults and to set one or more of the options.

The following options are available with the MOUNT (MOU) command qualifier /CACHE:

CREATE[:[region][:[main_partition][:[size]]]]
REGION:name
[NO]DEFER_WRITES
[NO]DIRECTORY[:extent_size]
[NO]LOGICAL[:extent_size]
[NO]OVERLAY[:extent_size]
[NO]READ_AHEAD[:extent_size]
[NO]VIRTUAL[:extent_size]

You cannot activate caching for a device unless a cache region exists. Therefore, when you use the MOUNT command to associate caching with a disk volume for the first time, the CREATE option is required.

The extent_size argument specifies the maximum size of an I/O request that is considered for caching. The extent size is a decimal number. The minimum value for an extent size is 1 block; the maximum size is 127_{10} blocks.

For a desription of the CACHE and REGION options, see Section 15.1.3. For a description of the other options, see Section 15.1.4.

15.1.2 Using the SET DEVICE Command

The DCL command SET DEVICE (or MCR command SET) associates caching with a disk volume that has been mounted with the MOUNT (MOU) /NOCACHE command. If the volume already has caching associated with it (that is, if you mounted the volume with the MOUNT /CACHE command), then the SET DEVICE (SET) command modifies the current data caching options.

The first time you use the SET DEVICE (SET) command to associate caching with a device, you set the default values for each of the data caching options. Subsequent SET DEVICE (SET) commands change only the options specified in the command line. All other options retain the values that you set when you associated caching with the device (using either the MOUNT [MOU] or SET DEVICE [SET] command).

Note

> Prior to Version 4.0 of the RSX-11M-PLUS and Micro/RSX operating systems, the DCL command SET DEVICE (and MCR command SET) used the system default values for any options that you did not specify in the command line.

For more information on modifying data cache operations, see Section 15.3.

Formats

```
DCL>SET DEVICE ddnn: /[NO]CACHE[:(option,option,...)]
```

```
MCR>SET /[NO]CACHE[:ddnn:[(option,...)]]
```

If you specify one option only, the parentheses are optional.

Parameter

ddnn

Specifies the device you are associating with caching. If the device is already associated with caching (that is, if you have already mounted or have set the device cached), then ddnn specifies the device you are modifying.

The device specification ddnn can also be a logical name assigned to the device.

Qualifiers

/NOCACHE

Deactivates caching for the specified device.

With the SET command, if you do not specify a device, /NOCACHE lists devices with disk data caching disabled.

You cannot specify any options with the /NOCACHE qualifier.

/CACHE

Enables caching or modifies the caching behavior for the specified device.

With the SET command, if no device is specified, /CACHE lists devices with disk data caching enabled.

You can enable or modify the following options:

```
CREATE[:[region][:[main_partition][:[size]]]]
REGION:name
[NO]DEFER_WRITES
[NO]DIRECTORY[:extent_size]
[NO]LOGICAL[:extent_size]
[NO]OVERLAY[:extent_size]
[NO]READ_AHEAD[:extent_size]
[NO]VIRTUAL[:extent_size]
```

The first time you enter the SET /CACHE command for a device, the CREATE option is required. To associate caching with a disk volume, you must create a cache region. However, if you modify the data caching attributes for the same device with subsequent SET commands, the CREATE option is optional.

The extent_size argument specifies the maximum size of an I/O request that is considered for caching. The extent size is a decimal number. The minimum value for an extent size is 1 block; the maximum size is 127_{10} blocks.

For a desription of the CACHE and REGION options, see Section 15.1.3. For a description of the other options, see Section 15.1.4.

To associate caching with a disk that you mounted with the MOUNT /NOCACHE command, use the SET DEVICE/CACHE (or SET /CACHE) command. Likewise, to disassociate caching from a disk that you mounted with the MOUNT /CACHE command, use the SET DEVICE /NOCACHE (or SET /NOCACHE) command.

15.1.3 Specifying a Cache Region

To associate caching with a disk volume, you must create a cache region.

To create a cache region, use the CREATE option with the MOUNT /CACHE (MOU /CACHE) or SET DEVICE /CACHE (SET /CACHE) command. To name (or rename) a region, use the REGION option.

The CREATE and REGION options are defined as follows:

CREATE[:[region][:[main_partition][:[size]]]]

Creates a cache region and associates caching for the specified device with that region. Specify the following arguments with the CREATE option:

region
: Names the cache region. The default is CACHE.

  To use a cache region other than the default (CACHE) provided by the system, specify a different name when you create the region.

main_partition
: Names the main partition for data caching. The default is GEN.

  In general, RSX-11M-PLUS and Micro/RSX systems define one main partition for general use. This partition, named GEN by default, may contain many regions. These regions can be commons, Extended Executive Partitions (EXPs), and task images.

size
: Specifies the number (in decimal) of disk blocks for the cache region. The default is 100_{10} disk blocks.

  Note that you can change the size of a cache region only when you create it.

REGION:name

Associates caching for the specified device with an already existing cache region (that is, a region that you previously created with the CREATE option).

You can specify a region to be used in place of the system default. The default region is CACHE.

15.1.4 Specifying Different Types of Caching

Disk data caching is available for various types of I/O operations. Caching of I/O requests is most effective when blocks of data are read repeatedly or are accessed sequentially in small increments. Thus, caching depends on the size of the request. If a request exceeds the specified (or default) extent size, it is not cached. (This prevents any single type of I/O from monopolizing the cache buffers.)

To enable or disable data caching for different types of I/O operations, use one or more of the SET DEVICE /CACHE (SET /CACHE) or MOUNT /CACHE (MOU /CACHE) command options. Note that you can specify each option for each device unit independently.

The following options are available:

DEFER_WRITES
NODEFER_WRITES

Enables or disables the deferred-write request attribute of data caching for temporary files. *Temporary files* are files that have been marked for deletion when all associated tasks close the file. These files do not need to be maintained across a system failure. Therefore, write requests to these files can be written into the cache at the time of the request and deferred until a later time to the disk.

Deferral of write requests speeds the completion time of I/O requests by postponing potentially long disk accesses. This operation is beneficial for disks with slow access time.

DIRECTORY[:extent_size]
NODIRECTORY

Enables or disables caching of directory I/O. *Directory I/O* consists of all I/O operations issued by an Ancillary Control Processor (ACP) task when context switching is enabled. Typically, caching of directory I/O is useful for files such as [0,0]BITMAP.SYS, [0,0]INDEXF.SYS, and the directory files (file type DIR).

The system default for this option is DIRECTORY; caching of directory I/O is enabled. In addition, the default extent size for DIRECTORY is 1 block. Depending on your system usage, an extent size of 3 blocks may be more effective.

LOGICAL[:extent_size]
NOLOGICAL

Enables or disables caching of logical I/O. *Logical I/O* requests consist of all Read Logical Block (IO.RLB) and Write Logical Block (IO.WLB) I/O functions that are not issued by an ACP. Typically, tasks that use logical I/O do not benefit from data caching. For example, the Backup and Restore Utility (BRU) reads file extents randomly throughout the disk. Because caching is most effective when blocks of data are accessed repeatedly or sequentially in small amounts, it would not benefit this type of BRU operation.

The system default for this option is NOLOGICAL; caching of logical I/O is disabled.

The default extent size for LOGICAL is 1 block.

OVERLAY[:extent_size]
NOOVERLAY

Enables or disables caching of overlay I/O. Caching of I/O overlay load requests (IO.LOD and IO.LOV) depends on the size of the load. If a load request exceeds the specified (or default) extent size, the request is not cached. This prevents overlay loads from monopolizing the cache buffers.

The system default for this option is OVERLAY; caching of overlay I/O is enabled. The default extent size for OVERLAY is 4 blocks.

VIRTUAL[:extent_size]
NOVIRTUAL

Enables or disables caching of virtual I/O. *Virtual I/O* consists of all Read Virtual Block (IO.RVB) and Write Virtual Block (IO.WVB) I/O functions. When context switching is disabled, requests issued by an ACP are also considered virtual I/O. (This characteristic distinguishes them from directory I/O.)

The system default is VIRTUAL; caching of virtual I/O is enabled. The default extent size for VIRTUAL is 5 blocks.

READ_AHEAD[:extent_size]
NOREAD_AHEAD

Enables or disables reading of data into the cache before you explicitly request it. To reduce the number of physical I/O operations, READ_AHEAD transforms a small I/O request into a larger request. The maximum number of blocks that can be read is equal to the maximum extent size.

The system default for this option is NOREAD_AHEAD; to transfer data into the cache, you must explicitly request it.

The default extent size for READ_AHEAD is 5 blocks. To use an extent size other than the default, specify a value that is equal to or greater than the extent size for virtual I/O requests. If you specify a smaller extent size, it is automatically increased to match the value for virtual I/O.

This option applies only to virtual I/O requests, and it is best for volumes containing data files that are accessed sequentially.

15.2 Monitoring Data Cache Operations

Once disk data caching is established through the MOUNT (MOU) or SET DEVICE (SET) command, you can monitor the performance of data caching with other commands provided by the system. Using these commands, you can evaluate the following aspects of data caching:

- Listings of all devices (cached and uncached) known to the system
- General data cache information about cache regions and devices (RMD C page)
- Detailed data cache information about cache regions and devices (RMD D page)

The sections which follow provide detailed descriptions of the DCL and MCR commands for monitoring data cache operations.

15.2.1 Displaying Information About Devices (Cached and Uncached)

The DCL commands SHOW DEVICE and SHOW DEVICE/[NO]CACHE (MCR commands DEV and SET) display information about devices recognized by the system.

Note that you cannot specify both a device and a qualifier in the same command line. To display information about a particular device, include a device specification. To list all cached (or uncached) devices, use the /CACHE (or /NOCACHE) qualifier.

Formats

```
DCL>SHOW DEVICE [dd[nn]:]

DCL>SHOW DEVICE /[NO]CACHE

MCR>DEV [dd[nn]:]

MCR>SET /[NO]CACHE
```

Parameters

dd

Specifies the devices about which you want to display information.

nn

Specifies the unit of a particular device about which you want to display information.

The device specification ddnn can also be a logical name assigned to the device.

If you do not specify a device, all the devices known to the system are listed. If you specify a device and not a unit, all the units of that device are listed. If you specify both a device and a unit, information about the specified unit is displayed.

Qualifiers

/CACHE

Lists all devices known to the system that are cached.

If you specify the /CACHE qualifier, but no devices are cached, the system does not display any information.

/NOCACHE

Lists devices that are not cached but could be cached.

15.2.2 Displaying General Cache Statistics

The Resource Monitoring Display (RMD) provides displays or "pages" of system information. The DCL command SHOW CACHE/REGION (or MCR command RMD C) allows you to display the C page, which shows general statistics about a particular cache region.

To display detailed data cache information about a specific device (that is, the RMD D page), use the /DEVICE qualifier and supply the device name (see Section 15.2.3).

After you analyze this information, you can use the SET command to modify system values and performance (see Section 15.1.2 and 15.3.)

Formats

```
DCL>SHOW CACHE [/REGION=name] [/RATE:nn]
```

```
MCR>RMD C [,REGION=name, RATE=nn]
```

DCL Qualifiers and MCR Setup Commands

/REGION=name
,REGION=name

Selects the cache region to be displayed. The default region is CACHE.

Note that the /DEVICE and /REGION qualifiers cannot be used together (see Section 15.2.3). If you do not specify either qualifier in the SHOW CACHE command line, /REGION is the default.

/RATE:nn
,RATE=nn

Specifies, in seconds, the rate between screen refreshes. The default rate is 1 second.

Example 15–1 is a sample RMD C page display; it is followed by an explanation.

Example 15-1: Cache Region Display (General Statistics)

```
RSX-11M-PLUS V4.0  BL40     Cache Statistics (General)       22-MAR-87 13:17:11
❶                               ❷
Cache Region Name: CACHE      Region Size: 47440 (2500. disk blocks)
               ❸      ❹     ❺      ❻         ❼     ❽     ❾     ❿        ⓫       ⓬
Device        Total Hit  Fail Load      Total  Hit  Fail Defer    Total    Cache
Name          Reads Rate Rate Rate      Writes Rate Rate Rate     I/O Ops  Used
DB6:         58834.  92%   4%   2%      19393.  62%   0%   0%      78227.   96%
DB7:             0.   0%   0%   0%          0.   0%   0%   0%          0.    0%

Total        58834.  92%   4%   2%      19393.  62%   0%   0%      78227.   96%    ⓭
```

The RMD C Display contains the following information about each cache region:

❶ Indicates the name of the cache region. By default, the name of the region is CACHE.

❷ Shows the actual memory size (in octal) of the cache region in 32-word memory blocks, followed in parentheses by the decimal number of disk blocks available in the extent storage area. (The memory size includes the private cache pool area as well as the extent storage area.)

❸ Displays the total count of all types of read requests issued to that device. These include VIRTUAL, READ_AHEAD, DIRECTORY, LOGICAL, and OVERLAY read requests.

❹ Displays the percentage of all read requests that were satisfied by the cache (the Read Hit Rate).

❺ Displays the percentage of all read requests that could not be satisfied by the cache (the Read Fail Rate). Requests to the cache may fail for one or more of the following reasons:

- An extent overlap
- A cache pool allocation failure
- A request size exceeding the specified extent size limit

Actual errors in loading the cache from the disk are not counted in this rate.

❻ Displays the percentage of all read requests that resulted in data being loaded into the cache (the Load Rate). The total of the Hit, Fail, and Load rates should be close to 100%; however, the total may not always equal 100. Read requests can fail due to a primary pool allocation failure that is not associated directly with read or write requests. The failure rate of primary pool allocation is shown on the RMD D page; it should be small.

❼ Displays the count of all write requests issued to the device. These include VIRTUAL, DIRECTORY, and LOGICAL write requests.

❽ Displays the percentage of all write requests that caused updating of blocks already loaded in the cache (the Write Hit Rate).

❾ Displays the percentage of all write requests that could not be directed to the cache because of an extent overlap (the Write Fail Rate). The combined hit and fail rates do not total 100% because a write miss is not considered a write failure. Also, write requests to a cache can fail because of a primary pool allocation failure, which is not associated directly with read or write requests. The failure rate of primary pool allocation is shown on the RMD D page; it should be small.

⑩ Displays the percentage of all write requests to temporary files that were deferred from write-through to the disk (the Defer Write Rate). The actual write operation to the disk was performed some time after the user's write request was completed.

⑪ Displays the total of all read and write operations for the device.

⑫ Indicates the percentage of the cache extent storage area used by the device.

⑬ Displays the grand totals for the cache region.

15.2.3 Displaying Detailed Cache Statistics

The DCL command SHOW CACHE/DEVICE (or MCR command RMD D) allows you to display the D page of the Resource Monitoring Display (RMD). The D page shows detailed statistics about cached devices. If you include a device specification, RMD displays information only about that device.

After you analyze this information, you can use the SET command to modify system values and performance (see Sections 15.1.2 and 15.3.)

Formats

```
DCL>SHOW CACHE/DEVICE[=ddnn:] [/RATE:nn]

MCR>RMD D[,DEVICE[=ddnn:][,RATE=nn]
```

Parameter

ddnn

Specifies the device for which you are requesting information. If you do not specify a device, the system displays information about all the devices known to the system.

DCL Qualifiers and MCR Setup Commands

/DEVICE=ddnn
,DEVICE=ddnn

Specifies the device about which you want to display detailed data cache information.

Note that the /DEVICE qualifer cannot be used with the /REGION qualifier. For more information, see Section 15.2.2.

The default device is the system disk (SY).

/RATE:nn
,RATE=nn

Specifies, in seconds, the rate at which the screen refreshes. The default rate is 1 second.

Example 15–2 is a sample RMD D page display; it is followed by an explanation. For more information about RMD, see Chapter 7 of this manual.

Example 15–2: Cache Region Display (Detailed Statistics)

```
RSX-11M-PLUS V4.0  BL40     Cache Statistics (Detailed)      22-MAR-87 13:17:22

Device Name:  SY0:                Region Name: CACHE    Region Size: 47440 ❶
Cache Status: Active,Defer ❷       Requests Being Cached: Dir,Ovr,Vir,Log,Rdh ❸

             Virtual   Readahead   Directory    Logical    Overlay      Total
Reads          14951.         0.      24398.      4842.     14676.     58867.  ❹
Read Hit Rate    87%         0%         93%        92%        95%        92% )
Read Load Rate    3%         0%          1%         4%         4%         2% |  ❺
                                                                             |
Read Overlap      0%         0%          5%         1%         0%         2% |
Extent Too Big    0%         0%          0%         0%         0%         0% )
Max Extent Size 127.       127.          3.       127.       127. ❻

Writes          8965.                 10380.        48.                19393.  ❼
Write Hit Rate   33%                    86%        70%        62% ❽
Write Overlap     0%                     0%         0%                    0%

Total I/O      23916.         0.      34778.      4890.     14676.     78260.  ❾

Primary Pool Allocation Failure Rate (as a % of Total I/Os):          0% )
Cache Pool Allocation Failure Rate (as a % of Total Read I/Os):       2% |  ❿
Read Load Failure Rate (as a % of Cache Load I/Os):                   0% |
Deferred Write Rate (as a % of Total Write I/Os):                     0% )
```

❶ Displays the region name and region size (see the description of the RMD C page). Note that the extent storage area size is not displayed (as it is in the RMD C page).

❷ Shows whether the cache is ACTIVE, ENABLED, DEFER, or a combination of all three.

- ACTIVE indicates that the device is being cached through the region.
- ENABLED indicates that the device will automatically be cached when it is mounted (even if the MOUNT command does not explicitly request that the device be cached).
- DEFER indicates that deferred write requests to temporary files are enabled for the device.

❸ Displays the types of cache I/O operations enabled for the device. The types can be VIRTUAL (VIR), READAHEAD (RDH), DIRECTORY (DIR), LOGICAL (LOG), and OVERLAY (OVR). However, note that values for RDH are not currently recorded; RMD displays them as zeros.

❹ Displays the number of read requests issued for each of the I/O types listed in the display, as well as the total of all read requests.

❺ Displays the Read Hit Rate, Read Load Rate, Read Overlap Rate, and the Extent Too Big rate for each type of request. Each of these rates is computed as a percentage of the total number of read operations for the particular type of request.

❻ Displays the maximum extent size for each of the I/O types.

❼ Shows the number of write requests issued for each of the I/O types and the total of all write requests. Note that READ_AHEAD and OVERLAY write requests do not exist, so these fields are blank.

❽ Displays the Write Hit Rate and Write Overlap Rate for each type of I/O request. Each of these rates is computed as a percentage of the total number of write operations of a particular type.

❾ Displays the total number of each type of I/O request, as well as the total number of all I/O operations.

❿ Displays the following rates, which are not accumulated by I/O type:

- The Primary Pool Allocation Failure Rate line shows how often the allocation cannot be done as a percentage of the total number of I/Os issued for the device.
- The Cache Pool Allocation Failure Rate is similar to the Primary Pool Allocation Failure Rate, except that allocation from the cache pool is only required on read operations. Therefore, the rate is computed as a percentage of the total number of read operations.
- The Read Load Failure Rate is the percentage of all cache load attempts that failed because of an I/O error.
- The Deferred Write Rate is the rate at which write requests to temporary files were able to be deferred. This is computed as a percentage of the total number of write operations.

All cache I/O requests require that an I/O packet be allocated from primary pool.

15.3 Modifying Data Cache Operations

After you monitor and evaluate the state of data cache operations, you can enhance data caching performance by modifying various cache parameters. To modify data caching on your system, use the DCL command SET DEVICE (or the MCR command SET).

The first time you use the SET DEVICE (SET) command, you associate caching with a device and set the default values for each of the data caching options. Subsequent SET DEVICE (SET) commands change only the options specified in the command line. All other options retain the values that you set when you associated caching with the device (using either the MOUNT (MOU) or SET DEVICE (SET) command). To modify the performance of a specific cached device, specify the device name in the command line.

You can also modify the size of a cache region; however, the size cannot be changed while the region is associated with a device. First, you must dissociate all devices from the region with the SET DEVICE /NOCACHE (SET /NOCACHE ddnn:) command. Then, use the CREATE and REGION options to create a modified cache region and reassociate the devices with that region (see Section 15.1.3).

To delete a region, use the REMOVE command. For example, if the cache region is *store*, use one of the following command lines:

```
DCL>

MCR>
```

For more information on the REMOVE command, see the *RSX-11M-PLUS Command Language Manual* or the *RSX-11M-PLUS MCR Operations Manual.* For more information on the syntax and options that are available with the SET DEVICE (SET) command, see Section 15.1.2.

15.4 The Data Caching Environment

This section describes the internal operations of data caching. Subsections include a detailed explanation of I/O cache considerations and data structures.

15.4.1 The Data Cache Region

The data cache region is composed of memory-resident copies of physically contiguous disk blocks, or "extents." The cache region stores blocks of disk data when a task requests a read operation. The Data Cache Manager (DCM) tags these blocks with their corresponding logical block numbers (LBNs) and then uses the LBNs to locate data during subsequent I/O requests.

If you enable read-ahead, the cache region creates a new extent size. The size of the extent is based on the size of the request and the number of additional blocks available in the file. The total size of the extents cannot exceed the maximum read-ahead value.

15.4.2 The Data Cache Manager

The Data Cache Manager (DCM) executes within the I/O system after all packet preprocessing and before disk driver activation. Because DCM is positioned at this point in processing, it can intercept I/O packets when all but the driver processing has been performed. If a task directs an I/O request to a device other than a disk, DCM maps and activates the appropriate device driver at the initiation entry point.

If deferred write request support is not enabled, DCM ensures that the data within the cache extents matches the data on the disk. If a single cache extent contains all the data to be read by an I/O request, the data transfer occurs from the extent and not from the disk. However, if the extent does not contain the requested data, or if the request overlaps only part of an extent, DCM forwards the I/O request to the appropriate driver for processing. The driver transfers the data from the disk without using the data or structures within the cache. When the physical I/O completes, DCM resumes processing and copies the requested data into your buffer.

During a write operation for I/O that is to be cached, DCM copies data from your buffer to the corresponding cache buffer. If there is no corresponding cache buffer, DCM forwards the request to the disk driver for processing. When the driver finishes copying the data to the disk, DCM completes the I/O processing by passing the packet to the I/O Finish routine ($IOFIN).

The MOUNT and DMOUNT commands issue the Set Characteristics function (IO.STC) as part of mount and dismount processing, respectively. IO.STC checks the volume label to validate the disk. If the volume label is correct, the device driver marks the volume "valid" by setting the volume valid bit. If you enable data caching, DCM uses IO.STC to ensure that the cache buffers reflect data for mounted (valid) disks. If you disable data caching, the driver uses IO.STC to ensure that I/O requests are processed for valid disks only.

When DCM receives the mount parameters, it initializes the cache, duplicates and dispatches the I/O packet, and retains the original request. Before initializing the cache, DCM purges the cache to eliminate any stale data.

The Executive I/O Finish routine ($IOFIN) returns the status of I/O processing. If an I/O packet is processed successfully, DCM increments a variable for the corresponding cache region to prevent deallocation of the cache. When DCM receives a dismount packet, it stalls the I/O packet until the device driver writes any outstanding buffers to the disk. DCM disables all further cache processing for that device. When the cache context for the device is clear (that is, when all the extents have been processed or eliminated), DCM forwards the stalled dismount request to the driver.

15.4.3 Deferred Disk Write Requests

Deferred disk write requests are implemented in two steps. First, the Executive recognizes an incoming I/O packet from a temporary file that is marked for deletion (also called a "temporary" file). Then, the Data Cache Manager (DCM) prevents the data already in the cache from being written to the disk. The subsections which follow describe these two actions in greater detail.

15.4.3.1 Recognition of I/O Packet Marked for Deletion

The data pool for the Files-11 Ancillary Control Processor (F11ACP) contains the File Control Block (FCB) bit SC.MDL. The SC.MDL bit indicates that a file has been marked for deletion. F11ACP propagates the bit to all window blocks associated with the file for later use by the Executive and DCM.

During a virtual I/O operation, a task calls the Executive module $MPVBN to convert the virtual block number (VBN) to a logical block number (LBN). At the same time, the Executive maps the window block. The window block maps an area for the file and stores the bit setting as part of the context for the virtual operation. The bit setting signals to DCM that the file is marked for deletion. DCM considers the file "temporary;" consequently, it uses data from the cache to process I/O requests.

The data on the disk remains unaltered until a write request overlaps only part of the cache extent or exceeds the boundaries of the extent (in either case, the data for the write request cannot be contained completely within the existing extent). If a write request requires a write operation to both the disk and the cache, the request is not deferred; instead, the data is transferred only to the disk and the extent is deleted.

15.4.3.2 Deferring Disk Data Transfers

When DCM defers a write operation to part (or all) of the storage blocks for a cache extent, it labels the extent with the Deferred Write Request (DFR) attribute. DFR indicates to the system that the extent has "new" data (that is, data that does not appear on the disk). Before DCM can process any other write requests that overlap the associated areas, the driver must write the data out to the disk.

Read and write requests to temporary files (that is, files marked for deletion) are processed using extents with the DFR attribute. Consequently, DCM executes these requests differently. The two subsections which follow describe these differences.

Read Requests to Extents with the DFR Attribute

You can disable DFR support by specifying the NODEFER_WRITES option for a disk. (See Section 15.1.4.) If DFR support is disabled, DCM ensures that the data within the cache extents matches the data on the disk. If a single cache extent contains the entire area to be read by a request, DCM uses the data from the extent. However, if a single extent does not contain the request, or if the request overlaps only part of an extent, DCM transfers the data from the disk without using the data or structures within the cache.

DCM processing remains the same for requests to areas contained completely within one cache extent. However, processing is modified for read requests that overlap part of one or several extents with the DFR attribute. If there are one (or more) extents for temporary files within an area, DCM reads the entire area from the disk (including the "stale" data from the DFR extent ranges). After the area is read from the disk, DCM overlays all DFR-associated extents (that is, it performs a "multiple overlay") into your current buffer. The multiple overlay supersedes the "stale" data that DCM read from the disk.

Write Requests to Extents with the DFR Attribute

If DFR support is disabled, cache support for write operations extends only to areas that are loaded into the cache by a prior read operation. If a write request maps completely into a cache extent, DCM updates the cache and writes the data out to the disk. If only part of a current disk extent covers the request, DCM deletes the extent and passes the original request to the disk driver.

If DFR support is enabled and there is a virtual request to a file marked for deletion, DCM does not write data out to the disk. If a write request maps completely into a cache extent, DCM updates the cache only. Because data is not written out to the disk, the disk contains old data. DCM labels the cache extent containing the new data with the DFR attribute.

If a virtual request does not completely map the extent previously marked with the DFR attribute, DCM stalls any subsequent I/O requests until the DFR extent is written to the disk. An I/O stall must occur to prevent the new data from the latest request from being superseded by the old data in the cache extent. If the I/O request is not stalled, the timing or selection of the device drivers could result in the loss of new data.

The DFR extents can be eliminated by removing the extent from the age list within the cache structures when the cache partition becomes full.[1] A write request from a task that is only partially contained within the cache extent also eliminates a DFR extent.

When an I/O is issued that causes a write overlap to occur to an area covered by a DFR extent, the following steps preserve the data on the disk and in the cache:

- DCM modifies the user task so that it reissues the request following a significant event. (For more information on events, see the *RSX-11M-PLUS and Micro/RSX Executive Reference Manual.*)
- Any extent within the affected area that is not marked DFR is erased.
- An internal I/O writes the DFR cache extent to disk.
- When the I/O completes, an internal completion routine is called to eliminate the extent.

Although the Executive sets SC.MDL for an entire extent, the data in the temporary file may only correspond to part of an extent. Regardless of the status of the extent, the requests to temporary files are deleted; however, the requests associated with a "regular" file continue to be written to both the cache and the disk.

[1] The age list is a list of the extents in the cache partition in order of latest manipulation. This list has a maximum number of entries allowed; if a particular extent is not referenced after a settable number of operations on other extents, the cache flushes the structure to make room for other areas that may be used more heavily.

15.4.4 Data Cache Structures and Symbolic Definitions

For disk caching purposes, a region is used to contain the cache-specific data structures and disk data buffers. This region can have any name. It is marked with the following Partition Control Block (PCB) status bits:

PCB Word	Status Bits and Functions
P.STAT	PS.FXD—Region is fixed in memory. PS.NSF—Region cannot be shuffled. PS.COM—Region is a common.
P.PRO	Protection word blocking all types of file access.
P.ST2	P2.RON—Read only (applies to external users). P2.APR—Depends on Executive (five for systems with separate Executive instruction and data space, six for systems without separate instruction and data space). P2.LMA—Do not remove this region. P2.CHE—Cache region (informational only).

Cache regions may be shared by multiple devices, or one region may be used exclusively by one device. When you enable caching for a device, DCM uses the system default cache region (CACHE) unless you specify another region.

Data Cache Region Format

The cache region is formatted when it is created. When data caching is enabled for a disk device, the Unit Control Block Extension (UCBX) is marked with the address of the data cache region (PCB). To force the binding of the region with the device to prevent illegal deallocation, the resident mapped task count field (P.RMCT) reflects the number of devices that currently have caching enabled for that region. The region cannot be deleted as long as P.RMCT is nonzero. When there are no devices with access for that region, the status bit P2.LMA is cleared, allowing the region to be removed from the system.

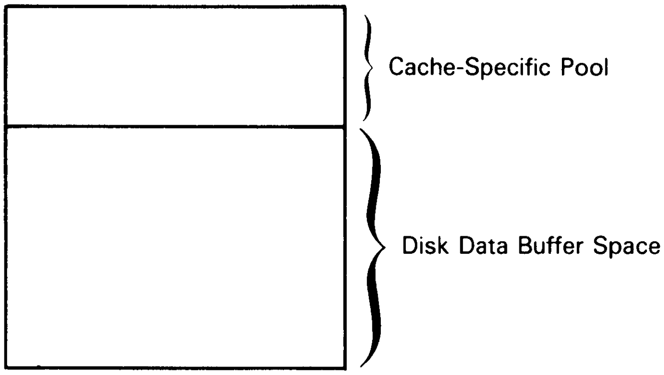
Data Cache Structures

The data cache region has two parts. (See Figure 15–1.) The beginning of the region contains pool space. The remainder of the cache region is used to buffer disk data blocks. It is allocated on 32-word boundaries in 256-word increments.

Systems without separate Executive instruction and data space map pool through Active Page Register (APR) 6 and have a 4K limitation; systems that support separate instruction and data space map pool through data space APR 5 and APR 6 with an 8K limitation. DCM uses the Executive pool allocation and deallocation routines ($ALOCB (entry point $ALOC1) and $DEACB (entry point $DEAC1), respectively) to allocate and deallocate space from this area. The dynamic data structures, which are cache-specific, are retained in this area to minimize the load of data caching on the primary pool resource.

The Cache Extent Descriptor (CED) is a data structure within the region's pool that describes all data buffer space in the cache region, whether the space is in use or not. This allows contiguous blocks of space to be reclaimed when the system needs them.

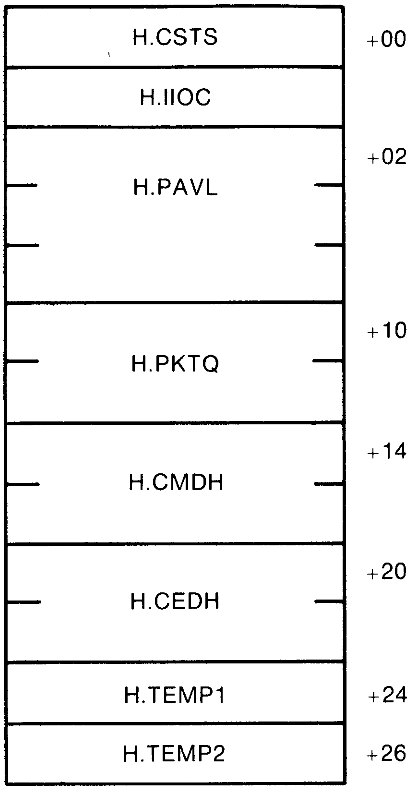
Figure 15–1: Format of Data Cache Region



ZK-4187-85

The pool space contains a header that is used to control various region-specific structures. (See Figure 15–2.)

Figure 15–2: Format of Pool Header Area



ZK-4188-85

The pool header area contains the following information:

H.CSTS Defines the region code status.

H.IIOC Defines the region internal I/O count, which is not used by the cache region.

H.PAVL Defines the available pool list. This is a standard 3-word pool list head, which contains the alignment word followed by the list head, which is a pointer to the next available block and the size of the block.

H.PKTQ Defines the pending packet queue (in descending priority order). These packets are waiting for data cache completion processing. They either have I/O in progress, are waiting for data movement to begin, or they are waiting to be dispatched to the user or the driver.

H.CMDH Defines the list head for region buffer space allocation.

H.CEDH Defines the list head for CEDs ordered by age (the oldest are at the head of the queue).

H.TEMP1 Defines the pointer used in allocating new extents.

H.TEMP2 Defines the pointer used in allocating new extents.

The region is formatted before caching is enabled on any dependent disk devices.

Part IV: Advanced Features

Chapter 16

Shadow Recording (RSX–11M–PLUS Systems Only)

Shadow Recording (SHA) is a system generation option for RSX–11M–PLUS operating systems. If you select the Full-functionality Executive during system generation, Shadow Recording is included by default. This chapter describes Shadow Recording, how to prepare your system for it, and how to control it. It also contains information on bad block handling and a list of error messages.

16.1 Introduction to Shadow Recording

Shadow Recording backs up all new data as it is written to a Files–11 disk. It creates two identical sets of disks called a *shadowed* pair. You can shadow more than one pair of disks, but shadowed disk pairs cannot overlap.

The first disk of the pair (the primary disk) is the original disk. It exists whether or not Shadow Recording is active. Any disk on your system, including the system disk, can be the primary disk of a shadowed pair. The second disk of the pair (the secondary disk) is an exact copy of the primary disk.

Shadow Recording has important uses in the following applications:

- Duplicating important information to prevent damage or loss. With Shadow Recording, critical data is duplicated on two disks. This redundancy prevents loss of data if a disk error occurs and provides quicker recovery time in critical applications. The length of time an application is unable to operate (the "down time") is reduced because you do not need to halt the application to correct for disk errors.
- Providing an online ("hot") backup so that a duplicate disk is immediately available (even when backup time or resources are not).

Shadow Recording operates transparently; it does not change any aspects of the system that are visible to you. During Shadow Recording, the Executive writes the same data to the secondary disk that it writes to the primary disk. However, when a disk read occurs, the Executive reads the primary disk first. If a read error occurs on the primary disk, the Executive reads the secondary disk. The Executive displays all I/O errors occurring on a Shadow Recording disk pair on the console device (pseudo device CO:).

If you encounter errors during any of the Shadow Recording procedures, refer to Section 16.4.

16.2 Preparing Your System for Shadow Recording

You can shadow more than one pair of disks, but shadowed pairs cannot overlap. For example, if you make DB0 and DB1 a shadowed pair, you cannot make DB1 and DB2 a shadowed pair. Also, the devices must be of the same type (for example, two RP06 or two RM02 disks).

Before you can use a pair of disk drives for Shadow Recording, you must have two identical disk packs available. The two disk packs must have the same bad block data. You can make the bad block data on the two disks functionally equivalent by following the procedure discussed next. The procedure assumes you are Shadow Recording the system disk. However, it is valid for any two devices of the same type.

To use Shadow Recording, perform the following steps:

1. Select Shadow Recording as an option during system generation.
2. Select a set of disks (three or more) that you can use for Shadow Recording. If disk errors occur, you can use the extra disks in the set for a quick recovery. In critical applications, this can decrease the down time.
3. Select two device units for the shadowed pair. One of these two devices will be the device that contains the primary disk that you want to shadow, the other will contain the secondary disk.

   For example, if your system has four RP06 disks and you decide to shadow record the system device (DB0), use one of the three remaining devices (DB1, DB2, or DB3) for the secondary disk.
4. Run the Bad Block Locator Utility (BAD) with the /LIST switch on all the disks in the set of Shadow Recording disks. BAD locates the bad blocks and lists them on your terminal. This list includes information you need for the next step.
5. Ensure that the disks have the same bad blocks by making the bad block descriptor block on the primary disk the logical OR of all the bad blocks in the set of disks.

   To make entries in the bad block descriptor block of the primary disk, initialize the disk by using the DCL command INITIALIZE (MCR command INI) with the /BADBLOCKS:MANUAL qualifier (/BAD=[MAN] keyword). Use one of the following formats:

   ```
   DCL>

   MCR>
   ```

   ### Parameters

   /BAD_BLOCKS:MANUAL (/BAD=[MAN])

   Includes bad block processing in the volume initialization and instructs INI to accept a bad block list specified from the terminal.

   ddnn

   Specifies the device mnemonic (dd) and unit number (nn) for the primary disk being initialized.

volumelabel

Specifies the Files–11 volume label for the primary disk.

After you enter the INITIALIZE (or INI) command, the system prompts for bad blocks, as follows:

```
INI>LBN(S)=
```

In response to this prompt, type the list of bad blocks. (Subsequent INI, Disk Save and Compare Utility (DSC), and Backup and Restore Utility (BRU) operations use the bad block information. For a description of DSC and BRU, see the *RSX-11M-PLUS Utilities Manual.*)

For more information on the Monitor Console Routine (MCR) command INITIALIZE VOLUME, see the *RSX-11M-PLUS MCR Operations Manual.* For information on the DIGITAL Command Language (DCL) command INITIALIZE, see the *RSX-11M-PLUS Command Language Manual.*

6. Use the Backup and Restore utility (BRU) to copy the data from your system disk to your primary disk. This disk becomes the primary disk and your new system disk for Shadow Recording.

   For example, if you want to shadow record your system disk, copy it to the primary disk. After this is done, you will have two system disks with identical data. However, the disks are not physically identical because of the bad block locations.

   When you start Shadow Recording, the contents of the primary disk (including the bad block information) are copied to the secondary disk. After Shadow Recording is enabled, the Executive writes the same data to the secondary disk that it writes to the primary disk. When you make the bad block information on the primary disk a logical OR of all the bad blocks in the set of disks (including its own), the write to the secondary disk does not occur for a defective block.

7. Use the DCL command MOUNT (MCR command MOU) on the secondary device with the /FOREIGN qualifier (/FOR keyword) to mount one of the disks you selected in step 2. Use one of the following formats:

```
DCL>
```

```
MCR>
```

Parameters

ddnn

Specifies the device mnemonic (dd) and unit number (nn) for the secondary disk.

Now, you are ready to use Shadow Recording, beginning with the START command (see Section 16.3.1).

16.3 Controlling Shadow Recording

Five commands control Shadow Recording:

- START
- STOP
- ABORT
- CONTINUE
- DISPLAY

Enter these commands from a privileged terminal only.

To invoke Shadow Recording, enter the task name SHADOW (or SHA) at the system prompt. Then, enter one of the five commands at the explicit prompt SHA> . For example:

```
>
SHA>
```

You can also enter both commands on one line, as follows:

```
>
```

The sections that follow describe the Shadow Recording commands.

16.3.1 START Command

The START command invokes a Shadow Recording task called SHAddn (or SHddnn), where ddnn is the device unit of the primary disk. This task copies the primary disk to the secondary disk. Shadow Recording occurs while this copy operation, called *catchup*, is in progress. Records are written to both the primary disk and the secondary disk without interfering with the catchup process.

After you issue the START command, Shadow Recording provides a dynamic backup of all blocks as they are written to the primary disk.

The START command performs the following functions:

- Verifies that the primary disk is a Files–11 device and is mounted
- Verifies that the secondary disk is mounted as a foreign device
- Verifies that both the primary and secondary disks are identical disk types
- Sets up the Shadow Recording data structure (UMB) that starts the Shadow Recording task within the Executive
- Starts the copying of the primary disk to the secondary disk

Format

```
START ddnn: TO ddxx:
```

Parameters

ddnn

Specifies the device mnemonic (dd) and unit number (nn) for the primary device.

ddxx

Specifies the device mnemonic (dd) and unit number (xx) for the secondary device (the shadowed copy of the primary device).

The START command copies the primary disk onto any secondary disk of the same device type. After you make the bad block information identical, and then use the START command, the secondary disk becomes a logical duplicate of the primary disk. All data structures (such as data blocks, bad blocks, headers, and directories) occupy the same logical locations on the secondary disk as they do on the primary disk.

Both disks have equivalent logical block numbers (LBNs). If they are system disks, both disks are bootable. In addition, after the START command is issued, the secondary disk becomes a Files–11 format disk. The START command makes an exact copy of the primary disk (including the Files–11 format) on the secondary disk.

If new data is written to the primary disk after Shadow Recording is initiated, Shadow Recording also writes the new data to the secondary disk. That is, when a write occurs on the primary disk, it also occurs on the secondary disk. However, the secondary disk is read only from the last portion that has been copied. The information about the last block copied is in the UMB control block. A fully redundant secondary disk does not exist until all the data is transferred from the primary disk to the secondary disk.

16.3.2 STOP Command

The STOP command performs the following functions:

- Verifies that the primary device is part of a shadowed pair
- Unlinks the UMB data structure, if no outstanding I/O exists, thereby stopping the Executive from Shadow Recording
- Marks the data structure for deletion, if there is outstanding I/O

Format

```
STOP ddnn:
```

Parameters

ddnn

Specifies the device mnemonic (dd) and unit number (nn) for the primary device.

The STOP command prevents Shadow Recording from continuing. However, while the primary disk is being copied to the secondary disk (that is, while *catchup* occurs), the STOP command cannot stop Shadow Recording. If you issue the STOP command during catchup, you receive an error message. To stop Shadow Recording during catchup, use the ABORT command (see Section 16.3.3).

The absence of outstanding I/O does not necessarily mean that all file activity has stopped. To ensure the integrity of the secondary data, all tasks that write records to the shadowed pair must stop before you issue the commands STOP or ABORT.

16.3.3 ABORT Command

The ABORT command performs the following functions:

- Verifies that the primary disk is part of a shadowed pair
- Aborts catchup if it is in progress and then executes a STOP command
- Stops Shadow Recording, even if catchup is in progress

Format

```
ABORT ddnn:
```

Parameters

ddnn

Specifies the device mnemonic (dd) and unit number (nn) for the primary device.

16.3.4 CONTINUE Command

The CONTINUE command performs the following functions:

- Assumes that the primary and secondary disks are identical
- Verifies that the primary disk is in Files–11 format and is mounted properly
- Verifies that the secondary disk is mounted foreign
- Verifies that both disks are identical device types
- Sets up the Shadow Recording data structure, which starts Shadow Recording within the Executive

Format

```
CONTINUE ddnn: TO ddxx:
```

Parameters

ddnn

Specifies the device mnemonic (dd) and unit number (nn) for the primary device.

ddxx

Specifies the device mnemonic (dd) and unit number (xx) for the secondary device.

The CONTINUE command allows you to restart Shadow Recording on the same two disks that you were using when the STOP command was issued. The CONTINUE command assumes that the two disks are physically alike, and it does not check the disks for equality. After you issue the STOP command, do not allow writing to occur on the primary or secondary disks.

16.3.5 DISPLAY Command

The DISPLAY command displays a list of all the shadowed pairs that are currently known to the system.

Format

```
DISPLAY
```

The shadowed device pairs are displayed in the following format:

```
xxxxxx          ddnn          ddxx
```

The three display fields contain the following information:

xxxxxx Displays the address of the UMB control block for Shadow Recording. This control block is in the system Executive pool space.

ddnn Specifies the device mnemonic (dd) and unit number (nn) for the primary device.

ddxx Specifies the device mnemonic (dd) and unit number (xx) for the secondary device.

16.4 Error Handling

The Executive writes data from a task buffer to the primary disk and then writes it from the task buffer to the secondary disk. (The START command, which copies the primary disk to the secondary disk, is an exception.) If write errors occur on one of the disks, you may be unaware of the errors until the Executive attempts to read the block in error.

To be notified of write errors as they occur, run Shadow Recording with write-checking enabled for both disks of the shadowed pair. The DCL command SET DEVICE/WRITE_CHECK (MCR command SET /WCHK=ddnn) enables write-checking on the device, ddnn. Note that a write-check of every write operation takes more time to execute than a write-only.

When write-checking is enabled, the Shadow Recording program prints the error messages on the console terminal. These error messages consist of device unit numbers and LBNs.

For a read operation, the Executive reads from the primary disk first. If it encounters an error on the primary disk, it reads the same record from the secondary disk. It is unlikely that both disks would develop an error in exactly the same place. However, if they do, a notification of read errors on the secondary disk means that the primary disk is also bad in the same location.

The following subsections discuss the procedures you can follow when a read error occurs.

16.4.1 Errors on the Primary Disk

If errors occur when the Executive tries to read from the primary disk, the Executive tries to read the same record from the secondary disk. You are notified of the errors by a message on the console. You can let your task continue, but be aware that the shadowed pair is no longer alike.

If you decide to stop the processing task (or tasks), then you can stop Shadow Recording with the STOP command. The secondary disk contains all the data that was on the primary disk, including the bad block record. Use the secondary disk as the new primary disk. Designate a new device of the same type for the new secondary disk. Then, mount the new secondary disk on the secondary device as foreign.

After mounting the old secondary disk on the primary device as the new primary disk, enter the START command. The START command copies the primary disk to the secondary disk, including the bad block record.

16.4.2 Errors on the Secondary Disk

If errors occur on the secondary disk, the shadow of the primary data is no longer valid. Stop Shadow Recording, mount a new secondary disk, and restart Shadow Recording. The catchup task (SHAddn) then copies the primary disk to the new secondary disk.

16.4.3 Errors on the Primary and Secondary Disks

To correct for a disk error, stop Shadow Recording and your tasks. Then, select a new primary disk and new secondary disk, mark the bad block information on the new primary disk, and copy the old primary disk to the new primary disk.

If the error occurs on different blocks on both disks, save the data by using the old primary and secondary disks as a shadowed pair input to the Backup and Restore Utility (BRU). BRU copies all the data that is on the input disk and writes it to the output disk. If you use a shadowed pair as input to BRU, only the valid or "good" data is written to the output disk. If primary disk errors occur, the Executive does an alternative read on the secondary disk.

BRU does not create a new primary disk; you must create a new primary disk from the BRU output (see Section 16.2).

For more information on BRU, see the *RSX-11M-PLUS Utilities Manual.*

16.5 Shadow Recording Messages

This section lists error messages that you may receive during Shadow Recording. The first message consists of time-stamping and a list of information, which is described in the explanation that follows. The remainder of the Shadow Recording messages appear in the following format:

```
SHA--message text
```

The messages are listed alphabetically, according to the text of the message. Each message is followed by a brief explanation and a suggested response to the error.

```
hh:mm:ss—Handler Error code-nn. on ddnn:
          Shadow Pair: ddnn: ddxx:
          Task: xxxxxx
          Logical Block: nnn.
          Bytes in xfer: nnn.
          File ID: n,n
          File Name: xxxxxx
          nn. errors lost due to saturation.
```

Explanation: This message contains the following information:

hh:mm:ss
Specifies the time of the error in hours, minutes, and seconds.

Handler Error code-nn.
Specifies the I/O error code (in decimal) for the error. (I/O error codes are summarized in the *RSX-11M-PLUS and Micro/RSX I/O Operations Reference Manual.*)

ddnn:
Specifies the device on which the error occurred.

Shadow Pair: ddnn: ddxx:
Specifies the devices containing the primary (ddnn:) and secondary (ddxx:) disks of the shadowed pair.

Task: xxxxx
Specifies the name of the task that was executing at the time the error occurred.

Logical Block: nnn.

Specifies the LBN (in decimal) on which the error occurred.

Bytes in xfer: nnn.

Specifies the number of bytes of data (in decimal) being transferred when the error occurred.

File ID
File Name

Specifies the file identification and file name. These are printed only when they can be determined at the time of the error.

nn. errors lost

Specifies, in decimal, the number (nn) of errors lost. When an error occurs on a shadowed disk, the Executive stores information about the error in pool and notifies the shadow error reporting task. Then, the task reads the information and prints the above message on the console terminal.

If the Executive stores more than 15 errors before the error reporting task can process them, pool is depleted. To prevent this from occurring, the Executive allows only as many errors as it can process to be queued in pool at one time. If the number of errors exceeds 15, the Executive records the errors without processing them. The message "nn. errors lost" tells you how many errors were ignored.

The term *saturation* refers to pool being filled with error messages.

This message is only printed when errors have not been processed.

User Action: See Section 16.4.

SHA—Abort error

Explanation: This message occurs if the Shadow Recording ABORT command cannot abort Shadow Recording during catchup.

User Action: Wait until catchup terminates and then issue the STOP command.

SHA—Bad indirect file specification

Explanation: The indirect command file name is incorrect.

User Action: Reenter the indirect command file specification correctly.

SHA—Bad volume type

Explanation: The disk is not in Files–11 format or the primary and secondary disks are different disk types.

User Action: Check the command line or disks that you are trying to use.

SHA—Catch-up completed

Explanation: The primary disk has been copied completely to the secondary disk.

User Action: No user action is required. This is an informational message.

SHA—Catch-up in progress

Explanation: You tried to issue a STOP command during the catchup process.

User Action: If catchup is still in progress, enter the ABORT command.

SHA—I/O error on logical block nnn.

Explanation: An I/O error occurred during catchup.

User Action: None, unless the I/O errors are frequent. If the I/O errors are frequent, enter the ABORT command, change disks, and restart Shadow Recording.

SHA—Illegal command

Explanation: The command that you entered is not a valid command.

User Action: Reenter the command with the correct spelling and syntax.

SHA—Input error on input command file
FILE=

Explanation: An error occurred while the indirect command file was being read.

User Action: Use a different indirect command file or use a copy of the specified command file.

SHA—Illegal parse data (internal error)

Explanation: This is an internal system error. You entered the command correctly.

User Action: Reenter the command. If the error persists, submit a Software Performance Report (SPR).

SHA—Invalid device

Explanation: The device is not a disk device.

User Action: Check the device specification and the device.

SHA—Maximum indirect command file depth exceeded

Explanation: The indirect command file was nested to too many levels. Only three levels are allowed.

User Action: Check the indirect command file.

SHA—No room available for UMB

Explanation: Not enough pool space exists to allocate space for the UMB control block for Shadow Recording.

User Action: Check your system. The system may be running out of pool space (see Chapter 8).

SHA—Not a shadowed pair

Explanation: When STOP or ABORT was issued, the wrong device was specified.

User Action: Reenter the command with the correct device unit.

SHA—Open error on input command file
FILE=

Explanation: The input command file could not be found.

User Action: Check the command file and its location.

SHA—Privileged command

Explanation: You are not using a privileged account or terminal.

User Action: Log in on a privileged account or terminal.

SHA—Send/Receive error

Explanation: An Executive directive Receive Data (RCVD$) error occurred in Shadow Recording during catchup. This is a system error.

User Action: No user action is required. However, if the error persists, submit an SPR to DIGITAL.

SHA—Shadow Recording not in system

Explanation: Shadow Recording is not part of your system.

User Action: If you have an RSX-11M-PLUS operating system, select support for Shadow Recording during system generation. Micro/RSX systems do not support Shadow Recording.

SHA—Start-up error

Explanation: Shadow Recording cannot spawn the catchup task, so Shadow Recording has been forced to exit.

User Action: Restart Shadow Recording with the START command.

SHA—Syntax error

Explanation: You entered a command with incorrect syntax.

User Action: Reenter the command with the correct syntax.

SHA—System error

Explanation: A system error occurred.

User Action: If possible, determine the effects of the error. If this error recurs, submit an SPR to DIGITAL.

SHA—Unknown command error

Explanation: Shadow Recording received a command line that it could not recognize.

User Action: Check the command that you entered or the indirect command file.

SHA—Volume currently part of shadowed pair

Explanation: The specified device is already being shadow recorded.

User Action: Check the devices being shadowed with the DISPLAY command.

SHA—Volume not mounted properly

Explanation: The primary disk is not mounted Files–11, or the secondary disk is not mounted foreign.

User Action: Check the disks and mount them correctly.

Chapter 17

Command Line Interpreters

This chapter introduces the concepts of command line interpreters (CLIs) for RSX-11M-PLUS and Micro/RSX operating systems. A brief description of two CLIs supplied with your operating system illustrates these concepts. Although these CLIs process command lines differently, the basic goal of each is the same: to provide a specific environment for the terminal user. You can use this information to write a CLI that is specific to your application.

A sample CLI (written in MACRO-11 and FORTRAN) is included in this chapter. The sample illustrates the CLI interface and clarifies the basic operations of a CLI. It is also supplied in the following files:

- TMCLI.MAC (the MACRO-11 version)
- TMCLI.FTN (the FORTRAN version)

On RSX-11M-PLUS distribution kits, these files are located on pseudo device LB: in the named directory [USER]. On Micro/RSX distribution kits, these files are located on the diskette or the tape backup set labeled REFERENCE in the [USER] directory.

17.1 Introduction to Command Line Interpreters

A command line interpreter (CLI) is a task that services commands for the operating system or an application. These commands are not prompted for by any task nor are they given to a task by a read operation on a terminal. No read Queue I/O (QIO) or asynchronous system trap (AST) processing of characters is involved. By default, the input from the terminal goes to the CLI for processing.

A CLI task is a complete user interface. All lines typed on a terminal are processed by the CLI (except those used to satisfy a read request from a task). You cannot bypass a CLI to send commands to the system or application. Thus, the CLI serves as a connection between an application command processor and any number of terminals.

Every terminal on an RSX system is set to a CLI; the CLI controls the use of the terminal. The following sections describe MCR and DCL, the two CLIs that are supplied with your operating system.

17.1.1 The Monitor Console Routine

The Monitor Console Routine (MCR) receives commands from the terminal and executes them to control the system. It consists of a set of privileged tasks that call Executive routines and interact with the system's database. Due to MCR's function in system control, it is tied closely with the Executive.

While MCR is a CLI, it is a special CLI. MCR does not use the CLI interface directives because it must work on systems that do not support these directives. Most other CLIs interact with an application running on the system, instead of interacting with the system itself. Therefore, MCR is not a model for other RSX-11M-PLUS or Micro/RSX CLIs.

17.1.2 DIGITAL Command Language

The DIGITAL Command Language (DCL) interpreter is a CLI that does not execute any commands. DCL provides a connection to the operating system by translating the commands it receives (in DCL syntax) to the equivalent MCR commands. Then, DCL passes the translated command line to the appropriate MCR task or utility for execution. This operation is invisible to you; you do not need to know about MCR or the utilities.

DCL is oriented toward operations, such as COPY or LINK, rather than toward tasks, such as the Peripheral Interchange Program (PIP) and the Task Builder (TKB). DCL presents a different external interface, one that is easier to learn and use than MCR.

17.2 Installing and Initializing a CLI

Before a CLI can process commands, the CLI must be made "known" to the system. That is, you must create an internal data structure that allows the system to recognize the existence of the CLI. This is done by installing the CLI task with the DCL or MCR command INSTALL.

Formats

```
DCL>
```

```
MCR>
```

Parameter

filespec

Specifies the name of the task image file containing the CLI you want to install.

Qualifier (Keyword)

/INTERPRETER (/CLI=YES)

Specifies that the task being installed is a CLI.

After you install the CLI, initialize it with the MCR command CLI /INIT, as follows:

```
>
```

The CLI command also supports other keywords and subkeywords. By specifying the appropriate keywords, you can make a CLI known to the system, you can eliminate it from the system, and you can enable or disable its recognition of commands. For example, if you specify the CLI /INIT command with the /MESSAGE subkeyword, the CLI receives messages from the system informing it of relevant events.

For more information on the MCR commands INSTALL and CLI, see the *RSX-11M-PLUS MCR Operations Manual*. For a description of the DCL command INSTALL, see the *RSX-11M-PLUS Command Language Manual*.

17.2.1 Specifying Task Names for CLIs

You can install CLIs with task names in the form ...xxx (a *prototype task name*) or with names in some other valid format, such as xxxxxx (a *conventional task name*).

If the CLI has a *prototype task name*, the system creates a separate copy of the CLI each time a command is entered on a terminal set to that CLI. The task is named xxxttnn, where ttnn is the device specification of the terminal from which the command was issued. The task follows the same rules as any other task installed in the system with a prototype task name. Each copy of the CLI runs with the same TI: and User Identification Code (UIC) as that of the issuing terminal (TI:). These tasks exit when they finish processing a request.

If the CLI has a *conventional task name*, all commands associated with the CLI are queued to a single copy of the task. Instead of running a task from each terminal, this task uses the console output device (CO:) as the TI:. CLIs with conventional task names specify the TI: and UIC fields in any RPOI$ or SPWN$ directives that they issue. If the task names do not specify these fields, the default values for TI: and UIC are incorrect.

CLIs with conventional task names have only one copy in the system. So, when there are no commands queued to the CLI, it stops (instead of exiting). The CLI maintains information from one command to the next.

DCL installs with a prototype task name (...DCL) because most DCL commands issue prompts for input. While DCL waits for a user to respond, it cannot do any command processing. If only one copy of DCL existed, command processing for all DCL terminals would stop while a prompt request to any one terminal was outstanding. Giving each terminal its own copy of DCL allows independent command processing for each terminal. Individual copies of the task exit when no commands are left for them; there is no need to maintain context between commands.

17.2.2 Prompts that Affect CLIs

There are three kinds of prompts issued by the system that affect CLIs. They are as follows:

CTRL/C prompts

Issued by the terminal driver when it receives a CTRL/C character. When you initialize the CLI, you can specify the string to be issued. However, because the terminal driver issues the prompt, the CLI cannot be notified when it is issued.

Empty-Line prompts

Result from terminal input consisting solely of a carriage-return character (a null command line). The MCR dispatcher task issues the default prompt; you can specify the default prompt when you initialize the CLI.

If you initialize the CLI with the /NULL qualifier, the system sends null command lines to the CLI as if they were commands. The CLI is responsible for any prompting or appropriate response.

Task-Exit prompts

Issued when a task that was started by a command line exits. The standard system action is to have the MCR dispatcher task issue the same default prompt string as in the case of an empty line.

To inform the CLI of these events, initialize the CLI with the /PROMPT qualifier. The /PROMPT qualifier causes the system to send these task-exit prompt requests directly to the CLI; no output is sent to the terminal. The CLI is responsible for any prompting or appropriate response.

A CLI initialized to receive either empty-line or task-exit prompts has to issue an SNXC$ directive when its processing of the prompt is complete. SNXC$ tells the terminal driver that the current command is complete, and it can send the next command if the terminal is in serial processing mode. Even if the terminal is not in serial mode, this directive can be issued without any adverse effects (see Section 17.3.4 for more information).

17.2.3 Handling Continuation Lines

Continuation lines are handled by the terminal driver before the command is passed to the CLI. Incomplete segments of command lines are detected by the presence of a hyphen (-) as the character immediately preceding the terminator. Any hyphens are removed and all the pieces of the command are put together before the CLI is aware of the command. The CLI does not process the continuation lines.

If you do not want to use hyphens to denote continuation lines, initialize the CLI with the /SNGL keyword. /SNGL tells the system to accept single-line commands only and to pass every line to the CLI as it is entered. No continuation line checks are made.

17.3 Writing an Application CLI

You do not need to be an expert on the internal operation of the operating system to design a CLI. However, an alternate CLI is expected to synchronize with the operating system in command handling and prompting. This interaction is achieved through a set of Executive directives for CLIs, which are described in the following subsection.

17.3.1 Using Executive Directives for CLIs

The most important Executive directive used by a CLI is Get Command for Command Interpreter (GCCI$). Any CLI you write must include this directive. GCCI$ supplies a command to the CLI from a queue maintained by the system. It also controls the internal status bits necessary for correct synchronization of prompting and general command handling in a manner transparent to the CLI task.

The following Executive directives may also be used by a CLI:

- The Request and Pass Offspring Information (RPOI$) and Send, Request, and Pass (SDRP$) directives direct commands to another task so that the CLI is free to receive more commands.
- When a command arrives for processing by the CLI, the Specify Command Arrival AST (SCAA$) directive specifies an asynchronous system trap (AST) in the CLI.
- The Get Command Interpreter Information (GCII$) directive returns information about a terminal or CLI to the task.

See the *RSX-11M-PLUS and Micro/RSX Executive Reference Manual* for a complete description of these directives.

17.3.2 Interacting with the Operating System

A CLI's operation starts when you enter a command into the system. To receive a command, the CLI must be associated with the terminal from which the command originated. You can type a command line on a terminal or execute a task that issues a command.

First, the command is processed by the system command dispatcher (the MCR... task). This task puts the command into a first-in/first-out (FIFO) queue of commands to be retrieved by the CLI. Then, the CLI runs (or restarts, if it is already active). The CLI gets the command from the system by issuing the GCCI$ directive. GCCI$ removes a command from the queue and copies it into a buffer in the CLI task space. It can also supply information to the CLI about the terminal.

When the CLI receives a command, it executes the command or passes the command to another task for execution. If you want the CLI to pass the command to another (non-CLI) task, use the RPOI$ or SDRP$ directives. These directives propagate the information the system needs for controlling, prompting, and getting exit status back to a parent task (if any). This is the recommended procedure for handling commands because it avoids the wait for the preceding command to complete execution. DCL works in this way, translating a command into a different format and dispatching it to another task for execution.

Note

> CLIs that process commands themselves need to control delays in execution. If you write a CLI that performs I/O to a terminal, a user could suspend command handling by not responding to a prompt or by pressing CTRL/S.

After a CLI processes a command, it reissues the GCCI$ directive to get another command from the queue. If the queue is empty, the CLI can stop or exit by issuing the GCCI$ directive with an appropriate null operation (NOP) specified. This prevents a command, queued between the return from the GCCI$ and the issuance of an EXIT or STOP directive, from failing to be processed until the next command is queued.

17.3.3 Receiving System Messages

The GCCI$ directive returns system messages to the CLI the same way it returns commands. However, the CLI differentiates messages from commands by clearing the first word of the command buffer (G.CCDV). If the CLI receives a command, it records the name of the device from which the command originated in G.CCDV. The code that identifies the message is returned in the unit number byte G.CCUN.

The other fields have the same meaning for messages as they do for commands. If the message does not contain any American Standard Code for Information Interchange (ASCII) text, the text field of the buffer is blank.

Note

> The CLI command includes both a /MESSAGE subkeyword to the /INIT keyword and a /MESSAGE keyword. The /MESSAGE subkeyword specifies that the CLI being initialized can receive messages. The /MESSAGE keyword sends an ASCII message to a CLI that has been initialized with the /MESSAGE subkeyword.

Message codes 1 to 127_{10} are reserved for use by DIGITAL. To create your own messages to CLIs, use codes in the range of 128_{10} to 255_{10}. The symbol shown for each message code is defined in the Executive definition library LB:[1,1]EXELIB.OLB. Tasks obtain the messages at build time by linking with the library.

The predefined message codes are as follows:

CLI Initialized into an Enabled Mode

Message Code: 1 (CM.INE)

Text: Displays the CLI name in Radix–50. The length of the text is 4 bytes.

Explanation: This message is sent to a CLI when it is initialized by a CLI /INIT command without the /DISABLE subkeyword. The CLI is enabled.

CLI Initialized into a Disabled Mode

Message Code: 2 (CM.IND)

Text: Displays the CLI name in Radix–50. The length of the text is 4 bytes.

Explanation: This message is sent to a CLI when it is initialized with the /DISABLE subkeyword to the CLI /INIT command. The /DISABLE subkeyword initializes the CLI in disabled mode. The system rejects any commands issued from the terminal until the CLI is enabled.

CLI Enabled

Message Code: 3 (CM.CEN)

Text: No text accompanies this message.

Explanation: This message is sent to a previously disabled CLI when it is enabled by a CLI /ENABLE command. The CLI can now accept commands.

CLI Disabled

Message code: 4 (CM.CDS)

Text: No text accompanies this message.

Explanation: This message is sent to the CLI when it is disabled by a CLI /DISABLE command. The CLI cannot accept commands. This effectively stops system use by terminals set to this CLI until the CLI is enabled again.

CLI Eliminated

Message Code: 5 (CM.ELM)

Text: No text accompanies this message.

Explanation: This message is sent to a CLI when it has been eliminated by an explicit CLI /ELIMINATE command. After this command executes, you can remove the CLI task.

All CLIs Eliminated, and must Exit

Message Code: 6 (CM.EXT)

Text: No text accompanies this message.

Explanation: When a CLI /ELIM=* command is issued, this message is sent to all CLIs on the system that were initialized with the /MESSAGE subkeyword to the CLI /INIT command. A CLI that receives this message should exit as soon as possible, because a system shutdown or other emergency procedure may be taking place.

Terminal Linked to CLI

Message Code: 7 (CM.LKT)

Text: Displays the ASCII name and unit number (in octal) for the terminal. The length of the text requires 4 bytes (2 bytes for the name and 2 bytes for the number).

Explanation: This message is sent to a CLI when a terminal becomes associated with it. This message could result from a SET command processed by MCR, or any other use of the SCLI$ directive.

Terminal Removed from CLI

Message Code: 8. (CM.RMT)

Text: ASCII name and unit number (in octal) for the terminal. The length of the text requires 4 bytes (2 bytes for the name and 2 bytes for the number).

Explanation: This message is sent to a terminal's former CLI when it is switched to another CLI.

General Message

Message Code: 9. (CM.MSG)

Text: Name (ASCII) and unit number (in octal) of the terminal from which the CLI /MESSAGE command originated, followed by the text from the command string. The length is equal to the length of the string (possibly zero) plus the length of the terminal name and unit number (2 bytes each).

Explanation: This message is the result of a CLI /MESSAGE command. It contains the text the issuer of the command included in the string, plus the name and unit number of the terminal on which the command was typed.

17.3.4 Interacting with Serial Command Processing

Serial command processing is a mode in which commands from a terminal are executed in sequential order; each command is executed after the preceding command is finished. The default for command line processing on RSX-11M-PLUS and Micro/RSX operating systems is parallel processing; however, you may want to write a CLI that synchronizes commands. For example, if you want to be able to enter several commands from the same terminal, without waiting for each command to complete before entering another one, the CLI must save your input in a temporary buffer while the command that is currently executing completes. In other words, the CLI must synchronize the command processing.

To achieve *command synchronization,* use the terminal driver to store characters in a type-ahead buffer. Then, if you enter a command while another command is still outstanding, the terminal driver can store the characters temporarily. After processing of the current command completes, the CLI must issue a call to the terminal driver, requesting the characters stored in the buffer. The driver sends the characters to the CLI, which then processes them as the next command.

Command completion is signalled to the system when the final task involved in processing a command exits. If a task exits at the completion of command execution and the CLI does not issue a prompt request, (for example, if the command was terminated by an ESCAPE character), the task-exit code in the Executive calls the terminal driver directly. However, if the CLI generates a task-exit prompt, it must also call the terminal driver at the completion of the prompt request processing.

Note

The exiting task may be either the CLI or a task chained to by a RPOI$ or SDRP$ directive.

If the CLI is not receiving prompt requests, the terminal driver call is done automatically by the system, and the CLI is not involved. However, if you initialize the CLI to receive prompt requests, the system passes the prompt request on to the CLI as if it were a command. It is the CLI's responsibility to write a prompt string to the terminal and to call the terminal driver to initiate the next command. To call the terminal driver, the CLI issues an SNXC$ directive when it finishes processing the prompt request.

If a task does not exit (that is, if the task does not terminate the command execution), the system cannot detect command completion. So, to mark the end of the command and to cause the next command to be initiated, you must use the SNXC$ directive. Issuing this directive causes a task-exit prompt request to be generated if the task-exit would have generated one. If no prompt request is generated, SNXC$ calls the terminal driver to send the next command to the system for processing. This directive can be issued by the CLI or by the final task chained to it by an RPOI$ or SDRP$ directive.

When you use the SNXC$ directive, you do not need to verify whether or not a terminal is actually in serial mode. If the terminal is not in serial mode, the terminal driver sends commands to the system as they are entered; the directive does not cause the terminal driver to be called because it is not necessary. The terminal driver still generates a task-exit prompt request if one would have been generated when the task exited.

The SNXC$ directive clears the status bits that tell the system to generate a prompt request or to call the terminal driver at task exit. Therefore, a task exiting after issuing an SNXC$ directive does not cause two prompt requests to be issued.

See the *RSX-11M-PLUS and Micro/RSX Executive Reference Manual* for more information on the SNXC$ directive.

17.4 Sample CLI

This section illustrates a sample CLI, which is written in both MACRO-11 and FORTRAN. This sample CLI appears in a simplified form. It is meant to be a learning tool for writing a CLI. The same MACRO-11 and FORTRAN programs used here are included on your distribution kit in the files TMCLI.MAC and TMCLI.FTN, respectively. On RSX-11M-PLUS distribution kits, these files are located on pseudo device LB: in the named directory [USER]. On Micro/RSX distribution kits, these files are located on the diskette or the tape backup set labeled REFERENCE in the [USER] directory.

The primary function of the TMCLI sample CLI is to receive commands and pass them to MCR for execution. In addition, TMCLI features two single-letter commands: T to type a file and M to switch back to MCR mode. The comments in each sample listing suggest several modifications that you can make when designing or setting up a CLI.

17.4.1 MACRO-11 CLI

The following example is a sample CLI written in MACRO-11. It is also supplied with your distribution kit in the file LB:[USER]TMCLI.MAC.

```
        .TITLE TMCLI - EXAMPLE CLI THAT IMPLEMENTS "T" AND "M" COMMAND
        .IDENT  /1.00/
;
; THIS TASK (TMCLI) ILLUSTRATES THE USE OF THE CLI CONTROL DIRECTIVES OF
; RSX-11M-PLUS V4.0.
;
; WHEN SET UP AS A TERMINAL'S CLI, IT PROVIDES THE FOLLOWING COMMANDS:
;
;       T       ->      TYPE FILE
;       M       ->      SET TERMINAL TO MCR
;
; IF THE FIRST TWO LETTERS OF THE COMMAND ARE NOT "T" FOLLOWED
; BY A SPACE, OR "M" FOLLOWED BY A CARRIAGE RETURN,
; THE COMMAND IS PASSED TO MCR FOR EXECUTION. THEREFORE,
; USERS AT TERMINALS SET TO THIS CLI WILL SEE THE ENTIRE MCR COMMAND SET
; AS BEING PRESENT, PLUS THE "T" AND "M" COMMANDS. IF THE /DPR SUBKEYWORD
; IS INCLUDED ON THE CLI /INIT COMMAND, USERS WILL KNOW IMMEDIATELY IF
; THEIR TERMINAL IS SET TO THIS CLI BY THE DOUBLE RIGHT ANGLE BRACKET PROMPT.
;
; IF TMCLI IS INITIALIZED WITH THE /MESSAGE KEYWORD, TMCLI WILL
; GREET USERS WHEN THEIR TERMINAL IS SET TO THE CLI, AND SAY GOODBYE
; TO THEM WHEN THEY SWITCH AWAY.
;
; IN ORDER TO BRING THIS CLI UP ON THE SYSTEM, ISSUE THE FOLLOWING
; COMMAND SEQUENCE:
;
;>MAC TMCLI,TMCLI/-SP=TMCLI
;>TKB TMCLI,TMCLI/-SP=TMCLI,[1,1]EXELIB/LB
;>INS TMCLI/CLI=YES/TASK=...TMC
;>CLI /INIT=TMCLI/MESSAGE/DPR="<15><12>/>>/"/TASK=...TMC
;
;       TO SET TERMINALS TO THIS CLI, ISSUE THE FOLLOWING COMMAND
;       WHERE TTNN: IS REPLACED BY THE TERMINAL NUMBER OR TI:
```

```
;
;>SET /CLI=TTNN:TMCLI
;
; NOTE THAT THE PROMPT ISSUED IN RESPONSE TO A CONTROL/C DEFAULTS TO THE
; NAME OF THE CLI, WHICH IS "TMCLI>".
;
; TMCLI CAN BE ELIMINATED FROM THE SYSTEM BY THE FOLLOWING COMMANDS, ASSUMING
; NO TERMINALS ARE SET TO IT:
;
;>CLI /ELIM=TMCLI
;>REM ...TMC
;
; EXPERIMENT WITH THE OTHER SUBKEYWORDS OF THE CLI /INIT COMMAND, THE CLI
; /ENABLE AND CLI /DISABLE COMMANDS, AND WITH THE /MESSAGE SWITCH ON THE /INIT
; COMMAND TO OBSERVE THE EFFECTS.  ALSO TRY CHANGING THE /TASK=
; SWITCH ON THE INS AND CLI /INIT COMMAND TO "/TASK=TMCLI" AND NOTICE
; THAT ONLY ONE COPY OF THE TASK WILL EXIST FOR ALL TERMINALS, RATHER THAN
; ONE PER COMMAND. IF YOU DO THIS, YOU SHOULD ALSO CHANGE THE OPTION
; ON THE GCCI$ DIRECTIVE FROM GC.CEX TO GC.CST SO THE TASK WILL STAY ACTIVE
; BUT STOPPED WHEN NO COMMANDS ARE OUTSTANDING FOR IT.
;

        .MCALL  GCCI$,RPOI$,EXIT$S,QIOW$,ALUN$,DIR$
;
; DIRECTIVE DPBS
;
GCCI:   GCCI$   CMDBUF,CBLEN,IBUF,IBLEN,0,GC.CEX
RPOI:   RPOI$   MCR...
QIOW:   QIOW$   IO.WVB,5,1,,,,<,,40>
ALUN:   ALUN$   5

;
; BUFFER AREAS
;
CMDBUF: .BLKB   80.+G.CCBF      ;LENGTH = HEADER PLUS COMMAND IN BYTES
                                ;NOTE THAT ON M-PLUS THE MAXIMUM COMMAND
                                ;LENGTH IS 255. BYTES, SO THE LENGTH SHOULD
                                ;BE <255.+G.CCBF>OR CHARACTERS MAY BE LOST.

CBLEN=.-CMDBUF                  ;CALCULATE LENGTH
        .EVEN

IBUF:   .BLKB   G.CCCA+2        ;LONG ENOUGH TO CONTAIN ALL INFORMATION
IBLEN=.-IBUF                    ;CALCULATE LENGTH
        .EVEN

TCMD:   .ASCII  /PIP TI:=/      ;BASIC PIP COMMAND TO IMPLEMENT "T" COMMAND
TCLEN=.-TCMD                    ;LENGTH OF COMMAND
BUF:    .BLKB   80.-TCLEN       ;BUFFER TO GENERATE COMMANDS IN
        .EVEN

MCR:    .ASCII  /M/<15> ;STRING TO RECOGNIZE AS A SET TO MCR REQUEST
MCMD:   .ASCII  %SET /MCR=TI:%     ;SET TERMINAL TO MCR COMMAND
MCLEN=.-MCMD                       ;LENGTH OF COMMAND
        .EVEN
```

```
;
; TEXT TO BE DISPLAYED UPON RECEIPT OF NEW TERMINAL-LINKED OR
; TERMINAL-REMOVED MESSAGES FROM SYSTEM.
;
LKT:    .ASCII  <15><12>/WELCOME TO THE EXAMPLE CLI/<15><12>
LKTLEN=.-LKT
        .EVEN

RMT:    .ASCII  <15><12>/GOOD BYE FROM THE EXAMPLE CLI/<15><12>
RMTLEN=.-RMT
        .EVEN


;
; MESSAGE FOR CHARACTERS LOST DUE TO BUFFER BEING TOO SMALL
;
CHLOST: .ASCII  <15><12>/SORRY, PART OF YOUR COMMAND WAS LOST/<15><12>
CLLEN=.-CHLOST


;
; ILLEGAL COMMAND MESSAGE
;
ILL:    .ASCII  <15><12>/ILLEGAL COMMAND/<15><12>
ILLLEN=.-ILL
        .EVEN


;
; CLI CODE
;


        .ENABL  LSB
START:  DIR$    #GCCI                   ;GET A COMMAND FROM SYSTEM
        BCS     100$                    ;IF CS, DIRECTIVE ERROR
;
; IF THE GCCI$ DIRECTIVE HAD THE GC.CST OPTION SPECIFIED, WE MAY HAVE
; JUST BEEN UNSTOPPED, SO WE HAVE TO GET THE COMMAND.
;
        CMP     #IS.CLR,$DSW            ;WERE WE JUST UNSTOPPED?
        BEQ     START                   ;IF EQ YES
        MOV     #CMDBUF,R0              ;POINT TO COMMAND BUFFER
        TST     (R0)                    ;IS THIS A SYSTEM MESSAGE?
        BEQ     MSG                     ;IF EQ YES
        CMPB    G.CCCT(R0),G.CCCL(R0)   ;DID ENTIRE COMMAND FIT IN BUFFER
        BLO     150$                    ;IF LO NO
        MOV     #RPOI,R5                ;POINT TO DPB OF RPOI$ DIRECTIVE
;
; AT THIS POINT, THE CLI HAS A COMMAND TO BE PROCESSED. THE CODE FROM
; THE "START:" LABEL TO HERE SHOULD BE FAIRLY COMMON AMONG ALL CLIS.
; FROM THIS POINT ON, THE CODE WILL BE SPECIFIC TO THE PARTICULAR LANGUAGE
; THIS CLI IS IMPLEMENTING. COMMANDS CAN BE ADDED OR DELETED AT WILL
; BY MODIFYING CODE IN THIS SECTION. AS STATED ABOVE, THIS EXTREMELY
; SIMPLE CLI PROVIDES TWO COMMANDS, AND GIVES EVERYTHING ELSE TO MCR.
; THIS CREATES AN OPEN-ENDED ENVIRONMENT. REJECTING UNRECOGNIZED COMMANDS
; INSTEAD OF PASSING THEM TO MCR WOULD LIMIT THE TERMINAL USER'S USE OF
; THE SYSTEM TO ONLY THOSE OPERATIONS ALLOWED BY THE CLI. THIS IS
; A SECURE WAY OF CONTROLLING SYSTEM USE, IN CASES WHEN IT IS NECESSARY.
; IN THIS EXAMPLE, CHANGING THE "BNE 50$" BELOW, TO A "BNE 60$"
; WOULD CREATE AN ENVIRONMENT WHERE THE ONLY OPERATION A USER COULD
; PERFORM WOULD BE TO TYPE FILES ON THE TERMINAL, OR SWITCH BACK TO MCR.
; ALL OF THE REST OF THE SYSTEM IS INACCESSIBLE. IF NO MEANS OF SWITCHING
; BACK TO MCR IS PROVIDED, IT IS IMPOSSIBLE FOR THE USER TO DO
; ANYTHING THE CLI DOES NOT EXPLICITLY LET HIM DO.
;

        CMP     #"T ,G.CCBF(R0)         ;IS IT A "T" COMMAND?
        BEQ     10$                     ;IF EQ YES
        CMP     MCR,G.CCBF(R0)          ;IS IT AN "M" COMMAND?

;
; CHANGE THE NEXT INSTRUCTION TO "BNE 60$" TO ELIMINATE PASSING UNKNOWN
; COMMANDS TO MCR.
;
        BNE     50$                     ;IF NE NO
;
; THE USER HAS ENTERED AN "M" COMMAND, SO SET THE TERMINAL TO MCR.
;
        MOV     #MCMD,R.POBF(R5)        ;PUT ADDRESS OF CMD TO ISSUE IN DPB
        MOV     #MCLEN,R.POBL(R5)       ;PUT LENGTH OF CMD IN DPB
        BR      ISSCMD                  ;ISSUE COMMAND

;
; THE USER HAS ENTERED A "T" COMMAND SO CONSTRUCT THE NECESSARY PIP
; COMMAND AND SEND IT TO MCR.
;

10$:    MOV     R0,R1                   ;COPY ADDRESS OF BUFFER WITH COMMAND
        ADD     #G.CCBF+1,R1            ;POINT TO SPACE AFTER THE T
        MOV     #BUF,R2                 ;POINT TO BUFFER THAT WILL CONTAIN CMD
        MOV     #TCLEN,R4               ;GET LENGTH OF INITIAL PART OF COMMAND
20$:    MOVB    (R1)+,R3                ;GET NEXT CHARACTER
        CMPB    #15,R3                  ;IS IT THE END OF THE COMMAND
        BEQ     30$                     ;IF EQ YES
        CMPB    #40,R3                  ;IS IT A SPACE
        BEQ     20$                     ;IF EQ YES, SKIP IT
        MOVB    R3,(R2)+                ;PUT CHARACTER IN OUTPUT BUFFER
        INC     R4                      ;COUNT CHARACTER
        BR      20$                     ;GET NEXT CHARACTER
30$:    MOV     #TCMD,R.POBF(R5)        ;PUT ADDRESS OF CMD TO ISSUE IN DPB
        MOV     R4,R.POBL(R5)           ;INSERT LENGTH OF COMMAND
```

```
;
; THIS COMMAND WILL BE PASSED TO THE MAIN COMMAND DISPATCHER (MCR...),
; WHICH WILL START A COPY OF PIP AND GIVE IT THE COMMAND. THE RPOI
; DIRECTIVE COULD BE USED TO GIVE THE COMMAND DIRECTLY TO ...PIP AND
; GIVE THE RESULTING TASK A DIFFERENT NAME. IN THIS EXAMPLE,
; WE COULD CALL THE RESULTING PIP TASK "TYPTNN" WHERE THE LAST
; THREE CHARACTERS (TNN) ARE REPLACED BY A DEVICE NAME CHARACTER AND
; UNIT NUMBER. FOR EXAMPLE, IF THE "T" COMMAND WAS TYPED ON TT10:
; THE RESULTING TASK NAME COULD BE MADE TO BE "TYPT10". THIS HAS THE
; ADVANTAGE OF ALLOWING THE TASK TO BE ABORTED BY AN "ABO TYP" COMMAND,
; INSTEAD OF "ABO PIP", THUS THE USER WOULD NOT HAVE TO KNOW THAT THE
; "T" COMMAND HE ISSUED IS ACTUALLY BEING SERVICED BY THE PIP TASK. THIS
; IS EXACTLY WHAT THE DCL CLI DOES. IT IS NOT DONE HERE
; TO SIMPLIFY THE EXAMPLE. PICKING UP THE TERMINAL NAME AND UNIT
; NUMBER, CONVERTING IT TO RAD50, AND SUPPLYING IT IN THE RPOI DIRECTIVE
; ALONG WITH A DIRECT REQUEST OF PIP IS LEFT AS AN EXERCISE TO THE USER.
; (BE CAREFUL ON RSX-11M-PLUS SYSTEMS WITH MORE THAN 64. TERMINALS).
; NOTE THAT REQUESTING PIP DIRECTLY AVOIDS A SUBSTANTIAL AMOUNT OF SYSTEM
; OVERHEAD.
;
        BR      ISSCMD                          ;EXECUTE COMMAND


;
; COMMAND IS NOT "T" OR "M", SO GIVE IT TO MCR AS IS.
;
50$:    MOV     #CMDBUF+G.CCBF,R.POBF(R5)       ;PUT ADDRESS OF STRING IN DPB
        MOVB    G.CCCT(RO),R.POBL(R5)           ;PUT LENGTH IN RPOI DPB


;
; ISSUE RPOI$ DIRECTIVE TO EXECUTE COMMAND
;
ISSCMD: MOV     (RO),R.PODV(R5)         ;PUT NAME OF ISSUING DEVICE IN RPOI
        MOVB    G.CCUN(RO),R.POUN(R5)   ;COPY UNIT NUMBER
        MOVB    IBUF+G.CCPU,R.POUM(R5)  ;PROPAGATE UIC OF TERMINAL
        MOVB    IBUF+G.CCPU+1,R.POUG(R5)
        MOV     IBUF+G.CCOA,R.POOA(R5)  ;COPY ID PARENT IN CASE WE WERE SPAWNED
        DIR$    #RPOI                   ;PASS COMMAND TO EXECUTOR TASK
        BCC     START                   ;IF CC, SUCCESS
        BR      110$                    ;ERROR


;
; CONTROL IS TRANSFERRED HERE IF UNKNOWN COMMANDS ARE TO BE REJECTED
; INSTEAD OF BEING PASSED TO MCR.
;
60$:    MOV     #ILL,R1                 ;POINT TO MESSAGE TO BE PRINTED
        MOV     #ILLLEN,R2              ;GET LENGTH OF MESSAGE
        CALL    ISSMSG                  ;ISSUE MESSAGE
        BR      START                   ;TRY FOR NEXT COMMAND


;
; ERROR HANDLING CODE
;

100$:   CLR     RO                      ;FATAL ERROR ON GCCI$
        BR      150$
110$:   MOV     #1,RO                   ;FATAL ERROR ON RPOI$
120$:   IOT                             ;FATAL
```

```
150$:   MOV     #CHLOST,R1              ;POINT TO MESSAGE TO BE PRINTED
        MOV     #CLLEN,R2               ;GET LENGTH OF MESSAGE
        CALL    ISSMSG                  ;ISSUE MESSAGE
        BR      START
        .DSABL  LSB

;
; A SYSTEM MESSAGE WAS RECEIVED INSTEAD OF A COMMAND.
;

MSG:    CMPB    #CM.LKT,G.CCUN(R0)      ;IS IT A NEW TERMINAL LINKED TO US?
        BEQ     10$                     ;IF EQ YES
        CMPB    #CM.RMT,G.CCUN(R0)      ;IS IT A TERMINAL BEING REMOVED
        BEQ     20$                     ;IF EQ YES
        CMPB    #CM.EXT,G.CCUN(R0)      ;IS IT AN ELIMINATE ALL PACKET
        BEQ     50$                     ;IF EQ YES
        CMPB    #CM.ELM,G.CCUN(R0)      ;IS IT A CLI ELIMINATION MESSAGE
        BEQ     50$                     ;IF EQ YES
        BR      START                   ;IGNORE ALL OTHER MESSAGES

;
; ISSUE WELCOME MESSAGE
;
10$:    MOV     #LKT,R1                 ;POINT TO MESSAGE TEXT
        MOV     #LKTLEN,R2              ;GET LENGTH
        ADD     #G.CCBF,R0              ;POINT TO TERMINAL THAT WAS SET
        BR      30$                     ;

;
; ISSUE GOODBYE MESSAGE
;
20$:    MOV     #RMT,R1                 ;POINT TO MESSAGE TEXT
        MOV     #RMTLEN,R2              ;GET LENGTH
        ADD     #G.CCBF,R0              ;POINT TO TERMINAL NAME
30$:    CALL    ISSMSG                  ;ISSUE MESSAGE
        JMP     START                   ;GET NEXT COMMAND
;
; WHEN AN ELIMINATE ALL MESSAGE IS RECEIVED, EXIT IMMEDIATELY.
;
50$:    EXIT$S
```

```
;
; ROUTINE TO PRINT TO TERMINAL
;
;
; INPUTS:
;
;       R0=ADDRESS OF NAME OF TERMINAL TO PRINT TO
;       R1=ADDRESS OF MESSAGE TO PRINT
;       R2=LENGTH OF MESSAGE
;
; OUTPUTS:
;
;       NONE
;
; THIS ROUTINE PRINTS THE SPECIFIED MESSAGE ON THE SPECIFIED TERMINAL.
; ALL ERRORS THAT MIGHT RESULT ARE IGNORED. IN GENERAL, CLIS SHOULD
; DO ASYNCHRONOUS I/O WITH COMPLETION ASTS, AND PROVIDE A TIME-OUT
; FACILITY TO KILL THE I/O IF IT HASN'T COMPLETED IN A CERTAIN
; AMOUNT OF TIME. THIS PREVENTS TERMINAL I/O FROM HOLDING UP
; COMMAND PROCESSING. THIS IS NOT DONE HERE FOR SIMPLICITY.
;

ISSMSG: MOV     (R0),ALUN+A.LUNA        ;SET UP DEVICE NAME
        MOVB    2(R0),ALUN+A.LUNU       ;AND UNIT NUMBER
        CLRB    ALUN+A.LUNU+1           ;CLEAR HIGH BYTE
        DIR$    #ALUN                   ;ASSIGN LUN 5 TO THE SPECIFIED DEVICE
        BCS     10$                     ;IF CS, SKIP I/O
        MOV     R1,QIOW+Q.IOPL          ;SET ADDRESS OF MESSAGE
        MOV     R2,QIOW+Q.IOPL+2        ;INSERT LENGTH OF MESSAGE
        DIR$    #QIOW                   ;DO I/O AND WAIT FOR IT
10$:    RETURN

        .END    START
```

17.4.2 FORTRAN CLI

The following example is a sample CLI written in FORTRAN. It is also supplied with your distribution kit in the file LB:[USER]TMCLI.FTN.

```
C
C THIS TASK (TMCLI) ILLUSTRATES THE USE OF THE CLI CONTROL DIRECTIVES OF
C RSX-11M-PLUS V4.0.
C
C WHEN SET UP AS A TERMINAL'S CLI, IT PROVIDES THE FOLLOWING COMMANDS:
C
C       T       ->     TYPE FILE
C       M       ->     SET TERMINAL TO MCR
C
C IF THE FIRST TWO LETTERS OF THE COMMAND ARE NOT "T" FOLLOWED
C BY A SPACE, OR "M" FOLLOWED BY A CARRIAGE RETURN,
C THE COMMAND IS PASSED TO MCR FOR EXECUTION. THEREFORE,
C USERS AT TERMINALS SET TO THIS CLI WILL SEE THE ENTIRE MCR COMMAND SET
C AS BEING PRESENT, PLUS THE "T" AND "M" COMMANDS. IF THE /DPR SUBKEYWORD
C IS INCLUDED ON THE CLI /INIT COMMAND, USERS WILL KNOW IMMEDIATELY IF
C THEIR TERMINAL IS SET TO THIS CLI BY THE DOUBLE RIGHT ANGLE BRACKET PROMPT.
C
C IF TMCLI IS INITIALIZED WITH THE /MESSAGE KEYWORD, TMCLI WILL
C GREET USERS WHEN THEIR TERMINAL IS SET TO THE CLI, AND SAY GOODBYE
C TO THEM WHEN THEY SWITCH AWAY.
C
C IN ORDER TO BRING THIS CLI UP ON THE SYSTEM, ISSUE THE FOLLOWING
C COMMAND SEQUENCE:
C
C>FOR TMCLI,TMCLI/-SP=TMCLI
C>TKB TMCLI/FP,TMCLI/-SP=TMCLI,[1,1]FOROTS/LB
C>INS TMCLI/CLI=YES/TASK=...TMC
C>CLI /INIT=TMCLI/MESSAGE/DPR="<15><12>/>>/"/TASK=...TMC
C
C IF USING FORTRAN IV-PLUS INSTEAD OF FORTRAN IV, THE FIRST TWO
C COMMANDS ABOVE WOULD BE:
C
C>F4P TMCLI,TMCLI/-SP=TMCLI.FTN
C>TKB TMCLI/FP,TMCLI/-SP=TMCLI,[1,1]F4POTS/LB
C
C TO SET TERMINALS TO THIS CLI, ISSUE THE FOLLOWING COMMAND
C WHERE TTNN: IS REPLACED BY THE TERMINAL NUMBER OR TI:
C
C>SET /CLI=TTNN:TMCLI
C
C NOTE THAT THE PROMPT ISSUED IN RESPONSE TO A CONTROL/C DEFAULTS TO THE
C NAME OF THE CLI, WHICH IS "TMCLI>".
C
C TMCLI CAN BE ELIMINATED FROM THE SYSTEM BY THE FOLLOWING COMMANDS, ASSUMING
C NO TERMINALS ARE SET TO IT:

C
C>CLI /ELIM=TMCLI
C>REM ...TMC

C
C EXPERIMENT WITH THE OTHER SUBKEYWORDS OF THE CLI /INIT COMMAND, THE CLI
C /ENABLE AND CLI /DISABLE COMMANDS, AND WITH THE /MESSAGE SWITCH ON THE /INIT
C COMMAND TO OBSERVE THE EFFECTS. ALSO TRY CHANGING THE /TASK=
C SWITCH ON THE INS AND CLI /INIT COMMAND TO "/TASK=TMCLI" AND NOTICE
C THAT ONLY ONE COPY OF THE TASK WILL EXIST FOR ALL TERMINALS, RATHER THAN
C ONE PER COMMAND. IF YOU DO THIS, YOU SHOULD ALSO CHANGE THE OPTION
C ON THE GTCMCI DIRECTIVE FROM GC.CEX TO GC.CST SO THE TASK WILL STAY ACTIVE
C BUT STOPPED WHEN NO COMMANDS ARE OUTSTANDING FOR IT.
C
```

```
C
C BUFFER AREAS
C
C    COMMAND BUFFER - THIS BUFFER IS USED IN THE GTCMCI DIRECTIVE.  IT CON-
C                     TAINS INFORMATION ON THE COMMAND RECEIVED (8 BYTES)
C                     AND THE COMMAND TEXT.
      BYTE CMDBUF(264)
      INTEGER*2 CBLEN         ! NOTE THAT ON M+ THE MAXIMUM COMMAND
      DATA CBLEN/263/         ! LENGTH IS 255.
      INTEGER*2 GCCDV, GCCCL  ! ASCII DEV NAM, # CHARS IN CMD LINE
      BYTE GCCCT, GCCUN, GCCBF(255) ! # OF CHARS, TERMINAL UNIT #, CMD
      EQUIVALENCE (GCCDV,CMDBUF), (GCCCL,CMDBUF(5)), (GCCCT,CMDBUF(4)),
      1           (GCCBF,CMDBUF(9)), (GCCUN,CMDBUF(3))
C
C    INFORMATION BUFFER - ALSO RETURNED IN THE GTCMCI DIRECTIVE.
C
      BYTE IBUF(14)
      INTEGER*2 GCCPU
      DATA IBLEN/14/,GCCPU/9/ ! LENGTH OF IBUF, LOGIN UIC OFFSET
      INTEGER*4 GCCOA         ! ADDR OF OFFSPRING CONTROL BLOCK FROM PARENT
      EQUIVALENCE (GCCOA,IBUF(7))
C
C    BUFFER IN WHICH TO GENERATE COMMANDS
C
      BYTE BUF(72)            ! BUFFER IN WHICH TO GENERATE COMMANDS
      REAL*8 TCMD(2)          ! FOR CONVENIENCE IN ASSIGNMENT STATEMENTS
      EQUIVALENCE (BUF,TCMD)
C
C    MCR TASK NAME IN RAD50
C
      INTEGER*4 IMCR          ! MCR TASK NAME IN RAD50
      DATA IMCR/6RMCR.../

C
C DIRECTIVE SYMBOLIC VALUE DEFINITIONS
C
      INTEGER*2 GCCCS, GCCEX, GCCST, GCCND
      DATA GCCCS, GCCEX, GCCST, GCCND /0, 1, 2, 128/  ! INCP VALUES IN GTCMCI
      DATA ISCLR /0/          ! IS.CLR RETURN CODE
      INTEGER*2 CMELM, CMEXT, CMLKT, CMRMT  ! TERMINAL CODES FOR SYSTEM MSGS
      DATA CMELM, CMEXT, CMLKT, CMRMT /5, 6, 7, 8/

C
C CLI CODE
C
C TO FORCE THE CLI TO EXIT INSTEAD OF RETURNING, THE GC.CEX OPTION IS SPECI-
C FIED IN THE GTCMCI DIRECTIVE BELOW.  IF THE CLI IS INSTALLED SO THAT ONLY
C ONE COPY OF THE TASK WILL EXIST FOR ALL TERMINALS, CHANGE THE VARIABLE
C GCCEX BELOW TO GCCST.  THIS SPECIFIES THE GC.CST OPTION SO THAT THE TASK
C WILL STAY ACTIVE BUT STOPPED WHEN NO COMMANDS ARE OUTSTANDING FOR IT.
C
   1  CALL GTCMCI(CMDBUF,CBLEN,IBUF,IBLEN,0,GCCEX,IDS) ! GET CMD FOR CLI
      IF (IDS .LT. 0) GO TO 100

C
C IF THE GTCMCI DIRECTIVE HAD THE GC.CST OPTION SPECIFIED, WE MAY HAVE
C JUST BEEN UNSTOPPED, SO WE HAVE TO GET THE COMMAND.
C
      IF (IDS .EQ. ISCLR) GO TO 1     ! WERE WE JUST UNSTOPPED?
```

```
        IF (GCCDV .EQ. 0) GO TO 200     ! IS THIS A SYSTEM MESSAGE?

        IGCCUN = GCCUN                  ! CONVERT BYTE TO INTEGER (UNIT #)
        CALL ASNLUN(5,GCCDV,IGCCUN,IDS) ! ASSIGN TERMINAL TO LUN 5
        IGCCCT=GCCCT                    ! CONVERT BYTE TO INTEGER
        IF (IGCCCT .GE. 0) GO TO 5
        IGCCCT = 256 + IGCCCT           ! CONVERT NEGATIVE TO UNSIGNED
   5    IF (IGCCCT .LT. GCCCL) GOTO 150 ! DID ENTIRE COMMAND FIT IN BUFFER?
C
C AT THIS POINT, THE CLI HAS A COMMAND TO BE PROCESSED. THE CODE FROM
C THE LABEL 1 TO HERE SHOULD BE FAIRLY COMMON AMONG ALL CLIS.
C FROM THIS POINT ON, THE CODE WILL BE SPECIFIC TO THE PARTICULAR LANGUAGE
C THIS CLI IS IMPLEMENTING. COMMANDS CAN BE ADDED OR DELETED AT WILL
C BY MODIFYING CODE IN THIS SECTION. AS STATED ABOVE, THIS EXTREMELY
C SIMPLE CLI PROVIDES TWO COMMANDS, AND GIVES EVERYTHING ELSE TO MCR.
C THIS CREATES AN OPEN-ENDED ENVIRONMENT. REJECTING UNRECOGNIZED COMMANDS
C INSTEAD OF PASSING THEM TO MCR WOULD LIMIT THE TERMINAL USER'S USE OF
C THE SYSTEM TO ONLY THOSE OPERATIONS ALLOWED BY THE CLI. THIS IS
C A SECURE WAY OF CONTROLLING SYSTEM USE, IN CASES WHEN IT IS NECESSARY.
C IN THIS EXAMPLE, CHANGING THE "GO TO 50" BELOW, TO A "GO TO 60"
C WOULD CREATE AN ENVIRONMENT WHERE THE ONLY OPERATION A USER COULD
C PERFORM WOULD BE TO TYPE FILES ON THE TERMINAL, OR SWITCH BACK TO MCR.
C ALL OF THE REST OF THE SYSTEM IS INACCESSIBLE. IF NO MEANS OF SWITCHING
C BACK TO MCR IS PROVIDED, IT IS IMPOSSIBLE FOR THE USER TO DO
C ANYTHING THE CLI DOES NOT EXPLICITLY LET HIM DO.

C
C IS IT A "T" COMMAND?
C
        IF ((GCCBF(1) .EQ. 'T') .AND. (GCCBF(2) .EQ. ' ')) GO TO 10
C
C CHANGE THE NEXT IF STATEMENT TO "GO TO 60" TO ELIMINATE PASSING UNKNOWN
C COMMANDS TO MCR.
C
C IS IT ALSO NOT AN "M" COMMAND?
C
        IF ((GCCBF(1) .NE. 'M') .OR. (GCCBF(2) .NE. 13)) GO TO 50 ! M<CR>

C
C THE USER HAS ENTERED AN "M" COMMAND, SO SET THE TERMINAL TO MCR.
C
        TCMD(1) = 'SET /MCR'
        TCMD(2) = '=TI:    '    ! SET /MCR=TI:
        IGRP = IBUF(GCCPU+1)    ! (CONVERT BYTE TO INTEGER) GROUP
        IMEM = IBUF(GCCPU)      ! MEMBER PART OF UIC

C
C ISSUE RPOI$ DIRECTIVE TO EXECUTE COMMAND
C
        CALL RPOI(IMCR,IGRP,IMEM,,TCMD,12,,GCCDV,IGCCUN,,GCCOA,IDS)
        IF (IDS .LE. 0) GO TO 110 ! ERROR?
        GO TO 1                 ! BACK TO START
```

```
C
C THE USER HAS ENTERED A "T" COMMAND SO CONSTRUCT THE NECESSARY PIP
C COMMAND AND SEND IT TO MCR
C
  10    TCMD(1) = 'PIP TI:='    ! PUT BEGINNING OF PIP CMD IN BUFFER
        I2 = 8                  ! POINTER INTO TARGET BUFFER
        DO 20, I1 = 10, CBLEN   ! BEGIN AT SPACE AFTER THE T IN CMDBUF
          IF (CMDBUF(I1) .EQ. 13) GO TO 30 ! END OF COMMAND? (<CR>)
          IF (CMDBUF(I1) .EQ. ' ') GO TO 20 ! SKIP ANY SPACES
          I2 = I2 + 1           ! COUNT CHARACTER
          BUF(I2) = CMDBUF(I1)  ! COPY INTO OUTPUT BUFFER
  20    CONTINUE
  30    IGRP = IBUF(GCCPU+1)    ! (CONVERT BYTE TO INTEGER) GROUP
        IMEM = IBUF(GCCPU)      ! MEMBER PART OF UIC

C
C ISSUE RPOI$ DIRECTIVE TO EXECUTE COMMAND
C
        CALL RPOI(IMCR,IGRP,IMEM,,TCMD,I2,,GCCDV,IGCCUN,,GCCOA,IDS)
        IF (IDS .LE. 0) GO TO 110 ! ERROR?
        GO TO 1                 ! BACK TO START

C
C IN THE ABOVE SECTION OF CODE, THE RPOI DIRECTIVE WAS USED TO PASS
C THE PIP COMMAND TO THE MAIN COMMAND DISPATCHER (MCR...),
C WHICH WILL START A COPY OF PIP AND GIVE IT THE COMMAND. THE RPOI
C DIRECTIVE COULD BE USED TO GIVE THE COMMAND DIRECTLY TO ...PIP AND
C GIVE THE RESULTING TASK A DIFFERENT NAME. IN THIS EXAMPLE,
C WE COULD CALL THE RESULTING PIP TASK "TYPTNN" WHERE THE LAST
C THREE CHARACTERS (TNN) ARE REPLACED BY A DEVICE NAME CHARACTER AND
C UNIT NUMBER. FOR EXAMPLE, IF THE "T" COMMAND WAS TYPED ON TT10:
C THE RESULTING TASK NAME COULD BE MADE TO BE "TYPT10". THIS HAS THE
C ADVANTAGE OF ALLOWING THE TASK TO BE ABORTED BY AN "ABO TYP" COMMAND,
C INSTEAD OF "ABO PIP"; THUS THE USER WOULD NOT HAVE TO KNOW THAT THE
C "T" COMMAND HE ISSUED IS ACTUALLY BEING SERVICED BY THE PIP TASK. THIS
C IS EXACTLY WHAT THE DCL CLI DOES. IT IS NOT DONE HERE IN ORDER
C TO SIMPLIFY THE EXAMPLE. PICKING UP THE TERMINAL NAME AND UNIT
C NUMBER, CONVERTING IT TO RAD50, AND SUPPLYING IT IN THE RPOI DIRECTIVE
C ALONG WITH A DIRECT REQUEST OF PIP IS LEFT AS AN EXERCISE TO THE USER.
C (BE CAREFUL ON RSX-11M-PLUS SYSTEMS WITH MORE THAN 64. TERMINALS).
C NOTE THAT REQUESTING PIP DIRECTLY AVOIDS A SUBSTANTIAL AMOUNT OF SYSTEM
C OVERHEAD.
C

C
C
C COMMAND IS NOT "T" OR "M", SO GIVE IT TO MCR AS IS.
C
  50    IGRP = IBUF(GCCPU+1)    ! (CONVERT BYTE TO INTEGER) GROUP
        IMEM = IBUF(GCCPU)      ! MEMBER PART OF UIC
        IGCCCT = GCCCT          ! # OF CHARS IN COMMAND

C
C ISSUE RPOI$ DIRECTIVE TO EXECUTE COMMAND
C
        CALL RPOI(IMCR,IGRP,IMEM,,GCCBF,IGCCCT,,GCCDV,IGCCUN,,GCCOA,IDS)
        IF (IDS .LE. 0) GO TO 110 ! ERROR?
        GO TO 1                 ! BACK TO START
```

```
C
C CONTROL IS TRANSFERRED HERE IF UNKNOWN COMMANDS ARE TO BE REJECTED
C INSTEAD OF BEING PASSED TO MCR.
C
  60    WRITE (5,62)
  62    FORMAT('OILLEGAL COMMAND')
        GO TO 1                 ! BACK TO START

C
C ERROR HANDLING CODE
C
 100    WRITE (5,102) IDS       ! ERROR IN GTCMCI
 102    FORMAT('OFATAL ERROR IN GTCMCI DIRECTIVE.  DSW=',I6)
        GO TO 250               ! EXIT
C
 110    WRITE (5,112) IDS       ! ERROR IN RPOI
 112    FORMAT('OFATAL ERROR IN RPOI DIRECTIVE.  DSW=',I6)
        GO TO 250               ! EXIT
C
 150    WRITE (5,152)           ! ENTIRE COMMAND DID NOT FIT IN BUFFER
 152    FORMAT('OSORRY, PART OF YOUR COMMAND WAS LOST')
        GO TO 1                 ! BACK TO START

C
C A SYSTEM MESSAGE WAS RECEIVED INSTEAD OF A COMMAND
C
 200    CONTINUE
        IF (GCCUN .EQ. CMLKT) GO TO 210 ! IS IT A NEW TERMINAL LINKED TO US?
        IF (GCCUN .EQ. CMRMT) GO TO 220 ! IS IT A TERMINAL BEING REMOVED?
        IF (GCCUN .EQ. CMEXT) GO TO 250 ! IS IT AN ELIMINATE ALL PACKET?
        IF (GCCUN .EQ. CMELM) GO TO 250 ! IS IT A CLI ELIMINATION MESSAGE?
        GO TO 1                 ! IGNORE ALL OTHER MESSAGES

C
C ISSUE WELCOME MESSAGE
C
 210    IGCCUN = CMDBUF(11)     ! CONVERT BYTE TERMINAL UNIT # TO INTEGER
        CALL ASNLUN(5,GCCBF,IGCCUN,IDS) ! ASSIGN LUN 5 TO SPECIFIED DEVICE
        WRITE (5,212)
 212    FORMAT('OWELCOME TO THE EXAMPLE CLI')
        GO TO 1                 ! BACK TO START

C
C ISSUE GOODBYE MESSAGE
C
 220    IGCCUN = CMDBUF(11)     ! CONVERT BYTE TERMINAL UNIT # TO INTEGER
        CALL ASNLUN(5,GCCBF,IGCCUN,IDS) ! ASSIGN LUN 5 TO SPECIFIED DEVICE
        WRITE (5,222)
 222    FORMAT('OGOOD BYE FROM THE EXAMPLE CLI')
        GO TO 1                 ! BACK TO START
C
C WHEN AN ELIMINATE ALL MESSAGE IS RECEIVED, EXIT IMMEDIATELY
C
 250    CONTINUE
        CALL EXIT
        END
```

Chapter 18
The DCL Task

The DIGITAL Command Language (DCL) is a general-purpose command language syntax implemented on several DIGITAL operating systems. The DCL task, in conjunction with the CLI dispatcher, provides support for DCL.

Note

> Micro/RSX distribution kits do not include the source files for the DCL task. Therefore, the information in this chapter does not apply to Micro/RSX systems.

This chapter describes several methods for installing the DCL task on your system. It also includes an explanation of how each method affects the parsing and translation of DCL commands. In addition, the chapter contains information on the Macro Metalanguage (MML), a set of MACRO-11 macros used to define DCL. Using MML, an experienced programmer can add, modify, or remove DCL commands.

18.1 Introduction to the DCL Task

The DCL task receives DCL commands from a command line interpreter (CLI) command dispatcher. After the dispatcher sends the commands to the DCL task, DCL translates them to equivalent Monitor Console Routine (MCR) commands. The MCR commands are then sent to the MCR CLI for interpretation and execution.

For additional information on DCL and MCR syntax or on the Executive directives used to implement alternate CLIs, see the following manuals:

- *RSX-11M-PLUS Command Language Manual* (DCL syntax and functions)
- *RSX-11M-PLUS MCR Operations Manual* (MCR syntax and functions)
- *RSX-11M-PLUS and Micro/RSX Executive Reference Manual* (Executive directives used to implement alternate CLIs)

Also, for a general description of the CLI command dispatcher and alternate CLIs, see Chapter 17.

The examples of source code in this chapter illustrate features of DCL and the DCL task, but the examples are changed from the actual source code for clarity. See the DCL source code in directory [23,10] for exact information. As a general example of DCL coding, the command tables for the ASSIGN command are also included in this chapter (see Section 18.5.1).

18.2 Installing the DCL Task

You can install the DCL task in any of the following ways:

- As an alternate CLI
- As a conventional task (installed with the install-run-remove option of the RUN command)
- As an MCR prototype task (installed with the task name ...DCL)
- As a catchall task (installed with the task name ...CA.)

The method you select depends upon the requirements of your system users and your installation. If you run DCL as a task (an uninstalled version, a prototype task, or a catchall task), you can use a subset of DCL features. However, the following restrictions apply:

- You cannot use the LOGIN command to log in to the sytem.
- You cannot change your defaults with the SET DEFAULT command.
- Breakthrough write requests do not break through.
- The system does not rename spawned tasks (for example, the DCL command DIRECTORY spawns a task named PIPTnn, not DIRTnn).

These restrictions do not apply if you install DCL as an alternate CLI.

18.2.1 Installing DCL as an Alternate CLI

Installing DCL as an alternate CLI is the most common way to use DCL. This allows you to classify users by CLI in the system account file. Users can log in using the HELLO or LOGIN commands. The HELLO task checks the account file for the CLI and sets the user's terminal to the designated CLI.

If you include support for alternate CLIs during system generation, you can use the following Executive directives:

- AST Service Exit (ASTX$)
- Get Command for Command Interpreter (GCCI$)
- Get Command Interpreter Information (GCII$)
- Get Task Parameters (GTSK$)
- Parent/Offspring Tasking directives
  - Connect (CNCT$)
  - Request and Pass Offspring Information (RPOI$)
  - Send, Request, and Connect (SDRC$)

    - Send Data Request and Pass Offspring Control Block (SDRP$)
    - Spawn (SPWN$)
- Run Task (RUN$)
- Stop (STOP$)
- Specify Command Arrival asynchronous system trap (AST; SCAA$)
- Specify Requested Exit AST (SREA$ or SREX$)

The GCCI$, GCII$, RPOI$, and SCAA$ directives are specifically added for alternate CLI support. For more information, see Chapter 17 and the *RSX-11M-PLUS and Micro/RSX Executive Reference Manual.*

To install DCL as an alternate CLI, the system uses the following two commands:

```
MCR>
MCR>
```

Unless you modify the system files SYSVMR.CMD and STARTUP.CMD, these commands are issued for you during the system generation and startup procedures. When you run the Virtual Monitor Console Routine (VMR) to create the system image file, SYSVMR.CMD issues the INSTALL command. The CLI /INIT command is included in the system STARTUP.CMD file.

18.2.2 Running DCL as a Task

When DCL runs from your terminal as a task, the terminal remains set to MCR. All DCL commands work in the normal way, but a carriage return on a line by itself causes the task to exit.

To run DCL as an uninstalled task, enter the RUN command, as follows:

```
>
```

If the task image file DCL.TSK is in the directory [3,54] on device LB:, precede the file name with a dollar sign ($) and omit the directory and device specification, as follows:

```
>
```

By default, the RUN command searches for the file in the library directory on device LB:.

18.2.3 Installing DCL as a Prototype Task

To install DCL as a prototype task, enter the following command line:

```
>
```

When you install DCL as a prototype task, you can enter DCL commands from an MCR terminal by preceding the command with the letters DCL. For example:

```
MCR>
```

In this example, MCR passes the command SHOW USERS to DCL. However, the terminal remains set to MCR.

18.2.4 Installing DCL as a Catchall Task

To install DCL as a catchall task, use the following command line:

```
>
```

When you install DCL with the name ...CA., all commands not recognized by MCR are passed to DCL; DCL runs as the "catchall" task. To enter DCL commands that are identical to MCR commands, such as MOUNT and SET, use valid MCR syntax. If you use DCL syntax to enter one of these commands, MCR returns an error message.

Note

> To avoid an infinite loop, do not install DCL as the catchall task for MCR if MCR is already installed as the catchall task for DCL.

For more information on catchall tasks, see Chapter 19.

18.3 Using DCL to Perform Flying Installs

The default DCL task returns an error for any command not recognized as a DCL command. However, tasks need not be installed for DCL to access them. You can modify DCL to allow any unrecognized commands to fall through to MCR for another chance at execution. This allows you to access MCR tasks and utilities without using the MCR or RUN commands. Instead, when DCL encounters a command it does not recognize, it attempts to install a task named after the command; in other words, DCL performs a *flying install.*

To allow DCL to perform flying installs (or to prevent this from occurring), modify the file [1,24]DCLBLD.CMD by following the instructions contained in the file. Then, rebuild the DCL task.

Note that the DCL command MCR is included in the default DCL task. This command allows you to issue a one-line MCR command without leaving DCL. The MCR command is not affected by the SET DEBUG command. In addition, you can enter any of the DCL SET commands from MCR. MCR has been modified to pass to DCL all SET commands that do not have a slash (/) in front of the keyword.

18.4 Task Building DCL

All of the changes to DCL described in this chapter require that the DCL task be rebuilt. The overlay structure of the DCL task uses a task-build process that requires two passes by the Task Builder (TKB) before it is complete. The first pass creates a symbol table (file type STB). During the second pass, TKB uses the STB file to build the DCL task.

Note

> The file [1,24]DCLBLD.CMD is included by the system generation procedure when you request DCL support. You should read this file before attempting to task-build DCL.

The following subsections summarize the task-build process.

18.4.1 The First-Pass Task Build

The following list summarizes the first task build:

- Each DCL command is defined in a parser table (PT), which defines the DCL syntax of the command.
- The PT is paired with a translator table (TT), which defines the MCR equivalents for the DCL command.
- The addresses of the PT and TT are included in a master command table (MCT), which lists all DCL commands.
- When DCL parses a command, it seeks the address of that command in the MCT. After locating the address, DCL uses the information in the PT and TT to translate the DCL command into an equivalent MCR command.
- The parser tables and translator tables are overlaid on the MCT; at the time of the first task build, the entries in the MCT are undefined symbols.

To execute the first task build, enter the following command line:

```
MCR>
```

This results in one or more undefined symbols for each command, but the STB file contains the PT/TT addresses for each command.

18.4.2 The Second-Pass Task Build

The following definitions summarize the second task build:

- The MCT symbols are defined by the STB file.
- The addresses in the PT/TT are doubly defined, once in the STB and once in their overlays. The two definitions are identical.

  This part of the process is accomplished through two Overlay Description Language (ODL) files (see the file DCLBLD.CMD for more information).

To execute the second task build, enter the following command line:

```
>
```

18.5 Understanding How DCL Interprets Commands

There are three steps involved in interpreting DCL commands. They are as follows:

1. Receiving the DCL command line
2. Parsing and translating the DCL command line to an equivalent MCR command line
3. Sending the equivalent MCR command line to the system for execution

The first step involves two tasks: the CLI dispatcher task (MCR...) and the DCL task. The CLI dispatcher task gets the command line from the terminal driver, places it in a CLI queue, and activates the DCL task (if it is not already active). The activated DCL task is named DCLTnn, where nn is the number of the issuing terminal. The nonprivileged DCL task issues the

Executive directive Get Command for Command Interpreter (GCCI$) to remove the command line from the CLI queue.

During the second step, the DCL command line is translated to an equivalent MCR command line. The translation involves two processes: the parser process and the translator process. The *parser process* checks syntax to ensure that the received command line is a valid DCL command line. Then, it constructs an intermediate form (IFORM) to pass data about the command line to the translator. The *translator process* uses this data to construct an equivalent MCR command line from the valid DCL command line.

The final step sends the equivalent MCR command line to the system for further interpretation and execution. This process dispatches the equivalent MCR command line to MCR (or to MCR utility tasks) with the Executive directive Request and Pass Offspring Information (RPOI$).

The RPOI$ directive is similar to the Exective directive Spawn Task (SPWN$); however, they differ in the following ways:

- RPOI$ allows the spawned task to be named. (For example, if the system spawns the MCR task PIP, it names the task COPTnn, which is the task name for the DCL command COPY.) The DCL task requires the task-naming ability to record the association between the DCL command name and the MCR or utility task that executes the command.
- RPOI$ does not create Offspring Control Blocks (OCBs). Instead of creating an OCB, RPOI$ takes the OCB attached to the DCL task and attaches it to the spawned task. As a result, the spawned task can communicate exit status directly to the parent task. In addition, the DCL task can spawn a task without waiting for the task to exit.

The following subsections describe the second step in interpreting DCL commands (that is, the parser and translator processes) in more detail.

18.5.1 DCL Command Tables

The parser/translator process involved in interpreting DCL commands is table-driven. To parse and translate a DCL command, the DCL task requires the following three tables:

- The master command table (MCT)
- A parser table (PT)
- A translator table (TT)

The tables for each command are contained in separate files. By convention, the file containing the MCT is COMMAND.MAC. The files for the PTs and TTs are named after the DCL command that they describe, followed by the file type MAC. For example, the PT and the TT for the DCL command COPY are in a file named COPY.MAC. However, note that the PT and TT for one command can be located in two separate files. For example, the PT for the DCL command SET is in a file named SET.MAC; whereas, the TT for the same command is located in a file named TSET.MAC.

By altering these tables, you can add, delete, or modify DCL commands. However, before you begin to make any changes, you need to understand how DCL uses these tables. You should also understand the language in which the existing tables are written.

The following subsections describe the DCL command tables in more detail. Section 18.5.2 explains the general characteristics of the macros and rules that define a metalanguage; Section 18.5.3 describes the Macro Metalanguage (MML), which defines the DCL command tables.

18.5.1.1 The Master Command Table

The master command table (MCT) contains an entry for each DCL command. Each entry provides the following information:

- The name of the DCL command
- The address of a parser table (PT) for the command
- The address of a translator table (TT) for the command
- The name of the overlay containing the PT and the TT (optional)

The DCL task uses MCT to recognize an arbitrary command and to locate the PT and TT for that command. MCT consists of calls to COMMAND macros; each DCL command has a corresponding macro. Arguments to the COMMAND macros specify the information required for each MCT entry (the command name, the PT and TT addresses, and optionally, the name of the overlay).

18.5.1.2 The Parser and Translator Tables

There are two tables (one parser and one translator) for each DCL command. Each PT and TT combination contains the information that the parser and translator processes need to translate a DCL command line to an equivalent MCR command line, as follows:

- The syntax of the DCL command
- The syntax of the equivalent MCR command (that is, the syntax of the MCR command to which the DCL command translates)
- The relationship (or *mapping*) between the DCL syntax and the MCR syntax

For example, the PT and TT for the DCL commands ABORT, ASSIGN, and MYOWN are contained in the files ABORT.MAC, ASSIGN.MAC, and MYOWN.MAC, respectively. So, for the ABORT command, the label ABORT:: appears in the parser table contained in the file ABORT.MAC. Likewise, the label ABO:: appears in the translator table in the same file (ABORT.MAC). Table 18–1 summarizes these command tables.

Table 18–1: DCL Command Tables

DCL Command	Master Command Table COMMAND.MAC	Parser Table (PT)	Translator Table (TT)
ABORT	COMMAND ABORT ABORT ABO	ABORT::	ABO::
ASSIGN	COMMAND ASSIGN ASSIGN ASN	ASSIGN::	ASN::
MYOWN	COMMAND MYOWN MYOWN MYN	MYOWN::	MYN::

The PT defines the syntax or structure of the DCL command; thus, it "drives" the parser process. Likewise, the TT defines the syntax or structure of the corresponding MCR command

and "drives" the translator process. The definitions within the tables are expressed in the Macro Metalanguage (MML).

The following section describes a metalanguage in general terms, using an English sentence as the example. Following the introduction of these general concepts is a more detailed explanation of the Macro Metalanguage (MML) in Section 18.5.3.

18.5.2 A General Description of a Metalanguage

MACRO–11 macros and rules define the DCL command tables. The macros and rules form a language, called the Macro Metalanguage (MML), which supports the following characteristics:

- Allows you to specify a prompt string for each syntax element. When a test for end-of-line (EOL) is successful, DCL prompts for more input with the specified string.
- Interacts with the utility. In response to a prompt, users can request help by typing a question mark (?).
- Allows the specification of an error message string for each syntax element. If DCL detects an error while processing a syntax element that has an error message associated with it, DCL issues the error message and aborts the parse.
- Allows the specification of contradictory bit masks for each syntax element. DCL uses the bit masks to detect contradictory syntax.
- Allows the use of globally defined syntax elements, such as file and device specifications.
- Allows MACRO–11 assembly language subroutines to be mixed with MML statements if a command cannot be completely defined with MML.
- Allows one DCL command to translate to more than one MCR command. For instance, the DCL command SET DEFAULT can generate the MCR commands ASN and SET /UIC.
- Supports "floating" qualifiers, which can be specified anywhere in the command line.
- Models a Backus-Naur Form (BNF) metalanguage. A BNF language specifies the sequences of symbols that constitute a syntactically valid statement in a given language. In this case, MML is used to describe the proper syntax of a DCL command and the equivalent syntax of an MCR command.

As an example of metalanguage syntax, consider a sentence in the English language. A syntax tree defines the structure or syntax of a sentence, as follows:

```
THE          OLD             MAN       WALKED       THE       DOG.
<article>    <adjective>     <noun>    <verb>       <article> <noun1>
      \           |          /            \             \       /
       \          |         /              \             \     /
        \         |        /                \        <direct object>
         \        |       /                  \              /
          \       |      /                    \            /
              <subject>                        <predicate>
                        \                      /
                         \                    /
                          \                  /
                           \                /
                               <sentence>
```

Note that the sentence "The old man walked the dog" is one of many sentences that could fit the above structure (syntax). The following description of the syntax tree is a metalanguage definition of the sentence:

<article>	::= THE
<adjective>	::= OLD
<noun>	::= MAN
<verb>	::= WALKED
<noun1>	::= DOG
<direct object>	::= <article> <noun1>
<predicate>	::= <verb> <direct object>
<subject>	::= <article> <adjective> <noun>
<sentence>	::= <subject> <predicate>

The metalanguage definition and the syntax tree describe the same structure; however, the metalanguage uses the following symbols:

- Left angle bracket (<)
- Right angle bracket (>)
- Two colons followed by an equal sign (::=)

The angle brackets denote category references; they enclose the name of a category. *Categories* define classes of objects. For example, the category <subject> defines English words that fit the structure of an <article> , followed by an <adjective> , followed by a <noun> .

The symbol ::= means "is defined as" and denotes a *production rule.* In a production rule, you specify the category name to the left of the symbol and the definition for the category to the right of the symbol. For example, the following production rule for <subject> references the category <adjective> :

```
<subject> ::= <article> <adjective> <noun>
```

Likewise, the production rule for the category <adjective> is as follows:

```
<adjective> ::= OLD
```

Thus, you can rewrite the production rule for <subject> as follows:

```
<subject> ::= <article> OLD <noun>
```

You can also perform additional substitutions for references to the categories <article> and <noun> to produce the following production rule:

```
<subject> ::= THE OLD MAN
```

You cannot substitute for the words THE, OLD, and MAN. These words are *literal strings.* For simplicity, literal strings are called *literals,* and category references are called *nonliterals.* A *category reference* means there is a production rule for the category that requires that the right part of the production rule be substituted for the name of the category.

A *category definition* (the right part of a production rule) consists of literals, nonliterals, or both, concatenated by implied ANDs. The implied AND has no symbol in the metalanguage; instead, it forms category definitions by concatenating arbitrary numbers of syntax elements. A *syntax element* is either a literal or a nonliteral. For example, in the category definition for <subject> , there are three nonliterals, as follows:

- <article>
- <adjective>
- <noun>

An implied AND concatenates these nonliterals.

Two additional notations in the metalanguage are as follows:

Exclusive-OR indicator	|
Optional syntax indicators	[]

Use the exclusive-OR indicator (|) to show that a category can be defined in several ways. For example, the production rule for the category <adjective> is as follows:

```
<adjective> ::= OLD | YOUNG | MIDDLE-AGED
```

Each definition in the production rule is separated by a vertical bar (|). When you substitute for a nonliteral (such as <adjective>) with more than one definition, use only one of the definitions in the substitution. For example, when you substitute for the nonliteral <adjective> , as it is defined by the production rule for <subject> , you produce the following three production rules:

```
<subject> ::= THE OLD MAN

<subject> ::= THE YOUNG MAN

<subject> ::= THE MIDDLE-AGED MAN
```

Use the optional syntax indicators (square brackets) to enclose optional syntax. For example, the following production rule defines the category <subject> :

```
<subject> ::= <article> [ <adjective> ] <noun>
```

The square brackets around the nonliteral <adjective> indicate that <adjective> is optional syntax. This syntax allows you to substitute for <subject> and produce either of the following production rules:

```
<subject> ::= <article> OLD <noun>

<subject> ::= <article> <noun>
```

In addition to denoting optional syntax, the square brackets also indicate iteration. That is, syntax denoted as optional can also be used to define the syntax repeatedly. For example, assume that the following production rule defines the nonliteral <adjective> :

```
<adjective> ::= WISE | OLD | GENTLE
```

You could then rewrite the production rule for <subject> (if <subject> is defined with brackets), as follows:

```
<subject> ::= <article> WISE OLD GENTLE <noun>
```

Note the repeated substitution for the nonliteral <adjective> .

Thus, you can use the symbols, rules, and logic described above as a language (that is, as a *metalanguage*) to define the structure of other languages. The structure or syntax of a language is called the *grammar* of that language. The grammar is used to determine mechanically whether an arbitrary sentence is a sentence in the language.

Our interest here is in the grammar for DCL and MCR, the RSX–11M–PLUS command languages. Sentences in these two grammars are called *commands*. For example, the following metalanguage definition is for the grammar of the DCL command ASSIGN:

```
assign ::= ASSIGN [ / <qual> ] <device> <device>

qual ::= LOCAL | LOGIN | GLOBAL | TERMINAL: <device>
```

Likewise, the following metalanguage definition is for the grammar of the MCR command ASN:

```
asn ::= ASN <device>=<device> [ / <qual> ]

qual ::= LOGIN | GBL | TERM= <device>
```

The following section describes the correspondence of the metalanguage notation presented in this section to Macro Metalanguage (MML) notation.

18.5.3 The Macro Metalanguage

The Macro Metalanguage (MML) is a set of MACRO–11 macros and rules that define the syntax for the DCL and MCR command languages. It implements the logic that is characteristic of a BNF-type metalanguage (as described in the previous section). By providing a means of generating (with the MACRO–11 assembler) the data structures that drive the parser/translator process, MML expresses the mapping from DCL syntax to equivalent MCR syntax.

The following subsections summarize the basic components of MML.

Syntax Elements (Literals and Nonlinterals)

In MML, there are two types of syntax elements: literals and nonliterals. Both are presented as arguments to the MACRO–11 assembly language AND and OR macros. An argument to either macro represents one syntax element (a literal or a nonliteral).

To distinguish a literal from a nonliteral, place a single quotation mark (') before the first character (for example, 'ASSIGN). In addition, to indicate the minimum number of characters that the parser must match for a parse to be considered successful, specify an *abbreviation count*. The abbreviation count is an optional argument that you can specify with a literal. For example, to indicate that the literal ASSIGN can be abbreviated to ASSI, rewrite 'ASSIGN as <'ASSIGN,4> .

Note

> The use of the left and right angle brackets (< and >) as symbols indicates to MACRO–11 that the argument ('ASSIGN,4) is one argument instead of two (that is, ASSIGN and 4). Do not confuse these symbols with the identical metalanguage symbols used to identify nonliterals.

Logical Operators (AND and OR Macros)

The Macro Metalanguage has two MACRO–11 macros to correspond to the implied AND and exclusive OR operators: AND macros and OR macros.

To define a *category*, place a MACRO–11 label on either the AND or the OR macro. The *label* is the name of the category, and the *arguments* to the macro are the definition for the category.

For example, the metalanguage notation for the category <qual> is as follows:

```
<qual> ::= GLOBAL | LOCAL | TERMINAL: <device>
```

The equivalent MML notation is as follows:

```
QUAL:  OR 'GLOBAL 'LOCAL TERM

TERM:  AND 'TERMINAL: $DEV
```

This example uses the following notations:

- Two categories: QUAL and TERM
- Two MACRO–11 macros: AND and OR
- Three category definitions for QUAL: 'GLOBAL, 'LOCAL, or TERM

  The category definitions for QUAL consist of two literals ('GLOBAL and 'LOCAL) and one nonliteral (TERM).
- One category definition for TERM: 'TERMINAL:$DEV

  The category definition for TERM consists of a literal ('TERMINAL) and a predefined nonliteral ($DEV). (See Section 18.10.2 for a description of common predefined syntax elements.) Note that 'TERMINAL: $DEV is specified on a separate line with the AND macro.

Optional Syntax Elements (OPT=T)

Use the MML notation OPT=T (optional=true) to indicate that syntax elements are optional. (This notation corresponds to the metalanguage notation [and]). The optional syntax notation OPT=T is an attribute of syntax elements (that is, literals and nonliterals). Literals and nonliterals in MML notation are arguments to AND and OR macros. Therefore, you can specify the optional syntax indicator OPT=T as part of the arguments.

To specify OPT=T as an attribute of a syntax element, enclose the syntax element and the OPT=T notation in angle brackets. For example, to specify that the terminal device specification $DEV in the previous example is an optional syntax element (nonliteral), rewrite the category TERM as follows:

```
TERM:  AND  'TERMINAL: <$DEV OPT=T>
```

Use the OPTR=T attribute to indicate that a syntax element is optional and to indicate that the parser/translator can process the syntax element repeatedly.

The notation OPT means a syntax element can occur, at most, one time. The notation OPTR means a syntax element can occur many times. In either case (OPT or OPTR), the element can also be specified to occur zero times (in other words, the element is optional). For a complete description, see Section 18.9.11.

Macros (OR., AND., and END)

To specify an arbitrary number of arguments for the AND and OR macros, use the OR., AND., and END macros. Use the AND. and OR. macros as continuation markers for the AND and OR macros, respectively. That is, if there is not enough space on the line for all the arguments to either the AND or OR macros, place additional arguments on the next line. To do this, place the AND. or the OR. macro on the next line, followed by the continuation of the specification of the arguments. For example, you could rewrite the category for <QUAL> to the following:

```
QUAL:  OR 'GLOBAL 'LOCAL
       OR. TERM
       END
```

Use the END macro to indicate the end of an argument list, even if the list does not continue on more than one line. For example, specify the END macro in the following category definition for <QUAL> :

```
QUAL:  OR 'GLOBAL 'LOCAL TERM
       END
```

18.5.4 Interpreting DCL Command Tables

The parser/translator process locates the PT and the TT for a particular DCL command line by searching the master command table (MCT) for the command entry. The parser/translator extracts from the command entry the address of the first AND or OR macro of the PT and the address of the first AND or OR macro of the TT.

The following sections describe how the parser and the translator interpret the PT and the TT.

18.5.4.1 Parser Processing

The parser starts processing at the address of the first macro in the PT. If the processing of this macro succeeds, then the parser process succeeds.

The parse is successful if the following two conditions are met:

- The input command line is a syntactically correct DCL command line (as defined by the PT).
- The intermediate form (IFORM) has been constructed.

The IFORM is a condensed representation of the input command line. If the processing of this macro fails, then the parser process fails.

The parser processes each AND macro argument from left-to-right. The process continues until all the arguments have been processed successfully or until one argument fails to process. If all arguments process successfully, then the processing of the macro succeeds; otherwise, the processing is aborted.

The parser also processes each OR macro from left-to-right. However, the criteria for determining whether or not the processing is successful differs from the criteria for the AND macro. If one of the arguments is processed successfully, then the processing of the macro succeeds. If one argument fails to process, the processing is aborted. The processing does not continue until all the arguments have been processed; its success or failure depends upon one argument.

18.5.4.2 Translator Processing

The translator starts processing at the address of the first macro of the TT. If the processing of this macro succeeds, then the translation succeeds and an equivalent MCR command line (as defined by the TT) has been constructed from the IFORM. If the processing of this macro fails, then the translator process fails. The processing of the AND and OR macros by the translator is the same as the processing of the AND and OR macros by the parser (as described in the previous section).

Arguments to AND and OR macros can be literals or nonliterals. Nonliterals are handled in the same way by both the parser and the translator. Literals, however, are handled differently by the parser and the translator (see Section 18.5.4.3).

18.5.4.3 Parser/Translator Handling of Literals

A string of American Standard Code for Information Interchange (ASCII) characters used as an argument to an AND or OR macro is called a *literal*. The string can be a keyword in the DCL or MCR command grammar. The parser and translator process literals differently.

To parse a literal, the parser attempts to match the literal to the input DCL command line. If the input command line contains the same ASCII string as the literal, then the match is successful and the processing of the literal succeeds; otherwise, nothing happens.

However, to translate a literal, the translator copies the literal from the TT to the MCR command line buffer. If you do not associate attributes with a literal in a TT, then the processing of the literal always succeeds. (In other words, writing a literal to the MCR command line buffer always succeeds.) A literal in a TT can fail only if you have associated one of the following attributes with the literal:

- MAP
- CBM
- BCT
- BST

When you associate these attributes with a literal, the translator processes the attributes before the literal is written to the MCR command line buffer.

The MAP attribute instructs the translator to check for a mapping number in the IFORM. If the number is not there, the translator does not write the literal to the MCR command line buffer and the processing of the literal fails.

The CBM, BCT, and BST attributes instruct the translator to check the status of certain bits in its internal data structures. If the status of the bits is not correct, the translator does not write the literal to the MCR command line buffer and the processing of the literal fails. (See Section 18.9 for more information on these attributes.)

18.5.4.4 Parser/Translator Handling of Nonliterals

In MML, when you use the label of an AND or OR macro as the argument to an AND or OR macro it is called a nonliteral; thus, a nonliteral is the address of another AND or OR macro. The processing of a nonliteral is the processing of the other AND or OR macro that the nonliteral addresses. That is, when the parser or translator encounters a nonliteral (as an argument to an AND or OR macro), the parser or translator transfers parsing or translation control to the macro that the nonliteral addresses.

The addressed macro is either an AND or OR macro and the parser/translator processes this macro like all AND or OR macros (as described in the previous section). If the processing of this macro succeeds, then the processing of the nonliteral succeeds; otherwise, the processing of the nonliteral fails.

18.5.4.5 Summary of Parser/Translator Functions

The following list summarizes the Parser and Translator functions:

- The parser table (PT) drives the parser process and the translator table (TT) drives the translator process.
- The parser/translator processes succeed only if the processing of the first AND or OR macro of the PT/TT succeeds.
  - The processing of an AND or OR macro succeeds only when the processing of the arguments to the macros succeeds.
- Arguments can be either literals or nonliterals.
  - A nonliteral instructs the parser/translator to process (recursively) another AND or OR macro. It succeeds only if the subsequent AND or OR macro succeeds.
  - A literal instructs the parser to ensure the input command line matches the literal. It succeeds only if the match is successful.
- A literal instructs the translator to output the literal (possibly conditionally) to the MCR command line. The translator succeeds only if either the literal is not conditionalized or the conditionals are satisfied.

18.5.4.6 Example of P/T Tables for DCL ASSIGN Command

The following is an example of a PT for the DCL command ASSIGN:

```
          MML notation for Parser Table

ASSIGN::  AND       'ASSIGN <QUAL OPTR=T>$DEV $DEV
          END
QUAL:     AND       '/ QUALI
          END
QUALI:    OR        <'LOCAL,4>
          OR.       <'LOGIN,4>
          OR.       <'GLOBAL,2>
          OR.       TERM
          END
TERM:     AND       'TERMINAL: <'TERMINAL,4>':
          END
```

The following is an example of a translator table (TT) for the MCR command ASN:

```
        MML notation for Translator Table
ASN::   AND     'ASN $DEV '= $DEV <LAUQ OPTR=T>
        END
LAUQ:   OR      '/LOGIN
        OR.     '/GBL
        OR.     MRET
        END
MRET:   AND     '/TERM= $DEV
        END
```

These examples show the first two steps in creating the tables for the DCL command ASSIGN. The last step is to express the mapping of the syntax elements in the PT to the syntax elements in the TT. This mapping provides the information that the parser/translator process needs to construct the MCR command line for ASN from the DCL command line for ASSIGN. The following section describes how to express this mapping.

18.5.5 Mapping DCL to MCR

For the translator to know which particular MCR command line to construct from a particular DCL command line, the PT and the TT must express a relation between the DCL syntax and the MCR syntax. The PT/TT expresses this relation by mapping elements from DCL syntax to elements in MCR syntax.

The parser passes the mapping information to the translator with a dynamic data structure called the Intermediate Form (IFORM). The parser processes mapping information and records it in the IFORM. The translator searches the IFORM for the information and then generates the MCR command line.

Mapping information is expressed in MML with the MAP=n attribute. MAP takes one or two values: the first value is the *primary mapping number*; the second value is the *secondary mapping number.*

Use one of the following formats to specify the MAP attribute:

- If you need only the primary mapping number, specify it in the following form:

  ```
  MAP=n
  ```

- If you need to specify both mapping numbers (which is allowed only in a TT), or if you need to specify only the second number, enclose the numbers in angle brackets and separate them with a comma, as follows:

  ```
  MAP=<n,n> or MAP=<,n>
  ```

- To associate the MAP attribute with a syntax element, enclose the syntax element and the MAP attribute in angle brackets, as follows:

  ```
  <syntax-element MAP=n>
  ```

- To map a DCL syntax element in a PT to an MCR syntax element in the corresponding TT, use MAP with the same mapping number for both syntax elements.

For example, in the ASSIGN command, the DCL qualifier /GLOBAL maps to the MCR keyword /GBL. That is, when you type the DCL qualifier /GLOBAL, it is translated to the the MCR keyword /GBL. To represent this mapping in the DCL command tables, use MAP with the same mapping number for both /GLOBAL and /GBL, as follows:

```
QUAL:  OR <'/GLOBAL MAP=1> ...
       END
LAUQ:  OR <'/GBL MAP=1> ...
       END
```

MML macros at QUAL: define DCL syntax. MML macros at LAUQ: (QUAL spelled backwards) define the corresponding MCR syntax.

In the previous example, when the parser reaches QUAL:, it tries to match the literal /GLOBAL to the input DCL command line. If the match is successful, then the parser queues the MAP number P1 (the P indicates a primary mapping number) to the IFORM queue.

When the translator reaches LAUQ:, it searches the IFORM queue for the MAP number P1. If it finds P1, it sends the literal /GBL to the IFORM for the MCR command line. However, if it cannot find the MAP number in the IFORM queue, it does not use the corresponding syntax element in the translation.

18.5.5.1 Using the MAP Attribute in the Parser Table

When you use the MAP attribute to associate a primary mapping number with a syntax element in the PT, you instruct the parser to queue the primary mapping number to the IFORM queue. Likewise, when you associate a secondary mapping number with a syntax element in the PT, you instruct the parser to queue the secondary mapping number to the IFORM queue. In both cases, the parser queues a mapping number only if the parser processes the associated syntax element successfully.

The MAP attribute in a PT can associate either a primary or a secondary mapping number with a syntax element, but not both.

18.5.5.2 Using the MAP Attribute in the Translator Table

When you use the MAP attribute in a TT, you instruct the translator to search the IFORM queue for a mapping number. When the translator finds the number, it marks the number as "used" and processes the associated syntax element.

The MAP attribute in a TT can associate a primary mapping number, a secondary mapping number, or both, with a syntax element. However, the search for a primary mapping number differs from the search for a secondary mapping number, as follows:

- To find a primary mapping number, the translator searches the entire IFORM queue from beginning to end.
- To find a secondary mapping number, the translator begins searching at the queue element containing the last used primary mapping number, and it stops searching at the next queue element containing a primary mapping number.

When you specify both a primary and a secondary mapping number in the TT (for example, MAP= <1,1>), you instruct the translator to search the entire IFORM queue for the specified primary mapping number such that the secondary mapping number follows without any intervening primary mapping numbers. The secondary mapping number qualifies the primary mapping number, but the primary mapping number is the object of the search.

If the translator finds a mapping number in the IFORM queue, the translator processes the associated syntax element. If the translator fails to find the mapping number, the processing of the associated syntax element fails.

When the translator finds a mapping number and successfully processes the associated syntax element, the translator marks as used the queue element containing the found mapping number. A *used mapping number* is a mapping number that is ignored by the translator. Marking a mapping number as used prevents infinite loops in the TT without requiring you to specify termination conditions.

18.5.5.3 Using the MAP Attribute Without a Syntax Element

You can specify the MAP attribute in both the PT and the TT without associating it with a syntax element, as follows:

```
<MAP=n>
```

If you specify this attribute as an argument to an AND or OR macro in a PT, the parser assumes a null syntax element, which is always parsed successfully.

If you specify a null syntax element in a TT, the translator (after finding the queue element containing the mapping number) checks to see if a character buffer is attached to the queue element. If a character buffer is attached, then the translator copies the contents of the buffer to the MCR command line buffer. If a character buffer is not attached, then the translator assumes a null syntax element, which is always translated successfully.

The parser attaches the character buffer to the queue element. The COP attribute instructs the parser to create the character buffer and to copy the text parsed by the associated syntax element to the character buffer (see Section 18.9.5 for details on the COP attribute).

18.5.5.4 Summary of the Mapping Procedure

The following list summarizes the mapping procedure:

- The parser records mapping information; the translator detects the mapping information.
- Mapping starts in the parser process. The parser queues mapping numbers to the IFORM queue, as it successfully processes syntax elements that have associated MAP attributes.
- Mapping is completed in the translator process. The translator detects mapping numbers in the IFORM queue and translates the syntax elements.
- The translator does not output a syntax element that has a MAP attribute associated with it unless the following two conditions are met:
  - The translator finds the mapping number in the IFORM queue.
  - The mapping number is not marked as used.

Example

The following is an example of a complete DCL command table (with DCL-to-MCR mapping) for the DCL command ASSIGN:

```
          ASSIGN MML notation for Parser Table

ASSIGN::  AND     'ASSIGN <QUAL OPTR=T><$DEV MAP=1><$DEV MAP=2>
          END
QUAL:     AND     '/ QUALI
          END
QUALI:    OR      <'LOCAL,3>
          OR.     <'LOGIN,3 MAP=3>
          OR.     <'GLOBAL,2 MAP=4>
          OR.     TERM
          END
TERM:     AND     <'TERMINAL,4>' <DEV MAP=5>
          END

          ASN MML notation for Translator Table

ASN::     AND     'ASN <MAP=1>'= <MAP=2><LAUQ OPTR=T>
          END
LAUQ:     OR      <'/LOGIN MAP=3>
          OR.     <'/GBL MAP=4>
          OR.     MRET
          END
MRET:     AND     '/TERM= <MAP=5>
          END
```

For a complete description of the MAP attribute, see Section 18.9.9. For information on related attributes, see the descriptions of the following attributes:

- DMAP
- NO
- COP
- OCOP
- TST
- DIR

The following section describes the IFORM, the dynamic data structure that passes mapping information from the parser to the translator.

18.5.6 The Intermediate Form

The Intermediate Form (IFORM) contains the essential information (extracted from the input DCL command line) that the translator needs to construct the MCR command line. The IFORM consists of three data structures: the queue, the character buffers, and the general bit-mask words.

Note

> To display the IFORM for a given DCL command, enter the DCL command SET DEBUG/FULL.

- When the parser successfully processes syntax elements with MAP attributes, it creates the *IFORM queue*. Then, each time the parser successfully processes a subsequent MAP attribute, it queues the mapping number to the IFORM queue.
- When a syntax element has both MAP and COP attributes, the parser creates a *character buffer*. The parser attaches the character buffer to the queue entry that contains the mapping number. The character buffer contains the text parsed during the processing of the associated syntax element.

  A syntax element with a COP attribute and no MAP attribute instructs the parser to copy the text parsed to the last queued mapping number (creating and attaching the character buffer, if necessary).
- The *general bit-mask words* are two words used as a set of 32 flags. The parser/translator process attaches no meaning to the flags, but MML provides a set of *attributes* to perform the following functions:
  - Set flags
  - Clear flags
  - Test to see whether or not the flags are set
  - Test to see whether or not the flags are clear

  You can use the flags to check for contradictory syntax or to pass information from the parser to the translator. For example, set a flag during the parser process if a DCL qualifier is parsed. Then, detect the set flag during the translator process to generate the equivalent MCR keyword.

The translator searches the IFORM queue for mapping numbers. If the translator finds the mapping numbers, it processes the syntax elements. If the translator cannot find the mapping numbers, then the syntax elements fail.

For example, assume you have specified a MAP attribute without an associated syntax element, as follows:

```
<MAP=n>
```

The translator first finds the queue element containing the mapping number. Then, it examines the queue element for a character buffer attached with the COP attribute. If the translator finds an attached character buffer, it copies the contents of the buffer to the MCR command line buffer. If it cannot, it assumes a null syntax element (which always succeeds).

Creating the IFORM

Use the following attributes in the PT to direct the parser in the creation of the IFORM:

Attributes	Meaning

MAP=n
MAP= <,n>
	Map

DMAP=n
DMAP= <,n>
	Map, unless duplicate exists

COP=T
OCOP=T
	Copy, or copy optional input

BS=n
BS= <,n>
BS= <n,n>
	Set bit in mask word

BC=n
BC= <,n>
BC= <n,n>
	Clear bit in mask word

NO=T	Repeat for negative form of syntax element

Use the following attributes (in the TT) to direct the translator in the examination of the IFORM:

Attributes	Meaning

MAP=n
MAP= <,n>
MAP= <n,n>
	Unmap

BST=n
BST= <,n>
BST= <n,n>
	Test for bit set in mask word

BCT=n
BCT= <,n>
BCT= <n,n>
	Test for bit clear in mask word

TST=T	Test for mapping number, do not mark as used
DIR=T	Search IFORM in reverse direction

For more information on these attributes, see Section 18.9.

Examples

$

Translation of this DCL command line produces the following IFORM:

Queue	Character Buffer
P1	DK1:
P2	SY:

$

Translation of this DCL command line produces the following IFORM:

Queue	Character Buffer
P3	
P5	TT12:
P1	DK1:
P2	SY:

18.6 Adding New DCL Commands

A DCL command consists of an entry in the master command table (MCT), a parser table (PT), and a translator table (TT). The PT defines the syntax of the DCL command; the TT defines the syntax of the corresponding MCR command. The entry in the MCT connects the PT and the TT to the DCL task. Thus, to add a new command to the DCL task, you must construct a PT and a TT and make an entry in the MCT.

Adding a command to the DCL task involves the following steps:

1. Creating the PT and the TT (that is, creating the file that contains the PT and the TT)
2. Creating the MCT entry
3. Creating an entry in the overlay description file for the DCL task
4. Incorporating the new PT/TT file and the modified MCT file into the DCL task

18.6.1 Creating the PT/TT File

The file that contains the PT and the TT is named after the verb of the command. The file type is MAC (for example, the PT and the TT for the COPY command are in a file named COPY.MAC). The PT/TT file must specify the MACRO-11 assembly directives .TITLE, .PSECT, and .END. The title of the file (specified as an argument to the .TITLE directive) must also be the same as the verb of the command. Normally, the program-section name is CDEF, which stands for "command definition." However, if MACRO-11 subroutines are interspersed with Macro Metalanguage (MML) notation (by using the $JSR predefined syntax element), the subroutines must be in the blank program section. (See Section 18.10.5 for details on $JSR.)

In addition to these directives, the PT/TT file must contain a MACRO-11 global label on the first MML macro of the PT and a global label on the first MML macro of the TT. By convention, the label on the PT is the same as the verb of the DCL command, and the label on the TT is the same as the verb on the corresponding MCR command. The following example shows the structure of the file that contains the PT and the TT for the DCL ASSIGN command:

```
         ASSIGN.MAC

         .TITLE  ASSIGN
         .PSECT  .CDEF
ASSIGN::

         Parser Table

ASN::

         Translator Table

         .END
```

The two global labels in the PT/TT file connect the PT and the TT to the DCL task. This is accomplished by making an entry in the MCT.

18.6.2 Creating the MCT Entry

The file COMMAND.MAC contains the master command table (MCT). The MCT consists of a list of MACRO–11 COMMAND macro calls. Each COMMAND macro call identifies a DCL command. The list of COMMAND macro calls is ordered alphabetically by the name or verb of the command. The COMMAND macro takes five arguments; the first four are mandatory and the fifth is optional.

Format

```
COMMAND   name, min, ptaddr, ttaddr, [ovrnam]
```

Arguments

name

Specifies the name (or verb) of the command.

min

Specifies the minimum number of characters necessary to make the name unique.

ptaddr

Specifies the label of the PT for the command.

ttaddr

Specifies the label of the TT for the command.

[ovrnam]

Specifies the 3-character name of the overlay for the PT and TT.

This argument to the COMMAND macro is optional. However, even if you do not specify this argument, the PT and TT must be in an overlay.

If PT and TT are in the same module, the default overlay name is the command name. If they are in separate modules, the PT overlay name is the command name (for example, SET.MAC) and the TT overlay name is the command name preceded by a T (for example, TSET.MAC).

The MCT is in two parts. The first part of the MCT includes all DCL command names (verbs) in alphabetical order, as described in this section. It is searched with a binary search algorithm. The second part describes the brief forms of commonly used DCL commands. If the search of Part 1 for a command verb fails (for example, if a line is too short), a search of Part 2 is initiated. Note that Part 2 is not necessarily in alphabetical order; it is searched from the top down.

18.6.3 Specifying the Overlay Entry

Each PT/TT must reside in an overlay. Use one of the following formats to overlay a PT/TT:

- One overlay for two or more PT/TTs
- One overlay for one PT/TT
- Two overlays for one PT/TT
- More than two overlays for one PT/TT

Specify an overlay by making an entry in the overlay description files (DCLBLD.ODL and DCDBLD.ODL) and connect the new overlay entry to the root segment overlay entry.

Format 1: One Overlay for Two or More PT/TTs

```
        .NAME   ovrnam
label:  .FCTR   ovrnam-DCLO/LB:cmdnam1:cmdnam2...:cmdnamn-L
```

Arguments

label

Connects the entry to the root segment.

ovrnam

Specifies the 3-character name of the overlay. Include this argument for each entry in the MCT whose command tables are in an overlay with two or more PT/TTs.

DCLO/LB

Specifies the object library containing the modules in the overlays.

:cmdnam1:cmdnam2: . . . :cmdnamn-L

Specifies the title of the file containing the first (cmdnaml), second (cmdnam2), and subsequent PT and TT.

Format 2: One Overlay for One PT/TT

```
label:  .FCTR DCLO/LB:cmdnam-L
```

Arguments

label

Connects the entry to the root segment.

DCLO/LB

Specifies the object library containing the modules in the overlays.

cmdnam-L

Specifies the title of the file that contains the PT/TT.

The title of the file containing the PT/TT must be the same as the name or verb of the command because the Task Builder (TKB) uses the title as the default for the overlay name.

This format does not specify the overlay name. Instead, the DCL task uses the name or verb of the command as the overlay name.

Format 3: Two Overlays for One PT/TT

```
label:  .FCTR (DCLO/LB:cmdnam-L,DCLO/LB:tcmdnam-L)
```

Arguments

label

Connects the entry to the root segment.

DCLO/LB

Specifies the object library containing the modules in the overlays.

cmdnam-L

Specifies the title of the file that contains the PT. By convention, the title is the same as the verb of the command. (For example, the file containing the PT for the DCL command SET is named SET.MAC with the title SET.)

tcmdnam-L

Specifies the title of the file that contains the TT. By convention, the title is the same as the verb of the command prefixed by the letter T. (For example, the file containing the TT for the DCL command SET is named TSET.MAC with the title TSET.)

This format requires separate files for the PT and TT. The separate files have the same format as a single file that contains the PT/TT; however, each file contains only one global label.

Format 4: More than Two Overlays for One PT/TT

```
label:  .FCTR
        (DCLO/LB:cmdnam-L,DCLO/LB:tcmdnam-L,DCLO/LB:ovrnam1-L)
```

Arguments

label

Connects the entry to the root segment.

DCLO/LB

Specifies the object library containing the modules in the overlays.

cmdnam-L

Specifies the title of the file that contains the PT.

tcmdnam-L

Specifies the title of the file that contains the TT.

ovrnam1-L

Specifies the title of a file containing additional portions of the PT or TT.

This format allows a PT or a TT to invoke additional overlays containing portions of the PT or TT. Since the PT and TT are read-only data, the additional overlay segments are loaded on top of the overlay that is invoking them. When the DCL task returns from the overlay that was invoked, it reloads the previous overlay. Control returns to the point in the previous overlay from which the additional overlay was invoked.

Use the predefined syntax element $CALLOV to invoke additional overlays from a PT or a TT. (See Section 18.10.5, for details on the $CALLOV predefined syntax element.) To specify additional overlays, repeat the previous format. That is, for each additional overlay, specify an additional argument to the MACRO-11 directive .FCTR by using the following format:

```
DCLO/LB:ovrnam-L
```

Next, connect the new overlay entry to the root segment of the DCL task by adding the label of the entry to the list of labels for other entries. The list of labels for other entries is found in the overlay description file after the label OLAY. By convention, enter the labels in alphabetical order. The following format shows the list of overlay entry labels:

```
OLAY:     .FCTR (l1,l2,l3,l4,l5,...,ln)
```

The variables l1, l2, and so on, represent the labels for the overlay entries.

Because there are two overlay description files for building the DCL task (DCLBLD.ODL and DCDBLD.ODL), you must make new overlay entries in both files. The format is the same for both overlay description files.

For details on how to build the DCL task, see Section 18.4. For details on overlay structures, see the *RSX-11M-PLUS and Micro/RSX Task Builder Manual.*

The last step required to add a command to the DCL task is the incorporation of all created or modified files into the DCL task. The procedure for incorporating the PT, TT, and MCT is described in the following section.

18.6.4 Incorporating the PT/TT and MCT

The last step in adding new commands to DCL (assembly and task building) is accomplished through the indirect command file [23,24]DCL.CMD, which is located on the same device as the DCL source files. This file prompts for the name of a file containing the names of all modules you have changed or created. The input file consists of one module name per line; each name is entered without a file type or version number. The command file DCL.CMD assembles the modules, updates the object libraries, and task builds the new version of DCL.

Before you run DCL.CMD, establish the appropriate environment, as follows:

- Set your command line interpreter (CLI) to MCR.
- Install the following tasks (if they are not already installed):
  - The MACRO-11 Relocatable Assembler (MAC)
  - The Librarian Utility Program (LBR)

  - The Peripheral Interchange Program (PIP)
  - The Task Builder (TKB)
- Verify that the following files are present on your system and that you have access to them:
  - Your DCL source files
  - DCLO.OLB and DCLR.OLB (overlay and root library files for DCL)
  - DCLMAC.MAC and COMMAC.MAC (prefix files for DCL)
  - DCLBLD.CMD (task-build command file)
  - DCLBLD.ODL and DCDBLD.ODL (overlay descriptor files)

  By default, DCL.CMD searches for the files in the User File Directory (UFD) with which it is invoked.
- Verify that the file RSXMC.MAC is in directory [11,10] on device LB:.

Then, invoke DCL.CMD with the following command line:

```
>
*Do you want expanded comments; [Y/N]:
```

Answer Y (Yes) to this question. The comments in DCL.CMD direct the remainder of the process.

If all source files assemble correctly, DCL.CMD builds the DCL task. (See Section 18.4 for information on how to determine if the DCL task was built correctly and how to install DCL as a CLI.)

18.7 Deleting Commands

To delete a command from DCL, break the link between the MCT and the PT/TT by deleting the appropriate COMMAND macro entry in the MCT, which is in the file COMMAND.MAC. Then, incorporate the modified COMMAND.MAC file into the DCL task by following the procedure outlined in the preceding section. You can also delete the entry in the overlay description files and the object modules for the PT/TT in the DCLO.OLB library, but this step is optional.

18.8 Modifying Commands

Modifying an existing DCL command involves modifying the PT/TT files to reflect the desired change and incorporating the modified files into the DCL task, as described in Section 18.4.

18.9 Attributes of Syntax Elements

Attributes associate characteristics with literal and nonliteral syntax elements. The parser /translator processes attributes either before or after it processes the associated syntax elements (depending on the attribute). The processing of attributes can influence the processing of the associated syntax elements.

Formats

```
<'literal   KEY=value KEY1=value ...>

<label       KEY=value KEY1=value ...>
```

Parameters

literal

Specifies the literal string in American Code for Information Interchange (ASCII).

label

Specifies the nonliteral (that is, the address of an AND or OR macro).

KEY

Specifies the attribute keyword identifying the attribute.

value

Specifies the value assigned to the attribute.

Note the use of the angle brackets (< and >). Use these MACRO–11 symbols to indicate that the enclosed parameters are to be considered as one argument (when expanding macros). Do not confuse these MACRO–11 angle brackets with the angle brackets used to indicate nonliterals in metalanguage notation.

18.9.1 AND

The AND attribute allows you to extend a syntax element by specifying a nonliteral that is a logical AND to the syntax element. The primary function of AND is to allow OR logic to be mixed with AND logic. Assign a nonliteral to the AND attribute as follows:

```
AND=label
```

The *label* is the nonliteral address of an AND or OR macro.

AND instructs the parser/translator to process the AND or OR macro that the label (the nonliteral) addresses. In addition, the parser/translator processes the syntax element with which the AND attribute is associated.

Example

In the following MML definition for QUAL, the second syntax element defines the structure FILE: <filespec> :

```
QUAL:  OR     'BRIEF
       OR.    <'FILE AND=FNAME>
       OR.    'FULL
       END
FNAME: AND    ': $FILE
       END
```

There are three syntax elements in this structure:

FILE — Specifies a literal to which the AND attribute connects the other syntax elements.

colon (:) — Specifies a literal, which connects to the literal FILE with the AND attribute.

<filespec> — Specifies a nonliteral, which also connects to the literal FILE with the AND attribute.

By definition of the AND attribute, the parser/translator processes all three syntax elements as one argument to the OR macro.

18.9.2 BC and BS

Bit Clear (BC) and Bit Set (BS) attributes operate on two global bit-mask words. The BC attribute instructs the parser/translator to clear the bits in these words; the BS attribute instructs the parser/translator to set the bits.

Use the BCT and BST attributes to test these bits (see the next section).

Formats

```
BC=n     or     BC=<,n1>      or     BC=<n,n1>

BS=n     or     BS=<,n1>      or     BS=<n,n1>
```

Arguments

n

Selects a bit mask that specifies bits in the first global word.

n1

Selects a bit mask that specifies bits in the second global word.

Specify a bit mask by using logical OR, single-bit specifications with the exclamation point (!). The symbols B1, B2, . . . B32 are predefined to correspond to single-bit specifications for each bit in the two global words.

For example, the expression BS=B1!B2!B5 sets bits 1, 2, and 5 of the first global word. See Section 18.9.3 for an example of how to use these attributes.

18.9.3 BCT and BST

Bit Clear Test (BCT) and Bit Set Test (BST) attributes test the status of bits in two global bit-mask words. The BCT attribute instructs the parser/translator to test whether bits are cleared. The BST attribute instructs the parser/translator to test whether bits are set. (See Section 18.9.2 for information on how these bits are set or cleared.)

The parser/translator does not clear the two global bit-mask words between the parse and the translation. You can use these words as part of the IFORM to pass information from a PT to a TT.

Formats

```
BCT=n     or     BCT=<,n1>     or     BCT=<n,n1>

BST=n     or     BST=<,n1>     or     BST=<n,n1>
```

Arguments

n

Selects a bit mask that specifies bits in the first global word.

n1

Selects a bit mask that specifies bits in the second global word.

For example, the expression BST= <,B19!B31> tests whether bits 19 and 31 of the second global bit-mask word are set.

Example

The following example illustrates the use of the BS, BC, BST, and BCT attributes:

```
COPY:   AND     FILE <FILE OPTR=T>QUA=QUAL
        END

FILE:   OR      <$NFIL          BS=B1   BCT=B2!B3>
        OR.     $FILE
        END

QUAL:   OR      <'INDEX         BS=B2   BCT=B1!B3>
        OR.     <'SEQUENTIAL    BS=B3   BCT=B1!B2>
        END
```

In this example, INDEX and SEQUENTIAL are qualifiers. $NFIL and $FILE are predefined syntax elements: $NFIL represents a file specification that has a node name prefixing it; $FILE represents a file specification without a node name.

In the example, the following combinations are contradictory:

- INDEX and SEQUENTIAL
- INDEX and $NFIL
- SEQUENTIAL and $NFIL

However, duplicates of each qualifier or attribute are not contradictory.

BCT/BST

The BS attribute instructs the parser to set the specified bits if the processing of the associated syntax elements is successful. The BCT instructs the parser to test the specified bits and to determine whether or not they are clear.

If the tested bits are set, the parser does not process the associated syntax elements. Thus, bits are set to remember which syntax elements process successfully. In addition, bits are tested to determine if a contradiction would result from the processing of syntax elements.

The main difference between the BC, BCT, BS, and BST attributes and the CBM attribute (described in Section 18.9.4) is as follows: CBM always prevents duplicate or contradictory syntax; however, it cannot pass information from the parser to the translator.

18.9.4 CBM and CCBM

The Contradictory Bit Mask (CBM) attribute prevents the parser or translator from allowing contradictory syntax. To indicate that two or more syntax elements cannot be specified in the same command line, specify CBM with each syntax element. First, assign a bit mask as the value to CBM. Then, specify the same bit in each bit mask associated with each contradictory syntax element.

CBM conditionalizes the processing of the associated syntax element. The parser uses the specified bit mask to set bits in two global words. If the parser detects that any bits specified in the bit mask are already set in the two global words, then the parser assumes a contradiction. The syntax element fails; consequently, CBM does not allow duplicate syntax elements.

Also, note that the parser/translator clears two global words between the parse and the translation. Therefore, you cannot use CBM to pass information from a PT to a TT.

Format

```
CBM=n or CBM=<,n1>or CBM=<n,n1>
```

Arguments

n

Selects a bit mask that specifies bits in the first global word.

n1

Selects a bit mask that specifies bits in the second global word.

To use the CBM attribute, specify a bit mask with an exclusive OR operation on single-bit specifications, as follows:

- Use the symbols B1, B2, . . . , B32, which are predefined to correspond to a single-bit specification.
- Use the exclamation mark (!), which is the MACRO–11 symbol for an exclusive OR operation.

For example, CBM=B1!B2!B5 specifies bits 1, 2, and 5 of the first global word, and CBM= <,B19!B31> specifies bits 19 and 31 of the second global word.

Example

The following example illustrates the use of the CBM attribute:

```
QUAL:   OR      <'LOCAL     CBM=B1!B2>
        OR.     <'LOGIN     CBM=B2>
        OR.     <'GLOBAL    CBM=B1>
        END
```

The example specifies that the following combinations of literals are contradictory:

- LOCAL and GLOBAL
- LOCAL and LOGIN

Note that the literals LOGIN and GLOBAL are not contradictory.

The example also specifies that each of these elements can be used only once.

CBM/CCBM

The Clear Contradictory Bit Mask (CCBM) attribute clears the bits in the two global words. The format for specifying CCBM is identical to the format that you use to specify the CBM attribute.

18.9.5 COP and OCOP

The Copy (COP) attribute is valid only in a PT and only with a literal syntax element. COP passes a literal from the input DCL command line to the IFORM. The translator can then use the literal in the MCR command line. The parser copies the literal to a character buffer and attaches it to the mapping number last queued. If you specify a MAP attribute with the COP attribute, the mapping number (assigned to the MAP attribute) is the mapping number last queued.

Format

```
COP=T
```

To instruct the translator to copy the character buffer to the MCR command line, specify the MAP attribute without specifying an associated syntax element, as follows:

```
<MAP=n>
```

When the translator processes a MAP attribute that has no associated syntax element, the translator examines the mapping number queue element for an attached character buffer. If a character buffer is found, the translator copies the contents of the character buffer to the forming MCR command line buffer. If a character buffer is not found, the translator assumes a null syntax element (which always processes successfully).

The Optional Copy (OCOP) attribute is identical to COP except for one significant difference. OCOP instructs the parser to copy the literal to the IFORM whether or not the processing of the literal succeeds. That is, the literal is optional and always succeeds if you associate OCOP with it.

Examples

The following code defines a terminal specification:

```
TERM:   OR <'TT COP=T AND=N>
        OR. <'HT COP=T AND=N>
        OR. <'VT COP=T AND=N>
        OR. <'TI COP=T AND=N>
        END
N:      AND <$ONUM,0,3><': OCOP=T>
```

This code parses and copies (as terminal names) the letters TT, VT, HT, or TI, followed by an octal number. (See Section 18.10.1 for a description of $ONUM, which defines an octal number.) The OCOP attribute assures the presence of the terminating colon by copying it into a character buffer (whether or not the user types a colon).

The following example illustrates the use of the OCOP attribute:

```
 DEV:   AND      <$ALF,2,2><$NUM,0,3><': OCOP=T>
```

The MML definition defines a device specification such that the colon on the end of the specification is optional. The translator copies the colon to the output MCR command line whether or not a colon is present in the input DCL command line.

18.9.6 DIR

The Direction (DIR) attribute is valid only in the TT and only when you specify it with a MAP attribute. DIR instructs the translator to search the IFORM queue for the specified mapping number. The search procedes from the end of the queue to the beginning of the queue. (If you do not specify the DIR attribute, the direction of the search is from the beginning of the queue to the end of the queue.)

Format

```
DIR=R
```

Note

The double angle brackets are necessary when the contents of the brackets include a MACRO–11 separator character (in this case, the comma) that is to be treated as a literal by MML.

Example

The following example of a TT fragment illustrates the use of the DIR attribute:

```
FILE:   AND     <MAP=1 DIR=R <MORE OPT=T>
        END

MORE:   AND     <<',>>FILE
        END
```

This example defines, recursively, the structure of a file specification list. The file specifications are mapped to mapping number 1. Due to the associated DIR attribute, the translator searches for the mapping numbers in reverse order; thus, the list of file specifications is output to the MCR command line in reverse order.

18.9.7 ERR

The Error (ERR) attribute allows you to associate error messages with syntax elements. If the parser or the translator fails to process a syntax element, and the syntax element has ERR specified, then the parser or translator prints the error message specified and aborts the parse or translation.

Formats

```
        ERR='string

        ERR=label
label:  ERROR </string/>
```

Example

To specify an error message for the device being assigned, use the following MML notation in the PT for the ASSIGN command:

```
 AND     'ASSIGN <QUAL OPT=T><$DEV ERR=<'Invalid device>>
```

If the parser cannot parse the input DCL command line text that corresponds to the device specification $DEV, the parser aborts the parse and prints the following message:

```
ASSIGN -- Invalid device
```

18.9.8 HLP

In a PT, the Help (HLP) attribute associates help text with syntax elements that are specified with the PRO attribute (see Section 18.9.12). When the parser prompts for input and a user types a question mark (?), the parser spawns a HELP command. The parser appends the text specified in the HLP attribute to "HELP command." The "command" is the name of the DCL command being processed.

Formats

```
        HLP='string

        HLP=label

label:  HELP </string/>
```

Arguments

string

Specifies the string appended to the HELP command.

label

Specifies the label of a HELP macro.

The parser appends the string to the command "HELP command," where "command" is the name of the current command being processed.

Example

```
 AND     <QUAL OPTR=T><$DEV PRO='Device? HLP=<'Logical name>>
```

If you use this format to specify the PT for ASSIGN and a user types a question mark (?) in response to the Device? prompt, the parser forms and spawns the command HELP ASSIGN LOGICAL NAME.

This PT requires a file named DCLASSIGN.HLP containing the necessary help text. For a description of help file format, see the *RSX-11M-PLUS Command Language Manual* or the *RSX-11M-PLUS MCR Operations Manual.*

18.9.9 MAP and DMAP

The MAP attribute is valid in both the PT and the TT. However, the parser and the translator interpret MAP differently. The following MAP attributes are valid in a parser table:

MAP=n — Instructs the parser to queue the specified primary mapping number n to the IFORM queue only if the associated syntax element is successfully parsed.

MAP= <,n1> — Instructs the parser to queue the specified secondary mapping number n1 to the IFORM queue only if the associated syntax element is sucessfully parsed.

Note that specifying COP with MAP in a PT instructs the parser to create and attach a character buffer to the mapping number and to copy the text parsed by the associated syntax element to the attached character buffer. For additional details, see Section 18.10.3, which describes COP.

The following MAP attributes are valid in a translator table:

MAP=n — Instructs the translator to search the IFORM queue from beginning to end for the specified primary mapping number n. If n is found, the translator places the element in the MCR command line. If n is not found, nothing happens.

MAP= <,n1> — Instructs the translator to search the IFORM queue for the secondary mapping number n1. Because the object of the search is a secondary (not primary) mapping number, the translator does not search the entire IFORM queue. The translator starts searching at the last used primary mapping number and stops the search at the next primary mapping number. If the secondary mapping number is found, then the associated syntax element is processed; otherwise, nothing happens.

MAP= <n,n1> — Instructs the translator to search the IFORM queue from beginning to end for the specified primary mapping number n.

If n is found, the translator searches for the specified secondary mapping number n1. If n1 is found, the associated syntax element is processed. If n1 is not found, then the translator searches for another occurrence of primary mapping number n.

This search process continues until either all primary mapping numbers n have been found and none of them have a secondary mapping number n1 after them, or until a primary mapping number n is found that does have a secondary mapping number n1 after it.

If you specify a MAP attribute in a TT without specifying an associated syntax element, the translator examines the queue element containing the specified mapping number for an attached character buffer. If the translator finds an attached character buffer, the translator copies the contents of the buffer to the forming MCR command line buffer. If the translator does not find an attached character buffer, the translator assumes a null syntax element (which always processes successfully).

To allow additional flexibility in the search for secondary mapping numbers, the translator supports the following additional formats for specifying the MAP attribute in a TT:

MAP= <X,n1> Instructs the translator to search the IFORM queue from beginning to end for the specified secondary mapping number n1. If n1 is found, the translator processes the associated syntax element. If n1 is not found, the associated syntax element fails to process.

The keyword X identifies this MAP attribute as an unconditional search for a secondary mapping number.

MAP= <X1,n1> Instructs the translator to seach the IFORM queue for the secondary mapping number n1. The keyword X1 instructs the translator to limit the search of the queue. The search is started at the beginning of the queue and proceeds until the first primary mapping number is detected.

If the secondary mapping number n1 is not found, then another search is conducted. The second search looks for the secondary mapping number n1 such that it follows a negative primary mapping number. If n1 is found following this search, the translator processes the associated syntax element. If n1 is not found, nothing happens.

The MAP attribute X searches for a secondary number as if it were a primary number (that is, unconditionally). The MAP attribute X1 searches for floating command qualifiers. (For information on floating command qualifiers, see Section 18.9.13, which describes QUA.)

When the translator finds a mapping number and processes the associated syntax element, the translator marks the mapping number as being in use. A used mapping number cannot be found by the translator in a subsequent search. Thus, marking mapping numbers prevents infinite loops in the TT. You do not need to specify termination conditions.

To override the marking of mapping numbers, use the TST attribute. To reset all marked mapping numbers, use the $RESET predefined syntax element (PSE; see Section 18.10). $RESET is useful for generating multiple MCR commands with the $NEW predefined syntax element.

The Duplicate map (DMAP) attribute is identical to MAP in most ways. However, it is only allowed in a PT. DMAP instructs the parser not to queue the mapping number to the IFORM queue if the same mapping number is already present in the queue. If the parser does not queue the mapping number, then the syntax element that MAP is associated with fails to process. Thus, the DMAP attribute can prevent duplicate syntax.

For additional information on the use of the MAP attribute, see Section 18.5.5.

When translating the DCL command MACRO to the MCR command MAC, the translator must determine which (if any) input file specification has the qualifier /LIST appended to it. For example, the translator must be able to determine (by means of information in the TT) that the file specification C is specified with the /LIST qualifier in the following DCL command line:

```
MACRO A,B,C/LIST,D
```

The IFORM maps the DCL file specification that has the /LIST qualifier appended to it to the correct position in the MCR command line.

The mapping can be understood more easily from the perspective of the translator. The translator generates the MCR command MAC from left to right. When the translator reaches the position in the output MCR command line where the list file must be placed, the translator looks in the IFORM queue for the mapping number that represents the file specification and the /LIST qualifier. Since the file specification and the qualifier are two distinct syntax elements, there are two mapping numbers in the IFORM queue to represent them.

To express an association between elements in the IFORM queue, use primary and secondary mapping numbers. Secondary mapping numbers (such as S1) have meaning only as related to primary mapping numbers (such as P1).

Due to the recursive definition of FILES, which represents the input file specifications, identical map numbers may be placed in the IFORM queue. However, the left-to-right positioning of the P1 map numbers distinguishes them from one another. To find the P1 map number that represents the file specification with the /LIST qualifier, the translator searches for a P1 map number directly followed by an S1 map number. There can be no intervening Pn map numbers. To instruct the translator to perform such a search in MML, use the argument <MAP= <1,1> > (see the label LST).

If a reference is made to a secondary map number without specifying a primary map number, the translator looks for a secondary map number associated with the primary map number that was used last.

Examples

```
            Parser Table

TERM:  AND 'TERMINAL: <$DEV,MAP=2>
       END

            Translator Table

MRET:  AND 'TERM= <MAP=2>
       END
```

Shows the use of primary mapping numbers with predefined syntax elements.

DCL file specifications, device specifications, and numbers map to themselves. So, when a user enters a file specification, device specification, or number, the same information is output.

In this example, a terminal device specification is being mapped to itself. The parser table specifies the following sequence of instructions at label TERM:

1. Place the mapping number P2 in the IFORM queue if the input command line matches a device specification $DEV.
2. Attach a character buffer to this queue element.
3. Copy the device specification to the attached character buffer.

The translator table specifies the following instruction at label MRET:

> If P2 is in the IFORM queue, then output the contents of any attached buffer to the forming MCR command line.

Parser Table

```
        AND     'MACRO FILES
        END
FILES:  AND     <$FILE MAP=1><'/LIST OPT=T MAP=<,1>><MORE OPT=T>
        END
MORE:   AND     <<',>>FILES
        END
```

Translator Table

```
        AND     'MAC <OBJ OPT=T><LST OPT=T>'= SELIF
        END
LST:    AND      <<',>><MAP=<1,1>>
        END
        SELIF:
   .
   .
   .
```

Illustrates the use of secondary mappings.

18.9.10 NO

The NO attribute is allowed only in a PT. NO instructs the parser to process either a literal (NO) or a literal (-) before it processes the associated literal. The NO attribute is valid only when associated with a literal syntax element.

Format

```
NO=T
```

If you specify a MAP with the NO attribute, the parser queues two different mapping numbers. If the parser does not process either the NO or the hyphen (-), the parser queues the mapping number assigned to MAP. If the parser processes either the NO or the hyphen (-), the parser increments the mapping number by one. Then, the parser queues the mapping number.

Example

```
 OR      <'LIST MAP=10. NO=T>
```

Illustrates the use of the NO attribute. In this example, the parser attempts to parse either NOLIST, -LIST, or LIST. If LIST is parsed, the parser adds the mapping number 10 to the IFORM queue. If either NOLIST or -LIST is parsed, the parser queues the mapping number 11.

18.9.11 OPT and OPTR

The Optional (OPT) attribute indicates that a syntax element is not required. You can use OPT in both the PT and the TT. The parser and the translator interpret OPT identically.

Use the Optional Repeat (OPTR) attribute in the same way to indicate that a syntax element is not required. In addition, use OPTR to indicate that the syntax element can be used repeatedly in the parse/translation until a failure results. In other words, the presence of OPTR indicates that the syntax element can be used to parse or translate syntax from zero to an infinite number of times.

Formats

```
OPT=T

OPTR=T
```

If the OPT and AND attributes are used in the same syntax element, the OPT modifies the AND. That is, the OPT does not modify the entire syntax element.

Examples

To specify that qualifiers (label QUAL) are optional syntax in the ASSIGN command, use the following MML notation:

```
 AND     'ASSIGN <QUAL OPT=T>...
```

The parser goes to label QUAL and processes the syntax defined at label QUAL. If the parser cannot process a syntax element in an AND macro, it assumes that the AND macro itself has failed. The OPT=T attribute tells the parser to ignore the failure and to continue processing the AND macro with the next left-to-right argument to the macro.

Similarly, to specify that qualifiers (label QUAL) are optional syntax in the ASSIGN command and that the qualifiers can be used repeatedly in the input command line, use the following MML notation:

```
 AND     'ASSIGN <QUAL OPTR=T>...
```

The parser goes to label QUAL and processes the syntax defined at the label. If the syntax element in the AND macro fails, the element cannot be used in the parse. If the syntax element was required, the failure of the element would imply the failure of the AND macro. However, the OPTR=T attribute tells the parser that the syntax element is optional; so, the parser ignores the failure and continues processing the AND macro with the next left-to-right argument. As long as the syntax element defined at label QUAL matches the input command, the parser uses the element repeatedly.

The following example is a fragment of code based on the LINK command translator table. The TT specifies a TKB switch that can be specified with or without arguments attached.

```
        OR.      <'/PR MAP=1 AND=PRIV OPT=T>
        END
                 .
                 .
                 .
PRIV:   AND      ': <MAP=3>
        END
```

In the example, the optional argument to the /PR switch is attached to mapping number 3. The OPT applies to AND=PRIV and not to the entire syntax element. If the OPT attribute is applied to the entire syntax element, then the entire OR macro (of which this fragment is a part) would always succeed.

18.9.12 PRO

The Prompt (PRO) attribute associates prompt strings with syntax elements. If the parser detects end-of-line while processing a syntax element that has the PRO attribute specified, the parser prints the prompt string specified, and suspends itself to wait for more input. The parser appends any input received to the input DCL command line. PRO is valid only in the PT; the translator does not prompt.

Formats

```
        PRO='string

        PRO=label

label:  PROMPT </string/>
```

Arguments

string

Specifies the ASCII string to be used as the prompt string.

label

Specifies the MACRO–11 label for a MACRO–11 PROMPT macro.

Thus, specify the prompt string directly after the equal sign (=) or as the argument to the PROMPT macro. The PROMPT macro allows global prompt strings, which conserve space.

Example

The following example illustrates the use of the PRO attribute:

```
AND  <QUAL OPTR=T><$DEV PRO='Device?><$DEV PRO=PRODEV>

PRODEV:   PROMPT </Logical device? />
```

The command requires two device specifications ($DEV). If the parser expects the first $DEV but reaches the end of the line, then the parser prompts as follows:

```
Device?
```

If the parser expects the second $DEV but reaches the end of the line, then the parser prompts as follows:

```
Logical device?
```

18.9.13 QUA

The Qualifier (QUA) attribute specifies qualifiers. You can specify floating or positional qualifiers. *Floating qualifiers* are qualifiers that are valid anywhere in the input DCL command line. *Positional qualifiers* are qualifiers that are valid only at specific positions in the input DCL command line or output MCR command line.

The positional form of the QUA attribute is valid in both a PT and a TT. The floating form is valid only in a PT.

Format

```
QUA=label
```

The *label* is a nonliteral; that is, it represents the address of an AND or an OR macro.

When the QUA attribute is used in a PT and the parser encounters a slash (/) in an appropriate location, it attempts to match the text following the slash with one of the elements of the AND or OR macro at the specified label. If an end-of-line is encountered after the slash, the parser issues a prompt. If an invalid qualifier is given, then an error message is printed and the parse is aborted. The list of qualifiers is repeatedly scanned until all qualifiers in the input command line have been parsed.

To specify a positional QUA attribute, use QUA as part of a syntax element. For example:

```
        AND     <'ASSIGN,3 QUA=QUAL>$DEV $DEV
        END

QUAL:   OR      <'GLOBAL,1>
        OR.     <'LOCAL,3>
        OR.     <'LOGIN,3>
        OR.     TERM
        END
```

In this example, any qualifiers (such as /GLOBAL) must immediately follow ASSIGN in the command line. (See the Examples section for more details on how the QUA attribute works.)

To specify floating qualifiers, use QUA as a separate argument to an AND macro, not as part of another argument. The parser attempts to match the qualifiers defined at the label before and after it processes each argument of the AND macro. That is, the qualifiers float before and after each syntax element defined by the AND macro. In addition, the parser attempts to parse the floating qualifiers after any prompts. If you are using QUA to specify floating qualifiers, it must be an argument to an AND macro, rather than an AND. extension macro. The floating-qualifier form of QUA is valid only in a PT.

Examples

```
        AND     'ASSIGN, 3 $DEV $DEV QUA=QUAL
        END

QUAL:   OR      <'LOCAL,3>
        OR.     <'LOGIN,3>
        OR.     <'GLOBAL,1>
        OR.     TERM
        END
```

Illustrates the floating-qualifier form of QUA. The label QUAL defines the allowable qualifiers. The parser checks for the qualifiers before, between, and after the $DEV syntax elements. Before the parser transfers parsing control to the OR macro at label QUAL, the parser attempts to parse the slash (/) character.

If the parser successfully parses the slash, it attempts to parse the qualifiers defined at label QUAL. If the parser detects the end of the input DCL command line after it parses a slash character, the parser prompts for more input.

If a slash is detected and the qualifiers defined at label QUAL fail to process, the parser issues an error message and aborts the parse. The parser repeatedly attempts to parse the qualifiers until it no longer can parse a slash character.

Thus, using QUA in the previous example is equivalent to the following:

```
        AND     <QUAL OPTR=T>$DEV <QUAL OPTR=T>$DEV <QUAL OPTR=T>
        END

QUAL:   AND     '/ <QUAL1 PRO='Qualifier? ERR=<'Illegal qualifier>>

QUAL1:  OR      <'LOCAL,2>
        OR.     <'LOGIN,2>
        OR.     <'GLOBAL,1>
        END
```

The following example illustrates the use of the QUA attribute in a TT:

```
ASN::   AND <<'ASN>><MAP=1>'= <MAP=2 QUA=LAUQ>
        END

LAUQ:   OR <'GBL     MAP=4>
        OR. <'LOGIN  MAP=3>
        OR. MRET
        END
```

The label LAUQ defines the MCR switches equivalent to the DCL qualifiers. The parser and the translator process QUA in the same way. However, instead of parsing the input DCL command line for qualifiers, the translator uses the qualifiers to construct the equivalent MCR command line.

QUA automatically includes slashes before each switch.

18.9.14 TST

The Test (TST) attribute is valid only in a TT and only when specified with MAP. This attribute instructs the translator not to mark a mapping element used in a translation as having been used.

Format

```
TST=T
```

Example

```
MCRCOP: OR      <PIP      MAP=1 TST=T>
        OR.     <PIP      MAP=2 TST=T>
        OR.     <NFT      MAP=3>
        END
```

Illustrates the use of the TST attribute. In this example, the OR macros instruct the translator to process the nonliteral PIP if either mapping numbers 1 or 2 are in the IFORM queue or to process the nonliteral NFT if mapping number 3 is in the IFORM queue. The TST attribute associated with the syntax element PIP instructs the translator not to mark mapping numbers 2 or 3 as having been used. This allows the syntax defined at label PIP to use these mapping numbers.

18.10 Predefined Syntax Elements

Predefined syntax elements (PSEs) are syntax elements defined for use by all DCL command tables. PSE names, by convention, are distinguished from local category names by having a dollar sign ($) as their first character.

If the syntax of a command does not match that of a predefined syntax element, then the parse fails. Several PSEs have two forms: one ends with an E and one does not, such as $NOD and $NODE, $FIL and $FILE, or $DEV and $DEVE. The form ending in an E produces an error message before aborting the parse; the shorter form does not. Use the shorter form when defining optional syntax.

The following list briefly defines the PSEs. Note that some PSEs require arguments. Specify the required arguments in the order shown. Also, to allow the MACRO–11 assembler to treat the PSE and arguments as one argument, enclose the PSE and its arguments in angle brackets.

18.10.1 Basic Data Types

The following list of PSEs define basic data types in MML. You can use them only in a PT. The parser copies to an IFORM character buffer the text in the input DCL command line that fits the structure defined by the PSE. The buffer is attached to the last queued mapping number and can be accessed by the translator for use in constructing the equivalent MCR command line.

Basic Data Type Syntax Elements

$NUM,A1,A2

Defines the structure or syntax of a decimal numeric string (0 to 9). The arguments A1 and A2 designate the minimum and maximum length, respectively, for the numeric string.

The PSE allows an optional decimal point on the end of the string; however, the decimal point is not copied to the IFORM character buffer.

$DNUM,A1,A2

Defines the structure or syntax of a decimal numeric string (0 to 9). The arguments A1 and A2 designate the minimum and maximum length, respectively, for the numeric string.

The PSE allows an optional decimal point on the end of the string. The decimal point is copied to the IFORM character buffer (whether or not it is specified in the input DCL command line).

$ONUM,A1,A2

Defines the structure or syntax of an octal numeric string (0 to 7). The arguments A1 and A2 designate the minimum and maximum length, respectively, for the numeric string.

$WONUM,A1,A2

Defines the structure or syntax of $ONUM or the wildcard character (*). The arguments A1 and A2 designate the minimum and maximum length, respectively, for the numeric string.

$DONUM,A1,A2[.]

Defines the structure or syntax of a decimal or octal numeric string (0 to 9). The arguments A1 and A2 designate the minimum and maximum length, respectively, for the numeric string.

The PSE allows an optional decimal point on the end of the string. The decimal point is copied to the IFORM character buffer if either the input DCL command line or the parsed string contains the characters 8 or 9.

$WDONUM,A1,A2

Defines the structure or syntax of $DONUM or an asterisk character (*). The arguments A1 and A2 designate the minimum and maximum length, respectively, for the numeric string.

$BNUM,A1,A2

Defines the structure or syntax of a binary numeric string (0 to 1). The arguments A1 and A2 designate the minimum and maximum length, respectively, for the numeric string.

$ALF,A1,A2

Defines the structure or syntax of an alphabetic string (A to Z). The arguments A1 and A2 designate the minimum and maximum length for the alphabetic string.

$ALFN,A1,A2

Defines the structure or syntax of an alphanumeric string (0 to 9 and/or A to Z). The arguments A1 and A2 designate the minimum and maximum length for the alphanumeric string.

$WALFN,A1,A2

Defines the structure or syntax of $ALFN with asterisk (*) and percent sign (%) characters. (These characters are used by some RSX-11M-PLUS utilities as wildcard characters.) The arguments A1 and A2 designate the minimum and maximum length for the alphanumeric string.

$R50,A1,A2

Defines the structure or syntax of a Radix-50 string (0 to 9, A to Z, dollar sign, and period). The arguments A1 and A2 designate the minimum and maximum length for the Radix-50 string.

18.10.2 Common Command Elements

The following PSEs define parts of the DCL command grammar. You can use them only in a PT. The text (except for $DATE) parsed from the input DCL command line that fits the structure defined by the PSE is copied to an IFORM character buffer that is attached to the last queued mapping number.

Common Command Syntax Elements

$FIL

Defines the structure or syntax of a file specification. Each element explicitly included in the file specification sets a bit in the global bit-mask words, as follows:

Element	Bit Setting
Device	B28
User Identification Code (UIC)	B29
File name	B30
File type	B31
File version	B32

These bits are also set by the other $FIL-based predefined syntax elements.

$FILE

Defines the structure or syntax of $FIL. Also, if the parser fails to process $FILE, the parser issues an error message and aborts the parse.

$WFIL

Defines the structure or syntax of $FIL. Also, the wildcard characters (the asterisk and the percent sign) are allowed as part of the file specification.

$WFILE

Defines the structure or syntax of $WFIL. Also, if the parser fails to process $WFILE, the parser issues an error message and aborts the parse.

$NFIL

Defines the structure or syntax of $WFIL if preceded by a DECnet node specification.

$NFILE

Defines the structure or syntax of $NFIL. Also, if the parser fails to process $NFILE, the parser issues an error message and aborts the parse.

$IFIL

Defines the structure or syntax of $FIL. Also, the indirect file indicator (@) must prefix the file specification.

$IFILE

Defines the structure or syntax of $IFIL. Also, if the parser fails to process $IFILE, the parser issues an error message and aborts the parse.

$NOD

Defines the structure or syntax of a DECnet node name specification.

$NODE

Defines the structure or syntax of $NOD. Also, if the parser fails to process $NODE, the parser issues an error message and aborts the parse.

$DEV

Defines the structure or syntax of a device specification.

$DEVE

Defines the structure or syntax of $DEV. Also, if the parser fails to process $DEVE, the parser issues an error message and aborts the parse.

$ODEV

Defines the structure or syntax of $DEV. Also, the colon character (:) is optional syntax.

$ODEVE

Defines the structure or syntax of $ODEV. Also, if the parser fails to process $ODEVE, the parser issues an error message and aborts the parse.

$UIC

Defines the structure or syntax of a UFD specification.

$UICE

Defines the structure or syntax of $UIC. Also, if the parser fails to process $UICE, the parser issues an error message and aborts the parse.

$WUIC

Defines the structure or syntax of $UIC. Also, the wildcard character (the asterisk) is allowed as part of the UIC specification.

$WUICE

Defines the structure or syntax of $WUIC. Also, if the parser fails to process $WUICE, the parser issues an error message and aborts the parse.

$OUIC

Defines the structure or syntax of $UIC. Also, the left square bracket ([ ), the right square bracket ( ]), and the comma (,) are optional syntax.

$OUICE

Defines the structure or syntax of $OUIC. Also, if the parser fails to process $OUICE, the parser issues an error message and aborts the parse.

$DATE

Defines the structure or syntax of a date specification. A date specification can take one of two forms: mm/dd/yy or dd-mmm-yy. $DATE produces the following data structures in the IFORM queue: the day is attached to secondary mapping number 210, the month is attached to secondary mapping number 211, and the year is attached to secondary mapping number 212. The output format is always dd-mmm-yy (with the first three characters of the month name spelled out) even if the slash format (with the number of the month) is used.

$DATE1

Defines secondary mapping numbers 213, 214, and 215 for the day, month, and year, respectively. Otherwise, $DATE1 is identical to $DATE.

$DATE1 serves as a second PSE for commands that require two dates (for example, SINCE and THROUGH).

$TIME

Defines the structure or syntax of a time specification. The time specification is of the form hh:mm[:ss].

18.10.3 Special Parser Elements

You can specify the following PSEs only in a PT.

Special Parser Syntax Elements

$ALL,label

Defines the syntax or structure of an arbitrary string of characters. $ALL matches all characters in the command line (left-to-right) until any one of a set of specified terminating characters is detected. The set of terminating characters are specified with the TERMINATORS macro. The label is a label on the TERMINATORS macro. The text parsed from the input DCL command line that fits the structure defined by $ALL is copied to an IFORM character buffer that is attached to the last queued mapping number.

The following is an example of $ALL with the TERMINATORS macro:

```
                  .
                  .
                  .
        AND. <$ALL,EXCEPT>
                  .
                  .
                  .
EXCEPT: TERMINATORS '"  ') '] $EOL
                  .
                  .
                  .
```

This example allows commas, quotation marks ("), and right angle brackets (]) as terminators, in addition to the $EOL predefined syntax element.

$EOL

Parses end-of-line characters. For instance, in the previous example, a carriage return or escape character is also acceptable as a terminator.

$COP,'lit

Instructs the parser to copy the specified literal, 'lit, unconditionally to an IFORM character buffer. The IFORM character buffer is always attached to the last queued mapping number.

$COP does not define syntax.

$DSB

Instructs the parser not to skip blanks. Normally, blanks are ignored. $DSB and $ESB must be in the same syntax element, such as AND or OR, but they do not need to be on the same line.

$DSB (disable skip blanks) does not define syntax.

$ESB

Restores the default state, which is to ignore blanks.

For example, a file specification cannot include blanks. The following could define a file specification:

```
$DSB $DEV $UIC <$ALFN,0,9>'. <$ALFN,0,3>'; <$ONUM,0,3>$ESB
```

(The actual file specification definition is more complex. This is only an example.)

$ESB (enable skip blanks) does not define syntax.

$DDR

Instructs the parser to disable delimiter recognition. Delimiter recognition is enabled by default and instructs the parser to ensure that all literal syntax elements have delimiting characters terminating them. $DDR must be used on the same line with $EDR.

$DDR does not define syntax.

$EDR

Enables delimiter recognition. This is the default case.

For example, the following line allows you to attach the first one to three characters to one mapping number while the remainder are attached to another.

```
COM1: AND $DDR <$R50,1,3 MAP=1><$R50,0,7 MAP=2>$EDR
      END
```

Notice that no delimiting character distinguishes the two strings, and the second string may be null.

$SEP

Defines the colon (:) or equal sign (=) for use as separators between qualifiers and arguments.

$OSEP

Defines an optional separator (a colon, equal sign, or null argument).

$CSEP

Defines the colon (:) or equal sign (=) for use as separators between qualifiers and arguments. Then, it copies a colon to the IFORM.

$COSEP

Defines an optional separator (a colon, equal sign, or null argument) and copies a colon to the IFORM (even if no input is received).

$FSEP

Defines the comma (,) and plus sign (+) as file specification separators.

$TERM

Defines a terminal specification and copies it to the IFORM.

$PROC

Defines a processor name for the Queue Manager (QMG) as either a device name followed by a colon or six Radix–50 characters followed by a colon.

$PRONC

Defines a processor name for the Queue Manager (QMG) as a device name followed by a colon or six Radix–50 characters not followed by a colon.

18.10.4 Special Translator Elements

You can specify the following PSEs only in a TT.

Translator Syntax Elements

$NEW

Indicates the start of an additional MCR command line. $NEW instructs the dispatcher subprocess of the DCL task that there are multiple MCR commands that must be dispatched serially.

$RESET

Resets the IFORM queue. Resetting the IFORM queue allows all used mapping numbers to be reused. $RESET is designed to be used with $NEW to define multiple MCR commands.

$FNAM

Specifies that only the file name in a file specification is to be copied to the MCR command line buffer as a result of unmapping a mapped file specification.

$SY

Converts the assigned device for SY: to ASCII and appends it to the MCR command line; for example, DB2.

$TI

Converts the assigned terminal device for TI: to ASCII and appends it to the MCR command line; for example, TT15.

$CUIC

Converts the current UIC for the user to ASCII and appends it to the MCR command line; for example, [301,113].

$LUIC

Converts the login UIC for the user to ASCII and appends it to the MCR command line.

18.10.5 Program Control Elements

You can specify the following predefined syntax elements in either a PT or a TT.

Program Control Syntax Elements

$JSR,A

Gains control of the central processing unit (CPU). The argument A must be the label of a MACRO–11 subroutine. To indicate failure, the MACRO–11 subroutine should (on return) set the carry bit to indicate failure and clear it to indicate success.

The routine must be in the blank program section.

$TST

Tests if a mapping number is in the IFORM queue. A MAP attribute specifies the mapping number. $TST succeeds only if the mapping number is in the IFORM queue; for example, <$TST MAP=1> .

$TSTNOT

Tests if a mapping number is not in the IFORM queue. A MAP attribute specifies the mapping number. $TSTNOT succeeds only if the mapping number is not in the IFORM queue. The syntax is identical to $TST.

$LOAD,name

Loads an overlay from a parser or translator table. The argument name specifies the name of the overlay as specified in the overlay description file.

$CALLOV,name

Loads and activates part of either a parser or translator table. The starting address within the overlay must be contained in the first word of the overlay. Use .WORD to specify the address. On return, the previous overlay is loaded. The argument (name) specifies the name of the overlay as specified in the overlay description file.

$NULL

Executes attributes without specifying a syntax element. $NULL succeeds if the attributes succeed.

$AB

Passes the left angle bracket (<) as an argument to the MACRO–11 assembler. $AB (angle bracket) is equivalent to the left angle bracket literal.

$BA

Passes the right angle bracket (>) as an argument to the MACRO–11 assembler. $BA is equivalent to the right angle bracket literal.

18.10.6 Example from DCL Command Tables

The following is an example of the DCL command tables used to translate the DCL commands ASSIGN, ASSIGN/QUEUE, ASSIGN/REDIRECT, and ASSIGN/TASK, to the equivalent MCR commands (ASN, QUE, RED, and REA).

```
                .TITLE  ASSIGN
                .IDENT  /00/
                .PSECT  .CDEF
                .ENABL  LC
;
; THIS  SOFTWARE  IS FURNISHED UNDER A LICENSE AND MAY
; BE USED OR COPIED ONLY IN ACCORDANCE WITH THE  TERMS
; OF SUCH LICENSE.
;
; COPYRIGHT (c) 1987 BY DIGITAL EQUIPMENT CORPORATION.
; ALL RIGHTS RESERVED.
;
; FUNCTION:
;
;       TRANSLATE THE DCL COMMANDS 'ASSIGN', 'ASSIGN/QUEUE', 'ASSIGN/REDIRECT',
;       'ASSIGN/TASK' TO THE SEMANTICALLY EQUIVALENT MCR COMMANDS 'ASN', 'QUE',
;       'RED', 'REA'.
;

;THE FOLLOWING TABLE DEFINES THE MAPPING OF THE DCL TO MCR SYNTAX:
;
;DCL SYNTAX ELEMENT      MCR SYNTAX ELEMENT      MAPPING LETTER/NUMBER
;------------------      ------------------      ---------------------
;
;<queuename>             <queuename>             P1
;<processorname>         <processorname>         P2
;
;<olddevice>             <olddevice>             P3
;<newdevice>             <newdevice>             P4
;
;<taskname>              <taskname>              P5
;<device>                <device>                P6
;<lun>                   <lun>                   P7
;
;<device>                <device>                P8.
;<logicaldevice>         <logicaldevice>         P9.
;/GLOBAL                 /GBL                    P10.
;/LOGIN                  /LOGIN                  P11.
;/TERMINAL=<termdev>     /TERM=<termdev>         P12.
;
;

        .PAGE
ASSIGN::
        OR      <QUEUE BS=B1><REDIR BS=B2><TASK BS=B3><LOGDEV BS=B4>
        END

QUEUE:  AND     '/ <'QUEUE,1><$PRONC MAP=1 PRO=$QNMSG ERR=$BQMSG>
        AND.    <$PROC MAP=2 PRO=$PRMSG ERR=$IPMSG>
        END
```

```
REDIR:  AND     '/ <'REDIRECT,1><$ODEVE MAP=3 PRO=$FRMSG>
        AND.    <$ODEVE MAP=4 PRO=$TOMSG>
        END

TASK:   AND     '/ <'TASK,2>$OSEP <$R50,1,6 MAP=5 PRO=$TKMSG ERR=$ITMSG>
        AND.    <$ODEVE MAP=6 PRO=$DVMSG>
        AND.    <$DNUM,1,5 MAP=7 PRO=$LUMSG ERR=$NEMSG>
        END

                 .TITLE  ASSIGN
                 .IDENT  /00/

                                     THE DCL TASK

LOGDEV: AND     <$ODEVE MAP=8. PRO=$DVMSG><$ODEVE MAP=9. PRO=$LDMSG>QUA=QUAL
        END
QUAL:   OR      <'GLOBAL,1 MAP=10. CBM=B1!B2>
        OR.     <'LOCAL,3 CBM=B1>
        OR.     <'LOGIN,3 MAP=11. CBM=B1>
        OR.     <'SYSTEM,1 MAP=10. CBM=B1!B2>
        OR.     <$TERM MAP=12. CBM=B2>
        END
        .PAGE
MCRASN::
        OR      <QUE BST=B1><RED BST=B2><REA BST=B3><ASN BST=B4>
        END

QUE:    AND     <<'QUE>><MAP=2>'/AS: <MAP=1>
        END

RED:    AND     <<'RED>><MAP=4>'= <MAP=3>
        END

REA:    AND     <<'REA>><MAP=5><<'>><MAP=7><<'>><MAP=6>
        END

ASN:    AND     <<'ASN>><MAP=8.>'= <MAP=9.><QUA=LAUQ>
        END
LAUQ:   OR      <'GBL MAP=10.>
        OR.     <'LOGIN MAP=11.>
        OR.     <'TERM= AND=VTERM>
        END
VTERM:  AND     <MAP=12.>
        END
        .ENABLE LC
$BQMSG: ERROR   </Illegal queue name/>
$DVMSG: PROMPT  </Device? />
$FRMSG: PROMPT  </From? />
$IPMSG: ERROR   </Illegal processor name/>
$ITMSG: ERROR   </Illegal task name/>
$LDMSG: PROMPT  </Logical device? />
$LUMSG: PROMPT  </Logical unit? />
$PRMSG: PROMPT  </Processor? />
$QNMSG: PROMPT  </Queue? />
$TKMSG: PROMPT  </Task? />
$TOMSG: PROMPT  </To? />
        .DSABLE LC
        .END
```

Chapter 19

Using the Catchall Task

The RSX–11M–PLUS and Micro/RSX operating systems include a catchall task (TDX) that "catches" commands that are not recognized by the DIGITAL Command Language (DCL) or the Monitor Console Routine (MCR). If MCR receives an unrecognized command, it searches for a task with that name and passes the command line to TDX. TDX allows you to run uninstalled tasks and to abbreviate command names.

RSX–11M–PLUS and Micro/RSX systems install TDX on your system for you. However, any task installed with the task name ...CA. is treated as a catchall task. The catchall task image is in the system library directory (usually directory [3,54]) and is named TDX.TSK. Once installed, TDX checks the typed command against its list of commands. If the commands match, TDX translates the command into a valid MCR command. The following section describes the TDX commands and their corresponding MCR commands.

19.1 TDX Commands

TDX commands are a shorter way of specifying MCR commands and qualifiers. Table 19–1 lists the TDX commands, their MCR translations, and the definition for each command.

Table 19–1: TDX Command Summary

TDX Command	MCR Translation	Command Meaning
ATS	ACT /ALL	Displays the names of all active tasks in the system.
ATS ttnn:	ACT /TERM=ttnn:	Displays the names of all active tasks on the specified terminal.
CHD	SET /DEF	Displays the current default directory for terminal TI:.

Table 19–1 (Cont.): TDX Command Summary

TDX Command	MCR Translation	Command Meaning
CHD g m	SET /DEF=[g,m]	Changes the current default directory to the directory specified in named directory mode. In nonamed directory mode, changes the current default directory and, if privileged, the protection User Identification Code (UIC).
CHU	SET /UIC	Displays the protection UIC for terminal TI: and, if in nonamed directory mode, the default User File Directory (UFD).
CHU g m	SET /UIC=[g,m]	Changes the current protection UIC to the UIC specified in named directory mode (privileged command). In nonamed directory mode, changes the default directory and, if privileged, the protection UIC.
CLR	None	Clears the issuing terminal's screen and sets the cursor to 0,0. Returns exit status of EX$SUC (if the terminal is a video terminal) and EX$WAR (if it is not).
CRE file	PIP file=TI:	Creates a new file without invoking an editor.
CVT val	None	Evaluates an arithmetic expression, converts that expression into different formats, and displays all the formats on your terminal. CVT accepts input in octal (nnn or nn,nn) or decimal (nnn. or nn.,nn.) words or bytes, hexadecimal numbers ($nnnn), Radix–50 (%ccc) or ASCII ('c or "cc" characters, or arithmetic expressions using +, –, /, *, and < .)
DLG	DEV /LOG	Displays information about all the logged-in terminals on the system.
DLN	NCP SHOW KNOWN NODES	Displays all known DECnet nodes.
FRE	PIP /FR	Displays the amount of available space on SY:, the largest contiguous space on SY:, the number of available file headers, and the number of file headers used.

Table 19–1 (Cont.): TDX Command Summary

TDX Command	MCR Translation	Command Meaning
FRE ddu:	PIP ddu:/FR	Displays the same information as FRE for a specified device.
SHQ	QUE/LI	Displays information about all entries in all print queues.
SYS	SET /SYSUIC	Displays the current system UIC.
TDX	None	Displays the current version of TDX.

19.2 Understanding How TDX Works

TDX works in three different ways, depending on how you make logical assignments for the pseudo device ZZ. In each case, if you type a DCL command, the system executes the command normally.

The subsections that follow describe how the three different assignments affect TDX.

19.2.1 No ZZ Pseudo Device Assigned

When you receive your system, there are no logical assignments for the ZZ pseudo device. If you type a command that is not part of DCL, the system processes the command as follows:

1. The system passes the command to the MCR commmand dispatcher.
2. If the unknown command is three characters long, the system looks for an installed task of that name and runs it.

   For example, if you type XYZ, DCL passes the command to the command dispatcher (MCR). MCR looks for an installed task named ...XYZ and runs it.
3. If the unknown command is not the name of an installed task, or if it is longer than three characters, the command fails and the system prints an error message.

Thus, one way to add a custom command is to write a task that does whatever you want it to do. Then, install the task with a name of the form ...abc, where abc is the 3-character string you want to use as a custom command name. For more information, see the description of the INSTALL command in the *RSX-11M-PLUS MCR Operations Manual,* the *RSX-11M-PLUS Command Language Manual,* or the *Micro/RSX User's Guide, Volume 1.*

19.2.2 Pseudo Device ZZ1 Assigned

If you assign the pseudo device ZZ1 to the system disk, you activate the *flying install* behavior of the catchall task. You can do the assignment by typing the following command line:

```
$
```

If ZZ1 is assigned and you type a command that is not part of DCL, the system processes the command as follows:

1. The system passes the command to the MCR commmand dispatcher.
2. If the unknown command is three characters long, the system looks for an installed task of that name and runs it if it finds it.

   For example, if you type XYZ, DCL passes the command to the command dispatcher (MCR). MCR looks for an installed task named ...XYZ and runs it.
3. If the unknown command is not the name of an installed task, or if it is longer than three characters, TDX issues a RUN command. TDX uses the unknown command as the task name.

   For example, if the command is SHRINK, the catchall task issues a RUN $SHRINK command. The dollar sign ($) tells the system to search the system directory (usually [1,54]), then the system library directory (usually [3,54]), for a task image file named SHRINK.TSK. If the task exists in either of the directories, the catchall task runs it using the name SHRTn, where n is the number of the terminal from which the command was issued. The task is removed after execution completes.
4. If the RUN $ command fails, the system prints an error message.

   (For information on using the RUN command with the dollar sign, see the *RSX-11M-PLUS MCR Operations Manual*, the *RSX-11M-PLUS Command Language Manual*, or the *Micro/RSX User's Guide, Volume 1*.)

Use the flying install behavior to run tasks that typically are not used enough to warrant installing them permanently. You should not do this with tasks that require a larger increment than the default.

19.2.3 Pseudo Device ZZ2 Assigned

If you assign the pseudo device ZZ2 to the system disk, you activate the *flying Indirect* behavior of the catchall task. You can do the assignment by typing the following command line:

```
$
```

If ZZ2 is assigned, and you type a command that is not part of DCL, the system processes the command as follows:

1. It passes the command to the MCR commmand dispatcher.
2. If the unknown command is three characters long, the system looks for an installed task of that name and runs it if it finds it.

   For example, if you type XYZ, DCL passes the command to the command dispatcher (MCR). MCR looks for an installed task named ...XYZ and runs it.

3. If the unknown command is not the name of an installed task, or if it is longer than three characters, TDX tries to invoke an indirect command file. TDX uses the unknown command as the command file name, and TDX looks for the file in your login directory.

   For example, if you type in XYZ and your login directory is [303,5], TDX looks for a directory called 303005.DIR and looks in that directory for a file named XYZ.CMD. If such a file exists, then TDX invokes it.

   If the unknown command was more than three characters long, TDX uses only the first three characters; the rest are ignored.

   ### Note

   Because your login UIC does not change when you issue a SET DEFAULT or SET UIC command, your default directory or UIC does not affect TDX.

4. If there is no directory named for your login UIC, or if the directory does not contain the command file, TDX looks for the file in the system library directory. If the file exists, TDX invokes it.

5. If the library directory does not contain the command file, TDX looks in your login directory for a command file named CATCHALL.CMD. If the file exists, TDX assigns the value XYZ to symbol P1 and invokes CATCHALL.CMD. P1 is passed to the command file for further processing.

   For an example of a CATCHALL.CMD file, see Example 19–1. For more information on using symbols (such as P1) with Indirect, see the *RSX-11M-PLUS Indirect Command Processor Manual* or the *Micro/RSX User's Guide, Volume 1*.

6. If there is no such command file in your login directory, the catchall task looks for CATCHALL.CMD in the system library directory. If the file exists, the catchall task assigns the value XYZ to symbol P1 and invokes CATCHALL.CMD. P1 is passed to the command file for further processing.

You can use this behavior of the catchall task not only for adding commands to the system but also for controlling which users can issue which commands. Also, the path from directory to directory and command file to command file is consistent, so the system manager can insert the catchall behavior at a number of points and on a user-by-user basis.

Note

Independent of the behavior of the catchall task, the Indirect Command Processor (Indirect) checks the library directory after checking the default directory for a command file invoked in the usual way (that is, with an at sign (@)). See the discussion of Indirect in the *RSX-11M-PLUS Indirect Command Processor Manual* or the *Micro/RSX User's Guide, Volume 1* for more information.

Example 19–1 is a sample command file for use with TDX. Experiment with it in a directory named for your login UIC before you put it in the library directory.

Example 19-1: A Sample CATCHALL.CMD File

```
                   .;
                   .; This indirect command file demonstrates one way
                   .; of using a catchall task.  The catchall task is
                   .; installed as ...CA. and is in the system
                   .; library directory in the task image file
                   .; TDX.TSK.
                   .;
           .ENABLE SUBSTITUTION
                   .;
                   .; You must always enable SUBSTITUTION when the
                   .; command file uses changeable symbols such as P1
                   .; or symbols defined with .ASK directives.
                   .;
           .ENABLE QUIET
                   .;
                   .; When QUIET is enabled, Indirect executes commands
                   .; without displaying them on the terminal.
                   .;
                   .; The following list of possible values of P1 is, in
                   .; effect, a table of pseudo commands that this file can
                   .; execute.  The labels (following the GOTOs) indicate the
                   .; part of the file where the action to be taken is
                   .; given.  If you type in FLY, the GOTO goes to the
                   .; label .FLY:, where the command TYPE DUO:[USER]FLY.TXT
                   .; is issued.  Note the use of full filespecs.  TDX,
                   .; like most system features, will use the defaults of
                   .; the user who issued the command if no other device
                   .; or directory information is provided in filespecs.
                   .;
                   .; If the pseudo command is not in the table, Indirect
                   .; reaches the comment that prints Foo on the screen
                   .; and then exits.
                   .;
                   .; Using a combination of the capacities of DCL and of
                   .; Indirect, you can implement pseudo commands to do
                   .; whatever you want.
                   .;
          .IF P1 EQ "FLY" .GOTO FLY
          .IF P1 EQ "FLU" .GOTO FLU
          .IF P1 EQ "MYD" .GOTO MYD
;
;
;          Foo
;
;
           .GOTO OVER
.FLU:      TYPE DUO:[USER]FLU.TXT
           .GOTO OVER
.FLY:      TYPE DUO:[USER]FLY.TXT
           .GOTO OVER
.MYD:     @DUO:[USER]MYDISK.CMD
           .GOTO OVER
```

(Continued on next page)

Example 19–1 (Cont.): A Sample CATCHALL.CMD File

```
                  .;
                  .; The .DISABLE DISPLAY prevents Indirect from
                  .; printing the @<EOF> that it ordinarily prints when
                  .; it has finished executing a command file.
                  .;
.OVER:      .DISABLE DISPLAY
.EXIT
```

19.3 Selecting MCR Options for TDX

You may want to add other commands to TDX by modifying the source file, [24,10]TDX.MAC. Routines for the commands are at the end of the source file. To reassemble the source file after making additions, use the TDXASM.CMD file located in directory [24,24].

If the command you enter does not match a TDX command, TDX attempts to use one of the following MCR command options (in the following descriptions, the letters xxx represent the first three characters of your command):

- The first option installs, runs, and then removes a task, as follows:

  ```
  MCR>
  ```

  "Params . . . " represents the command line that you enter. Your command line cannot exceed 39 characters because TDX adds other characters to this line. To select this option, place the following command in your LOGIN.CMD file:

- The second option searches for an indirect command file. You can specify this option with one of the following four command lines:

  ```
  MCR>
  MCR>
  MCR>
  MCR>
  ```

  TDX searches for one of the @[. . .]xxx.CMD indirect command files in the order indicated. Once TDX locates one of these files, the search stops. You can design the indirect command file to perform a variety of functions, such as installing tasks, providing HELP, or issuing error messages.

  To use this option, place the following command in your LOGIN.CMD file:

The option used by TDX depends on whether the logical devices ZZ1 or ZZ2 exist on your system. You can assign one of these devices in your LOGIN.CMD file. Note that the presence of the ZZ1 assignment masks the presence of the ZZ2 assignment. If no assignment is made, TDX will not exercise either option.

You can also select an option by placing the following command in the system startup file:

The number n equals 1 or 2, depending upon which ZZn option you select.

If the command you enter does not match a TDX command or if TDX cannot locate the necessary task, TDX issues the following message:

```
MCR -- Task not in system
```

19.4 Installing Tasks Permanently

Instead of invoking TDX to install tasks as you need them, you can install some tasks permanently. You should install the following types of tasks permanently:

- Tasks requiring a larger increment than the default
- Tasks used frequently
- Tasks requiring commands longer than 39 characters

For more information, see the description of the INSTALL command in the *RSX-11M-PLUS MCR Operations Manual,* the *RSX-11M-PLUS Command Language Manual,* or the *Micro/RSX User's Guide, Volume 1.*

Appendix A
RSX-11M-PLUS UFD Conventions

The RSX-11M-PLUS and Micro/RSX operating systems observe a set of User File Directory (UFD) conventions for files on disk. These conventions provide a consistent method of locating, allocating, and maintaining a file.

A UFD is itself a file, named in the format gggmmm.DIR where ggg and mmm are octal numbers in the range 1_8 to 377_8 (0 is reserved). The numbers represent the file owner's group and member number respectively. For more information on file ownership and directories, see the *RSX-11M-PLUS MCR Operations Manual*, the *RSX-11M-PLUS Command Language Manual*, or the *Micro/RSX User's Guide, Volume 1.*

The following sections describe group and member numbers and their use in identifying the contents of system UFDs. A section on file naming conventions is also included.

A.1 Group Number

The group number identifies groups of directories. By convention, group numbers 1_8 to 200_8 are reserved by DIGITAL for system files, and group numbers 201_8 to 377_8 are reserved for user files. Table A-1 lists the group numbers and their usage.

Table A-1: Group Number Identification

Group Number	Usage
11	Executive files
12	Monitor Console Routine (MCR) files
13	Files-11 Ancillary Control Processor (FCP) files
14	Resource Monitoring Display (RMD) files
15	Executive utility tasks (EUT)
16	Multiuser utility tasks
20	File system utility tasks (such as MOU, DMO, INI, or UFD)

Table A-1 (Cont.): Group Number Identification

Group Number	Usage
23	DIGITAL Command Language (DCL) files
24	Catchall utility task (TDX)
25	Queue Manager (QMG) files
27	Online configuration
45	K-series files
50	File Control Services (FCS) files
75	Replacement Control Task (RCT)
77	Indirect Command Processor (ICP) Files
104	Error Logging Control files
121	Line printer despooler
125	Shadow Recording
126	Resource Accounting tasks
200	System Generation indirect command files

A.2 Member Number

The second number (member number) identifies the different file types within each group. Table A-2 lists the member numbers.

Table A-2: Member Number Identification

Member Number	File Types
10	Source modules
24	Object modules, assembly and task-build command files, and object module libraries for the Executive and for privileged and nonprivileged tasks
34	System listings and map files for privileged and nonprivileged tasks
40	SLP correction files for files located under member number 10
54	System image and task image files
200	System Generation indirect command files

A.3 Example of System UFDs

The system UFDs consist of a group number and a member number. These numbers identify the specific contents of each directory. For example, to locate all the Monitor Console Routine (MCR) listing and map files, find the group number for MCR files (12) and the member number for listing and map files (34). Then, combine the group and member numbers to specify the

UFD [12,34], which contains the MCR listing and map files. Table A–3 lists the MCR UFDs and the files contained in each UFD. Table A–4 lists the UFDs used by the operating system.

Table A–3: MCR User File Directories (UFDs)

UFD	Usage
[12,10]	MCR source files
[12,24]	MCR object modules, assembly command files, and library build command file
[12,34]	MCR listing files from assembling source files in [12,10].
[12,40]	SLP correction files for files under [12,10]

Table A–4: System UFDS

UFD	Usage
[1,1]	System and macro object module libraries
[1,2]	System message and help files
[1,3]	Lost files found by the File Structure Verification Utility (VFY)
[1,4]	Postmortem and snapshot dumps
[1,6]	Error Logging and Resource Accounting files
[1,7]	Spooling queue file and transparent listing file
[1,11]	Work space for system maintenance
[1,24]	Object module libraries, task-build command files for mapped tasks, and overlay descriptor files for privileged tasks
[1,34]	Executive and task map files
[1,54]	Executive and system tasks
[200,1]	Sample files for system introduction
[200,2]	Field service files

A.4 System File-Naming Conventions

RSX–11M–PLUS and Micro/RSX operating systems observe certain conventions for naming files.

Format

```
nnnsss.typ
```

Fields

nnn

Specifies the system identification for the task. For example, DMP is the identification for the File Dump Utility (DMP) program.

sss

Specifies a suffix for noting the relationship of the file to the task specified in the nnn identification. For example, PIPBLD.CMD is the command file for building the Peripheral Interchange Program (PIP).

The following suffixes are valid:

ASM	Assembly command file
BLD	Task Builder (TKB) command or overlay descriptor file
RES	File Control Services (FCS) memory-resident version of the task
FSL	FCS supervisor-mode library version of the task

typ

Specifies the standard mnemonic for file type. RSX–11M–PLUS and Micro/RSX systems have a set of conventional and default file types that are used to reflect the contents of a file (see Table A–5).

Table A–5: System File Types

Type	File Contents
BAS	BASIC-11 language source file
BAT	Batch file (convention only)
BIC	Diagnostic binary file (chainable)
BIN	Diagnositc binary file
BLD	Indirect command file used as input to system generation procedure
B2S	BASIC-PLUS-2 language source file
CBL	COBOL language source file
CDA	Crash dump binary file
CFS	Error Logging control file string
CLB	Indirect Command Processor command file library
CMD	Indirect command file
CMF	Preprocessed indirect command file
COR	SLP correction file
CRF	Cross-Reference Processor (CRF) symbol table file
DAT	File containing data (as opposed to a program)
DIR	Directory file
DMP	File Dump Utility (DMP) output file
ERR	Error Logger output file

Table A-5 (Cont.): System File Types

Type	File Contents
FTN	FORTRAN IV or FORTRAN-77 language source file
HLP	Help file
JOU	EDT editor journal file
LOG	Batch or console log file
LST	Listing file
MAC	MACRO-11 source file
MAP	TKB memory allocation map
MLB	Macro library file
OBJ	Object module file (output from either the MACRO-11 assembler or a compiler)
ODL	TKB overlay descriptor file
OLB	Object module library file
PAS	Pascal-11 language source file
PAT	Object Module Patch Utility (PAT) correction file
PMD	Postmortem Dump (PMD) file
POB	Patched object module used by the PAT Utility
SML	System Macro Library file
STB	Task symbol table file
SYS	Bootable system image or other system file
TMP	Temporary file
TSK	Task image file
TXT	Text file
ULB	Universal library file

Index

RSX-11M-PLUS and Micro/RSX
System Management Guide
AA-JS14A-TC

READER'S COMMENTS

Your comments and suggestions are welcome and will help us in our continuous effort to improve the quality and usefulness of our documentation and software.

Remember, the system includes information that you read on your terminal: help files, error messages, prompts, and so on. Please let us know if you have comments about this information, too.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

____

____

____

____

____

____

Did you find errors in this manual? If so, specify the error and the page number.

____

____

____

____

____

____

What kind of user are you? ____ Programmer ____ Nonprogrammer

Years of experience as a computer programmer/user: ______________________

Name ______________________________ Date______________

Organization ______________________________________

Street ____

City ______________________________ State ________ Zip Code ________
or Country

--- Do Not Tear - Fold Here and Tape ---

digital™

No Postage
Necessary
if Mailed
in the
United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Corporate User Publications—Spit Brook
ZK01–3/J35
110 SPIT BROOK ROAD
NASHUA, NH 03062-9987

--- Do Not Tear - Fold Here ---

RSX-11M-PLUS and Micro/RSX
System Management Guide
AA–JS14A–TC

READER'S COMMENTS

Your comments and suggestions are welcome and will help us in our continuous effort to improve the quality and usefulness of our documentation and software.

Remember, the system includes information that you read on your terminal: help files, error messages, prompts, and so on. Please let us know if you have comments about this information, too.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

____
____
____
____
____
____

Did you find errors in this manual? If so, specify the error and the page number.

____
____
____
____
____
____

What kind of user are you? ____ Programmer ____ Nonprogrammer

Years of experience as a computer programmer/user: ______________

Name ______________________ Date______

Organization ______________________

Street ______________________

City______________________ State or Country ________ Zip Code________

-- Do Not Tear - Fold Here and Tape

digital™

No Postage
Necessary
if Mailed
in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Corporate User Publications—Spit Brook
ZK01-3/J35
110 SPIT BROOK ROAD
NASHUA, NH 03062-9987

-- Do Not Tear - Fold Here

digital™