RSX-11M
Operator's Procedures Manual

Order No. DEC-11-OMOGA-B-D

RSX-11M

Operator's Procedures Manual

Order No. DEC-11-OMOGA-B-D

RSX-11M Version 2

digital equipment corporation · maynard. massachusetts

First Printing, November 1974
Revised, September 1975

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

The following are trademarks of Digital Equipment Corporation:

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-11
DECCOMM		

LIMITED RIGHTS LEGEND

Contract No. _____

Contractor or Subcontractor: Digital Equipment Corporation

All the material contained herein is considered limited rights data under such contract.

CONTENTS

CONTENTS (Cont.)

FIGURES

TABLES

PREFACE

0.1 MANUAL OBJECTIVES AND READER ASSUMPTIONS

The intent of this manual is to enable its user to successfully operate an RSX-11M system. It is designed to be self-contained. However, if the operator wishes to have deeper insight into system construction and the precise meaning of terminology used throughout this manual, a reading of the RSX-11M System Generation Manual is strongly recommended.

Although the introduction to this manual has a tutorial slant, it does not claim to train operators. The reader is assumed to be familiar with computer operating procedures, both realtime and batch, to have a basic vocabulary of computer-related terms, and to have had experience operating both the computer console and the terminal devices supported by RSX-11M for use as operator consoles.

Finally, the term "operator" includes not only the conventional understanding of that term, but also anyone who chooses to interface directly with RSX-11M to accomplish some task. The expanded definition of an operator is necessitated by the varied settings in which an RSX-11M system will operate; some of those settings will not have an operator in the conventional sense.

This manual is intended mainly for RSX-11M users; however; it can also be used as a reference document for RSX-11S operation; since RSX-11S is subset-compatible with RSX-11M.

0.2 STRUCTURE OF THE DOCUMENT

Chapter 1 is a capsule overview of the RSX-11M system, focusing on those aspects of the system related to the operator interface. It also contains the system-unique terminology used throughout the manual.

Chapter 2 discusses conventions for terminal operations and device names.

Chapter 3 discusses the file system and file specifiers.

Chapter 4 describes the operator commands.

Chapter 5 describes the use of indirect files and the Indirect File Processor.

Chapter 6 contains the error messages produced by the Monitor Console Routines (MCR).

Appendix A contains all RSX-11M MCR commands in summary form.

Appendix B contains a summary of MCR commands and error messages for RSX-11S. RSX-11S MCR is a proper subset of RSX-11M MCR; thus, RSX-11S MCR differs only in the subset commands it will accept.

0.3 ASSOCIATED DOCUMENTS

Other manuals closely allied to the purposes of this document are described briefly in the RSX-11M/RSX-11S Documentation Directory, Order No. DEC-11-OMUGA-B-D. The Documentation Directory defines the intended readership of each manual in the RSX-11M/RSX-11S set and provides a brief synopsis of each manual's contents.

CHAPTER 1

INTRODUCTION

RSX-11M is a multiprogramming, realtime operating system. Its fundamental function is to provide the control for sharing system resources among any number of user-prepared tasks. These tasks are written in a supported source language: i.e., MACRO-11, FORTRAN IV, or FORTRAN IV-PLUS. The source language is subsequently translated, linked to form a task image, and stored on a file-structured volume in a named file that is accessible by the file system.

Tasks stored on a file-structured volume may be installed into an RSX-11M system and subsequently run by issuing a command to the Monitor Console Routine (MCR). The terms "install" and "run" have precise meanings which will be discussed shortly. MCR provides the language interface between the operator and the system.

A task runs in a partition. A partition is a contiguous block of memory having the following attributes:

1. A name;
2. A defined size;
3. A fixed starting address, and
4. A defined type.

There are two versions of RSX-11M: mapped and unmapped. From the operator's point of view, differences exist between the two versions only with respect to installing tasks into a partition.

In unmapped systems, a task is linked, installed, and run in a partition having a specific base address. Such a task cannot be run in any partition whose base is not identical. In mapped systems, a task may be installed into any partition large enough to contain it, but it can only run in the partition into which it was installed. In both systems, if the task is checkpointable (i.e., may be temporarily suspended to make the CPU available to higher priority tasks, its checkpoint area must be large enough to contain the partition into which it was installed.

RSX-11M supports two types of task partitions: system-controlled and user-controlled. Space in a system-controlled partition (mapped systems only) is dynamically allocated by the system to as many tasks as will fit simultaneously in the partition. A user-controlled partition (both mapped and unmapped systems) is allocated to only one task at a time.

In addition, a user-controlled partition, which is a main partition, can be subdivided into as many as seven non-overlapping subpartitions. The subpartitions occupy the identical physical memory as the main

partition. However, tasks may not be resident in a main partition and its subpartitions simultaneously. If a main partition is occupied, the subpartitions cannot also be occupied. All subpartitions can have tasks residing in them; therefore, up to seven potentially parallel task executions can exist within a pre-empted main partition.

The goal of subpartitioning is to reclaim large storage areas in unmapped systems. For example, when a large task requiring a main partition is either no longer active or can be checkpointed, subpartitioning can be employed to make room for a number of smaller realtime tasks.

Tasks running in partitions compete for system resources based on their priority and resource availability. The highest priority task which has all the resources it needs to run, and which can make use of the resources it needs, will be in control of the CPU.

As tasks request system services, they become blocked, usually waiting for an I/O transfer to complete. During this blockage, RSX-11M looks for another task to run. Running a task in the blocked intervals of other tasks is termed multiprogramming.

Tasks in all partitions compete on the basis of priority for the CPU. Thus, the greater the number of user-controlled partitions (including main partition and subpartitions), and the larger the size of the dynamically allocted, system-controlled partitions, the greater the possibilities are for competition.

In addition to competition between memory resident tasks, another level of competition is introduced by the checkpointing facility. A task currently running may, if declared checkpointable, be pre-empted from memory, moved to mass storage, and replaced by a higher priority task installed to run in the same partition. Later, when the higher priority task has finished, the previously swapped-out task is returned to memory and continues operation from the point of interruption. This roll-in, roll-out process is termed checkpointing. These fundamental concepts are illustrated in Figure 1-1.

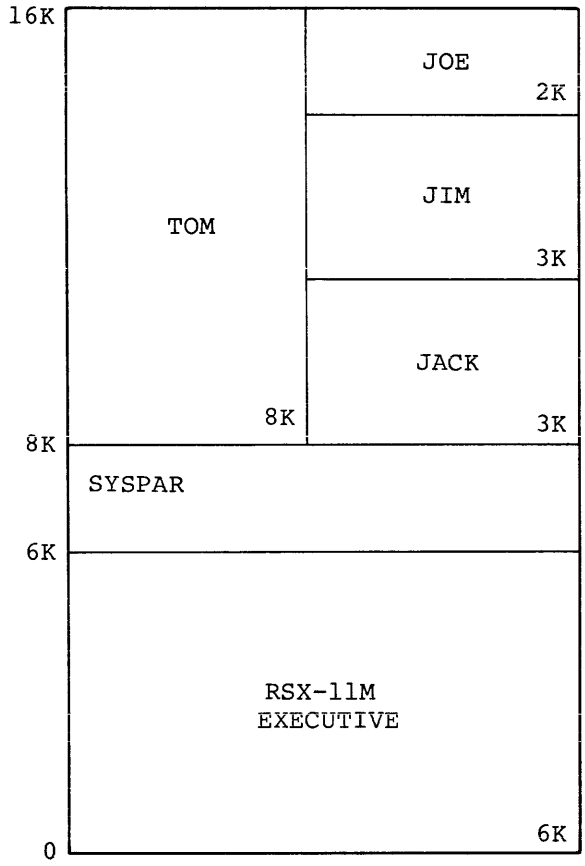


Figure 1-1
Sample System Memory Layout

Suppose a 16K unmapped system exists with the memory layout shown in Figure 1-1. The Executive region, requiring 8K, consists of the Executive and the user-controlled main partition SYSPAR. This partition contains the file system, MCR, and the Task Termination Notification routine (TKTN). Furthermore, the file system is checkpointable and of lower priority than MCR or TKTN. Thus, if the file system is running and an operator requests MCR, the File System will be checkpointed, and MCR will be loaded and initiated.

The user area is composed of a user-controlled main partition named TOM, 8K in length, and three subpartitions, named JOE, JIM, and JACK. The 8K partition is used for program preparation tasks such as the language processors and the Task Builder. These programs will usually have a low priority and may be checkpointable.

The three subpartitions are for realtime tasks. Any time one of these tasks needs the processor and it has a higher priority than the task occupying the main partition, the contents of the main partition will be checkpointed (if the occupant task is checkpointable).

Suppose the partitions JOE, JIM, JACK, and SYSPAR are occupied and the tasks in them are ready to run. To which task does the Executive give the processor? As stated earlier, the task selection process is

entirely dependent on task priority; this holds true for SYSPAR as well. Hence, the highest priority task will run next, regardless of where it resides in memory.

Earlier, the two terms

install, and

run

were said to have precise meanings. These meanings will be defined while sketching the creation of a task. Assume that an operational RSX-11M system exists.

An RSX-11M task comes into existence through four basic steps:

1. A program is written in a supported source language (MACRO-11, FORTRAN IV, or FORTRAN IV-PLUS) and stored on a file-structured volume via a source language processor.

2. The source code is submitted to the applicable translator, and an object file is built on a file-structured volume.

3. The object file is submitted to the Task Builder, and the output, known as the task-image file, is also stored on a file-structured volume.

4. Finally, via an MCR command, the task is installed into the operational system.

Installation involves recording a number of task parameters in a system-resident table called the system task directory (STD). The recorded task parameters include its name, length, and volume address. An installed task is one which has an entry in the STD. Such a task is not in memory, nor is it in competition for CPU resources. It is simply known to the system and is considered dormant until activated.

A dormant task may be activated via the MCR Run command, or it may be activated internally by another active task which issues a RQST$ (Request) or RUN$ (RUN) Executive directive. When such a request is received, a sequence of events is initiated that allocates resources, brings the task into memory, and places it in active competition with other tasks. Thus, an installed task is one that is known to the system, which is dormant, since it is not eligible to compete for resources. On the other hand, an active task is also known to the system, but it is eligible to compete for system resources.

Note that the number of installed tasks can, and usually will, far exceed the number of active tasks.

The concept of dormant versus active is much more important in a realtime system than in a batch-oriented system. A dormant task uses very little memory; yet, when the task is needed to service a realtime event, it can quickly and efficiently be introduced into active competition for system resources, since most of the parameters describing the task, including its volume address, are already resident in main memory.

1.1 SUMMARY

Successful operation of an RSX-11M system is enhanced by the operator's understanding of one of RSX-11M's structures and three of its facilities:

The structure is:

Partitions (both user-controlled and system-controlled), and

the facilities are:

Multiprogramming;

Checkpointing, and

Task creation, installation, and activation.

By understanding these terms, the operator can comprehend the dynamic behavior exhibited by RSX-11M, especially in larger configurations which exploit its potential.

CHAPTER 2

PROCEDURES AND CONVENTIONS

2.1 TERMINAL OPERATIONS

Effective terminal interaction with RSX-11M requires a complete understanding of the following:

1. Terminal operations;
2. Peripheral device naming and assignment;
3. File naming and file specifiers, and
4. Indirect file construction (see Chapter 5).

The RSX-11M operator performs a role in two major areas:

1. The control of on-line processes, and
2. Emergency on-line system alteration.

Operator functions fall into four classes:

1. Initialization Commands

   Initialization includes:

   Entering date and/or time;
   Installing tasks;
   Preparing mass storage media for file processing;
   Memory re-partitioning, and
   Setting terminal:
   - types;
   - buffer sizes;
   - slave status, and
   - privilege.

2. Informational Commands

   These functions provide the operator with a description of the structure of the running system. The operator can request:

   Time and date;
   Task names;
   Partition definitions;
   Symbolic device names and status, and
   Logical device assignments.

3. Control Commands

   Control functions include:

   Task execution;
   Task resumption;
   Task abort;
   Fix and Unfix tasks in memory, and
   Logical device assignments.

4. System Maintenance Commands

   During on-line operations, an operator or software diagnostic engineer can save the current state of the system for subsequent restoration (via the Save command); also, any word in the system can be examined or altered, using the Open Register command. If the Executive Debugging Tool (XDT) has been generated into the system, control can be passed to it by use of the Breakpoint command.

2.1.1 Special and Control Characters

Special characters are used to control an operator terminal. Control characters are typed while the control key (CTRL) and one other key are simultaneously depressed. The two control characters CTRL/Z and CTRL/U are echoed as ^Z and ^U. Other CTRL characters, although echoed, are non-printing characters and, therefore, do not appear on the display medium.

The characters used in terminal operations are described in Table 2-1.

Table 2-1
Characters Used in Terminal Operations

SYMBOL	EXPLANATION
CTRL/C	Typing CTRL/C causes the MCR recognition sequence discussed below to be initiated. ^C is not echoed.
CTRL/I	Horizontal tab causes spacing to the next tab stop. These stops are set by the software following every eighth character position. ^I is not echoed.
CTRL/K	Typing CTRL/K causes a vertical tab (appears on the terminal as four line feeds). ^K is not echoed.
CTRL/L	Typing CTRL/L causes a form feed. Paging is not done; eight line feeds are placed on the display medium. ^L is not echoed.
CTRL/O	CTRL/O is used to suppress the display of unwanted output on a terminal. The subsequent effect of entering CTRL/O depends on the state of the terminal when the character is entered. ^O is not echoed. CTRL/O is controlled internally by a single bit called the Disable-output bit. When this bit is set, output to the terminal is disabled; when cleared, output can proceed. Initially, the bit is cleared. When CTRL/O is entered, the bit is complemented. Thus, a typical response of entering successive CTRL/O characters is to stop and start output to the terminal. Characters that would have normally been displayed during the stopped interval are discarded. If more than one task is sending output to a terminal, it is essential that CTRL/O affect only the task delivering output at the time CTRL/O is entered; control is implemented as follows: The disable bit is cleared for attached terminals at: The issuance of an Attach QIO directive; The issuance of a Detach QIO directive; The solicitation of input, and The arrival of unsolicited input. For unattached terminals, the bit is also cleared each time an output request directed to the terminal is initiated.

Table 2-1 (Cont.)
Characters Used in Terminal Operations

SYMBOL	EXPLANATION
CTRL/O (Cont.)	Given these conditions for clearing the disable bit, and the fact that entering CTRL/O always complements the bit, the following applies at the entering terminal: For an attached terminal, output can be stopped by entering CTRL/O. The interrupted output stream will be discarded until the next: Detach; Solicited input; Unsolicited input, or Another CTRL/O. For unattached terminals, entering CTRL/O stops output, but remains in effect for only the current buffer of output, since, at the next I/O initiation, the system always clears the Disable-Output bit for unattached terminals.
CTRL/Q	Typing CTRL/Q after typing CTRL/S will resume output suspended by the previous CTRL/S. ^Q is not echoed.
CTRL/S	Typing CTRL/S during output suspends additional output until CTRL/Q is typed. The functions of CTRL/S and CTRL/Q are convenient when using a display terminal which transmits at rates above 600 baud. ^S is not echoed.
CTRL/U	Typing CTRL/U prior to typing a line terminator causes the previously typed characters to be deleted back to the default prompting symbol ">". The system responds with a carriage return and line-feed so that the line can be retyped. Echoed as ^U.
CTRL/Z	A break character indicating end-of-file. It is used as a signal to system tasks, such as MACRO-11, PIP, and TKB, indicating that the user is finished and the subject task may exit. Echoed as ^Z.
ESC	The ESC character represents the ESCape or ALTMODE key. Entering this line terminator leaves the carriage (or cursor) of the terminal intact. It has a specialized meaning to MCR when terminating a command. It will suppress the default prompt from MCR commands when they complete. However, an ESCAPE character terminating a Run command produces special results (see the description of the Run command for a detailed explanation). ESC is not echoed.

Table 2-1 (Cont.)
Characters Used in Terminal Operations

SYMBOL	EXPLANATION
CR	Typing the RETURN key will end line input and cause the carriage (or cursor) to return to carriage position 1.
\	The back slash symbol is used to delimit characters deleted by typing the RUBOUT key. Typing a RUBOUT deletes the last character typed. Several contiguous characters can be deleted by typing successive RUBOUTs.

2.2 TERMINAL CHARACTERISTICS

There are two types of MCR commands: privileged and non-privileged. Non-privileged commands may be invoked at any terminal, whereas privileged commands may be invoked only at privileged terminals. The privileged characteristic is associated with a given terminal at system generation; subsequently, any privileged terminal can, via the MCR Set command, establish any other terminal as privileged or non-privileged.

CAUTION

Privileged commands can destructively interfere with system operation and with themselves. Caution is urged when more than one privileged terminal is present on the system.

Non-privileged commands are available to any unattached* terminal in the system. More than one terminal may be in use at a time, and all MCR non-privileged commands are available to each terminal. A terminal may be attached to a task other than MCR, in which case, all input or output is directed to or from that task. Thus, one person can run PIP at one terminal, while another runs MACRO-11 at another terminal; still a third may be controlling the system using MCR commands. This capability is restricted only by memory availability for the various tasks.

Commands are serviced by either:

1. Independent tasks (such as Mount and Install), or
2. The MCR Dispatcher and a set of overlays.

* Attached terminals are those dedicated to a single task. Only the task to which it is dedicated can perform I/O to such a terminal.

Whenever a line is entered from an unattached terminal, the MCR Dispatcher reads the command line. It then takes the first three characters of the line (these are unique and form a part of the command) and searches a table of commands processed by overlays. If the MCR Dispatcher finds the command in this table, it loads the overlay to process the command. If it does not find the name, it prefixes

... (three periods)

to the command and searches the system task directory (STD) for a task by that name (e.g., ...MAC), as described in the following paragraphs.

If the MCR Dispatcher finds the task and it is not active, the Dispatcher will request the Executive to run the task. If the task is active or not in the system, the Dispatcher will search for the task, as follows:

<command-name>n

where n is the terminal unit number from which the command was issued (e.g., MAC1). If the Dispatcher finds the task and it is not already active, the Dispatcher will request that the task be run. If the Dispatcher does not find task <command-name>n, but task ...<command-name> is installed in a system-controlled partition and is not checkpointable, the Dispatcher will create a task by the name of <command-name>n and request it to be run. This task will be identical (uses the same task-image file) to the parent task ...<command-name>. This feature is extremely useful in multi-terminal systems used for program development.

At the time the Dispatcher makes the request, it also sends the complete command to the task it just requested. This scheme makes it possible for users to add (or delete) operator services to meet their application needs.

For example, an installation could write a routine which displays the day of the year using the internally stored current date. If the task is called DAY, the user would build and install it as:

...DAY

Then the user-issued command:

>DAY

will result in task initiation.

2.3 TERMINAL INPUT CONVENTIONS

An unattached operator terminal waiting for input is in one of three states:

1. Ready to accept unsolicited input;
2. Ready to accept solicited input to MCR; or
3. Ready to accept solicited input to a task other than MCR.

A terminal ready to accept unsolicited input displays the character:

```
>
```

on the terminal. This single-character display on a terminal is referred to as the default prompt. Input entered following a default prompt is always implicitly directed to MCR, and the MCR Dispatcher identifies, analyzes, and responds to subsquent input.

Whenever CTRL/C is entered as the first character on a terminal awaiting input, MCR is explicitly initiated, and the prompt:

```
MCR>
```

appears on the display medium. Any input entered following the

```
MCR>
```

prompt is explicitly directed to MCR. In the case of the default prompt, however, another task could internally solicit input; if the soliciting task did not itself issue a prompt, then input (though thought to be directed to MCR) would in fact be sent to the task most recently soliciting input. The default prompt merely indicates that MCR will field unsolicited input.

To avoid erroneous input, any task intending to solicit input should:

1. Attach to the terminal;
2. Prompt with a properly formatted identifier (i.e., tsk>), and then
3. Solicit input.

Once a task is initiated by MCR, it will generally prompt with:

```
tsk>
```

where tsk is a 3-character task name. All DEC system tasks are identified with a 3-character prompt; it is advisable that user programs follow this convention. When a task prompt has been issued and is then immediately followed by a read, the next line of input is directed to the soliciting task.

When a task is sending characters to a terminal without attaching to it, the output can be interrupted by entering any character. At the completion of the current I/O operation, the character will be echoed, and the operator can issue a command to MCR.

For attached terminals sending data, a similar interruption can be achieved by entering CTRL/C. Then, as with unattached terminals, at the completion of the current I/O operation:

```
MCR>
```

will be displayed.

These conventions enable tasks delivering output to be interrupted conveniently.

MCR commands are usually initiated as unsolicited input and are terminated by typing a carriage return; the Open command is an exception, because it is terminated by typing the ESCape key. If any other command to MCR is terminated with an ESCape character, MCR will not respond with the default prompt. Typing CTRL/C as the first character in a line, or anytime when a terminal is sending data, will always obtain MCR's attention, even for attached terminals. Unsolicited input, except CTRL/C, will be rejected if the terminal is attached. All examples in this manual use the default prompt.

If a task prompt is awaiting input, the sequence:

CTRL/C (Carriage Return)

will obtain MCR's attention. Following a single input line to MCR, the original task whose prompt was overridden is again given control of the terminal.

2.4 START-UP PROCEDURES

After performing the applicable start-up procedure (see following sections), the RSX-11M system message is then displayed at the computer console terminal. At this point, the operator should enter the time and the date, and establish his user identification code (UIC) via the MCR Set command. The step-by-step start-up procedure is carried out as described in the following sections.

2.4.1 Start-Up Using BM792-YB Bootstrap ROM

The following procedures are used in starting up the BM792-YB bootstrap ROM:

1. Depress HALT.
2. Set the console switches to 173100.
3. Depress LOAD ADDRESS.
4. Enter the address of the load device in the console switches:

RK11 Disk	177406
RF11 Disk	177462
RP11 Disk	176716
TC11 DECtape	177344

5. Lift up the HALT key.
6. Press START.

The start-up sequence in section 2.4.6 is then displayed on the console terminal.

2.4.2 Start-Up Using MR11-DB Bootstrap ROM

Use the following procedures in starting up the MR11-DB bootstrap ROM:

1. Depress HALT.
2. Enter the address of the load device in the console switches:

RF11 Disk	173100
RK11 Disk	173110
TC11 DECtape	173120
TM11 Magtape	173136
RP11 Disk	173154

3. Depress LOAD ADDRESS.
4. Lift up the HALT key.
5. Press START.

The start-up sequence in section 2.4.6 is then displayed on the console terminal.

2.4.3 Start-Up Using BM873-YA Bootstrap ROM

The following procedures are used in starting up the BM873-YA bootstrap ROM:

1. Depress HALT.
2. Enter the address of the load device in the console switches:

RF11/RH70 Disk	173000
RK11/RH70 Disk	173010
TC11 DECtape	173030
TM11 Magtape	173050
RP11 Disk	173100
RC11 Disk	173144
KL11/DL11	173210
TA11 Cassette	173230
PC11 Papertape	173312

3. Depress LOAD ADDRESS.
4. Lift up the HALT key.
5. Press START.

The start-up seqence in section 2.4.6 is then displayed on the console terminal.

2.4.4 Start-Up Using BM873-YB Bootstrap ROM

To start up the BM873-YB bootstrap ROM, use the following procedures:

1. Depress HALT.
2. Enter the address of the load device in the console switches:

RH11/RS03/RS04 Disk (unit 0)	173000
RH11/RS03/RS04 Disk (unit in switch register)	173002
RK11 Disk (unit 0)	173030
RK11 Disk (unit in switch register)	173032
TC11 DECtape (unit 0)	173070
TM11 Magtape (unit 0)	173110
RF11 Disk	173136
RH11/RH70/TM02/TU16 Magtape (unit 0)	173150
RC11 Disk	173212
RH11/RH70 Device Combination (unit zero)	173230
RH11/RH70 Device Combination (unit in switch register)	173232
RH11/RH70/RP04 Disk (unit 0)	173320
RH11/RH70/RP04 Disk (unit in switch register)	173322
RP11/RP02/RP03 Disk (unit 0)	173350
RP11/RP02/RP03 Disk (unit in switch register)	173352
KL11/DL11 Console Terminal Reader	173510
TA11 Cassette (unit zero)	173524
TA11 Cassette (unit in switch register)	173526
PC11 Papertape Reader	173620

3. Depress LOAD ADDRESS.

3a. For units other than zero, enter unit number in switch register.

4. Lift up the HALT key.

5. Press START.

The start-up sequence in section 2.4.6 is then displayed on the console terminal.

2.4.5 Start-Up Using M9301-YA or M9301-YB Bootstrap ROM

The M9301-YA or M9301-YB bootstrap ROM is started up using the following procedures:

1. Press the BOOT Switch.

2. In response to the "$" prompt on the console terminal, enter one of the following:

```
DB - RJP04 Disk                 M9301-YB only
DK - RK11 Disk
DP - RP11 (RP02/03) Disk
DS - RS03/04 Disk               M9301-YB only
DT - TC11 Dectape
DX - RX11 Floppy Disk
CT - TA11 Cassette Tape
MC - Mixed Massbus Device       M9301-YB only
MM - TM02/TU16 Magtape          M9301-YB only
MT - TM11/TU10 Magtape
PR - PC11/PR11 Paper Tape Reader
TT - KL11/DL11 Console Terminal Reader
```

   The two alphabetic characters may be followed by an optional single octal digit for bootstrapping from units other than zero and terminated by a carriage return.

The start-up sequence in section 2.4.6 is then displayed on the console terminal.

2.4.6 Start-Up Display Sequence

If the system being bootstrapped was built with a different device configuration than the current host, a series of display lines appears on the console in the form:

```
DEVICE ddnn:  NOT IN CONFIGURATION*
```

where:

dd = device name.
nn = unit number.

---

* Devices not in the configuration are declared off-line and the display OFFLINE is appended to these devices as part of the output of the Devices command.

Then the system displays:

```
   RSX-11M <version-number> <base level> <memory size> [MAPPED]
>RED ddnn:=SY0:
>MOU ddnn:
>@[1,2]STARTUP
```

where:

<version number>	- the release version number, e.g., Release 2 of RSX-11M is V02.
<base level>	- The software development group at Digital periodically brings together new versions of the software under development. Internal groups may use this software to ensure correct operation and to build new software. These development milestones are called base levels. Release 2 of RSX-11M is base level BL12.
<memory size>	- the memory size (in the form of nnnK) of the machine on which the system was bootstrapped.
[MAPPED]	- MAPPED is displayed for systems built to use the memory management unit.
dd	- Device name.
nn	- Unit number.
[1,2]STARTUP	- The file specifier for the indirect command file which is invoked automatically at bootstrap time and which contains user commands for initializing the bootstrapped system.

The system is now ready for operation. The RED and MOU display are internally-generated Redirect and Mount commands. The Redirect command establishes the load device as the system device SY:, and the Mount command mounts the load device as a Files-11 volume.

2.5 ERROR DETECTION AND HANDLING

Error detection and reporting are provided for the various MCR commands. When an error is detected, an appropriate message (prefixed by the name of the command) is displayed at the entering terminal. Messages which are unique to a given command are listed with the discussion of the command in Chapter 4. All error messages are listed in Chapter 6 in alphabetical order.

2.6 COMMAND STRINGS

When typing MCR command strings, the following conventions apply:

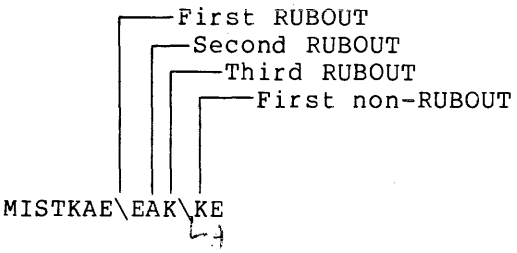
1. Command strings are terminated either by a carriage return (<CR>) or ESCape (<ESC>) key depression. If the command is terminated by a <CR>, a default prompt will be displayed on completion of the command. If the command is terminated by an <ESC>, the default prompt will not be displayed when the command completes. Default prompt suppression is used in conjunction with a special feature of the Run command. Thus, if the following sequence is entered:

   ```
   >RUN TEST <ESC>
   ```

   no default prompt will be displayed when the Run command completes the request to initiate the task TEST. Rather, as a feature of the Run command terminated by an ESCape action, the default prompt will be displayed when the task TEST exits. This feature enables the operator to distinguish when a task, which produces no output on the terminal, has completed.

2. A command must be separated from its parameters by at least one space or one tab.

3. If an error is discovered while entering a command string prior to typing a terminator, the line may be deleted back to the prompting character (>) by typing CTRL/U (formed by simultaneously pressing the CTRL and U characters). RUBOUT, which is echoed as a backslash (\), may be used to delete the last character or contiguous characters typed. The first RUBOUT typed will cause \ and the character deleted to be echoed. Further RUBOUTs cause only the deleted character to be echoed. The next non-RUBOUT character entered will cause a \ to be echoed, along with the character entered. Subsequent non-RUBOUT characters (with no intervening RUBOUTs) are echoed. Thus:

   Produces:

   ```
   MISTAKE
   ```

4. Any number of alphanumeric characters may be concatenated to the first three letters of the command and before its arguments or command string terminator (carriage return or ESCape). One or more spaces or tabs must separate the command name from its operands. Spaces or tabs are useful in improving the readability of printed copy. For example, the

Install command could be invoked to install the task FAULT by typing:

```
INS FAULT
```

or

```
INSTALL FAULT
```

2.7 PERIPHERAL DEVICE NAMING AND ASSIGNMENT

During system generation, all peripheral devices attached to the system are explicitly described. To simplify the naming of these devices, each device is given a unique name which is used in all commands referring to the device.

All peripheral devices are referenced by using a 2-character ASCII device name and an optional 1- or 2-digit octal unit number (for example, DK0: and MT1:). The combination of device name and unit identifier is referred to as the device-unit pair or simply the device-unit. If the unit number is omitted, the system uses unit 0 by default; thus, LP: indicates line printer 0. Table 2-2 lists the device names supported by RSX-11M. The hardware controller designations appear in parentheses.

Table 2-2
RSX-11M Peripheral Devices

PERIPHERAL DEVICES	DEVICE-UNIT
Analog-to-Digital Converter (AD01-D)	ADnn:
(AFC11)	AFnn:
Card Reader (CR11)	CRnn:
Cassette (TA11)	CTnn:
DECtape (TC11)	DTnn:
DISK (RH11/RH70/RP04)	DBnn:
(RF11)	DFnn:
(RK11)	DKnn:
(RP11)	DPnn:
(RH11/RH70/RS03/RS04)	DSnn:
(RX11)	DXnn:
Industrial Control System Local and Remote (ICS/ICR-11)	ICnn:
Laboratory Peripheral System (AR11)	ARnn:
(LPS11)	LSnn:
Line Printer (LS11/LP11/LV11)	LPnn:
Magtape (RH11/RH70/TM02)	MMnn:
(TM11/TMA11)	MTnn:
Paper Tape Punch (PC11)	PPnn:
Paper Tape Reader (PC11/PR11)	PRnn:
Parallel Line Interface (DA11-B)	XBnn:
Synchronous Line Interface (DL11-E)	XLnn:
(DP11)	XPnn:
(DQ11)	XQnn:
(DU11)	XUnn:
Terminal (DL11/DH11/DJ11)	TTnn:
Universal Digital Controller (UDC11)	UDnn:

Table 2-2 (Cont.)
RSX-11M Peripheral Devices

PSEUDO DEVICES	DEVICE-UNIT
Console Listing	CL:
Console Output	CO:
Network	NT:
Pseudo Input Terminal	TI:
System Default Device	SY:

Programs communicate with peripheral devices through logical unit numbers (LUNs). By relating a LUN to a device-unit, input/output issued to that LUN will be directed to the device represented by the device-unit designation. LUNs are associated with device-units at one of three stages in the task creation process:

1. During task build;
2. Via an MCR Reassign command after the task has been installed, and
3. At task run time via the Assign LUN Executive directive.

2.7.1 Device Reassignment

The device assignment process effects a name connection. It relates a program name (LUN) to a physical device-unit; nothing more is implied. If the device-unit has been generated into the system, the name connection will succeed. Success, however, does not necessarily mean the device can be immediately accessed. The device may, for example, be attached to another task, or it may be physically inoperable; neither of these conditions will prevent a valid assignment from occurring. Successfull access can subsequently be made when the device-unit becomes unattached or the malfunction is corrected.

2.7.2 Device Redirection

The operator, in practice, does not often reassign devices. A much more common operator function is that of device redirection.

Redirection causes I/O directed to one device-unit to bypass the original name connection and instead be directed to a different physical device-unit. Physical device redirection is a system wide phenomenon. Thus, any task referring to a redirected device will have its I/O redirected to the target device.

Suppose, for example, that a task has LUN 1 assigned to TT:. Assume during the task's execution that TT: becomes inoperable, making it impossible for the task to continue. The operator can redirect I/O destined for TT: to TT1:. This redirection remains in effect until another Redirect command changes it. Redirection can occur to any level. Thus, if TT1: in the above example becomes inoperable, I/O can be redirected to TT5: or any other appropriate device.

Note that redirection does not in any way alter the original LUN-to-physical-device-unit assignment. If the task is removed from memory and then subsequently re-activated, and the internal redirection is still in effect, the redirect path will automatically hold for the task.

2.7.3 Pseudo Devices

The redirect process usually involves five pseudo devices:

CO: Used for console output

CL: Used for console listing

NT: Used for network communications

TI: Used for terminal input

SY: Used as the system device

Virtually every task in the system, including system tasks, has a need to communicate with one or more of these pseudo devices. Unlike the real devices in the system (e.g., DK0:, TT3:, and LP1:), pseudo devices are so named because they do not correspond to real devices until they are redirected.

In a given task, LUN 1 may be assigned to CL:. This task, in delivering output to LUN 1, wants the data sent to the console listing device; the task itself is not concerned with the particular device in the system actually serving as the console listing device.

The operator can change the device to be used as the console listing device by redirecting CL: to another device-unit (e.g., from a line printer to a terminal). The effect of this redirection is to cause all I/O directed to CL: to go to the terminal, rather than to the line printer. Note that the tasks directing I/O to CL: are unaware of this change and are not altered in any way.

When a pseudo device-to-real device redirect connection is made, the pseudo device assumes the privileged/non-privileged attribute of the real device. Thus, if CO: is redirected to TT0: (which is privileged), and then subsequently redirected to TT1: (which is not privileged), CO: loses its privileged status.

Normally, CO: is redirected to the main operator's terminal, and CL: is normally redirected to the line printer. The psuedo device TI: (terminal input/output) is the pseudo device most frequently used. When a task is initiated via an MCR Run command, the terminal from which it is started is established as the task's TI: terminal. The initiated task can communicate with the user via the TI: pseudo device. If a task is initiated internally via a Run Executive directive, TI: is defaulted to CO:.

2.7.4 Logical Devices

Logical device names may be assigned to physical devices by means of the MCR Assign command. A logical device name consists of a 2-character ASCII name and an optional 1- or 2- digit octal unit number. Any reference to a logical device will be mapped by the system into the physical device to which it is assigned. There are two types of logical device assignments: local and global. Local logical device assignments are in effect only for tasks initiated from the entering terminal. That is, local logical device assignments are associated with a particular terminal. Global logical device assignments affect all tasks running in the system.

The logical device table is always searched by the system before the physical device table. This precedence allows logical device assignments to override physical device names. Local device assignments override any global assignments, and global assignments override physical device names. Similarly, identical local logical device names may be used at different terminals without interference.

This feature, which is a system generation option, allows a programmer to write a program without concern for the device actually used at run time. He may reference a logical device name and assign it to the desired device prior to run time. For examples of logical device assignment, see the description of the Assign command in Chapter 4.

CHAPTER 3

FILES AND FILE SPECIFIERS

3.1 INTRODUCTION

This chapter is not mandatory reading for operators. It is included to add depth to the operator's understanding of files, file naming conventions, and file protection.

3.2 FILE FUNDAMENTALS

Files are owner-named areas on direct access volumes. Volumes are data-carrying media which can be file structured. This structure and the software which supports it is called Files-11. In RSX-11M, disks and DECtapes comprise the supported volume types. DECtape may be either file structured or contain a data format supported by the File Transfer Program (FLX). Cassettes and magtapes are not file-structured volumes; the structure on these devices is supported by the File Transfer Program (FLX) or user I/O.

Files behave much like devices in that task I/O may be directed to them. However, since a file is just a portion of a random access volume, files can be created as long as space for them exists on the volume. Thus, they are much more dynamic and flexible than physical devices. Due to the fact that more than one programmer's files may exist on the same volume, conventions are required for naming the areas and techniques for protecting files belonging to different owners from being altered, except by the owner or an other user with the owner's permission.

Filenames are made unique by use of a user identification code (UIC). A file cannot be accessed unless the UIC under which it is stored is known. Knowing the UIC does not guarantee access, however, since each file existing under a given UIC can be individually protected. Individual file protection is discussed in Section 3.2.1.

Associated with each UIC is a user file directory (UFD). The UFD is a file which contains the names of all files currently existing under a given UIC. A UFD is itself a file belonging to a specific user and has all the characteristics of a file, including the manner in which access to it is granted.

UICs are specified in the format [nnn,nnn]; nnn is a 3-digit number, usually represented in octal, having a range of 0 thru 377(8). The first number in the pair is referred to as the group number; the second, the member number. To eliminate repetitive entry of UICs in command strings, a default UIC can be established for each terminal in

the system, either at system generation or dynamically from any terminal with the Set command. When a command requiring a UIC is entered from the terminal, and the UIC is not explicitly specified, the default is used.

3.2.1 File Protection*

Four types of file actions exist:

Read;

Write;

Extend, and

Delete.

The users that are allowed to perform these actions are divided into four categories:

System -- All system software (all tasks with a group number of 10(8) or less),

Owner -- The user under whose UIC the file is stored,

Group -- Any user whose UIC has the first three digits in common with the owner,

World -- Any user not included in the three categories described above.

When protection is specified, any combination of the four actions (read, write, extend, delete) can be assigned to each of the four groups. For example, the system and the owner can have read, write, extend, and delete privileges, the group can have read privileges, and the world can be denied access to the file.

In RSX-11M, the basic protectable entity is the file. The UIC is required to obtain the opportunity for accessing a given set of files. But each file stored under a given UIC is individually protected. Access to a specific file, given the UIC, is established by the owner. The owner can prevent access to each of his files, and he can extend different access rights to the four classes of users described above.

The owner of a file establishes access privileges to these four user classes in one of five ways:

1. At volume initialization time.

   The Initvolume command allows the user to specify default file protection for all files created on the volume.

---

* Volumes are protected from illegal access by non-privileged tasks if they have been mounted with the MCR Mount command. See the Mount command for additional details (Section 4.3.2).

2. The MCR User File Directory command.

   This command establishes access rights to the UFD file only. Often, a user will permit access to his UFD, but will restrict access to the files catalogued in the UFD.

3. PIP File Utility.

   PIP has an option which enables the owner to alter access privileges of his files.

4. Within a user task.

   Using the WRITE ATTRIBUTES Files-11 I/O function, a user can write new access protection attributes to files he owns.

5. At mount time.

   In the Mount command, the user may specify the default file protection to be given files created after the mount operation; this specification overrides that specified during volume initialization.

3.2.2 File Specifiers

Any component of RSX-11M which has a requirement to refer to files does so via a standard command string with the following general format:

```
outflespcl,...outflespcn = inflespcl,...inflespcn
```

Outflespc is an output file specifier, and inflespc is an input file specifier.

Any number of specifiers are possible, the actual number being determined by the task which will use the command string. In no case, however, may the total length of the command string exceed the maximum line length (80 bytes).

Each file specifier (whether input or output) has the following format:

```
dev:[g,m]filename.type;version/switch1.../switchn
```

where:

dev: = The physical device-unit on which the volume containing the desired file is mounted, for example, DK0: or DT1:. The designation consists of a 2-character ASCII name followed by an optional 1- or 2-digit octal unit number and a colon.

[g,m] = The user identification code (UIC) associated with the user file directory containing the desired file. The UIC consists of a group number and a member number.

filename = The name of the file. In RSX-11M, a filename can be up to nine alphanumeric characters in length. The filename and type are always separated by a period (.).

type = A means of distinguishing among various forms of one file. For example, a source FORTRAN program might be named COMP.FTN, while the object code associated with that program might be called COMP.OBJ. The use of file types is discussed in section 3.2.5. The file type and version are always separated by a semicolon (;).

version = An octal number used to differentiate among various versions of a file. For example, when a file is first created, it is assigned a version number of 1. If the file is subsequently opened for editing, the original file is retained for backup, and a new file is created with the same filename and type, but with a version number of 2. The version number is in the range 0 thru 77777.

/switch = A 2-character ASCII name identifying a switch option. The switch itself may have three forms. If the switch designator, for example, is SW, then:

```
/SW      Sets the switch action;
/-SW,    Negates the switch action, and
/NOSW    Also negates the switch action.
```

In addition, the switch identifier may be followed by any number of values. The permitted values are ASCII strings, octal numbers, or decimal numbers. The default for a value may be octal or decimal. Values which are terminated by a decimal point are explicitly decimal. Values preceded by a pound sign (#) are explicit octal. Finally, any numeric value may be preceded by a + or - sign; plus is the default notation. If explicit octal (#) is used, the sign (if one is used) must precede the # (pound) sign. The following are valid switch specifications:

```
/SW:27:MAP:29.
/-SW
/NOSW:-#50:SWITCH
```

The number of permissible values and the switch interpretations themselves depend entirely on the particular task to which they are directed.

3.2.3 Volumes and Files

The following discussion of volumes and files is intended as a structural sketch of the procedures needed to create, name, and access files. Details appear under the discussion of the associated MCR commands.

In the previous section, filename strings were discussed. A filename, however, refers to a file on a file-structured volume, and several steps are required before such a file can actually be created and subsequently referred to by name.

Volumes are data carrying media which can be file structured. In RSX-11M, disks and DECtapes comprise the supported file-structured volumes.

Before the RSX-11M file system cna access or create a file from a user-supplied filename, the volume must be initialized, it must be mounted, and a user file directory must be created. These three steps are accomplished by the MCR Initvolume, Mount, and User File Directory commands; all three commands are discussed in Section 4.3.2.

A file-structured volume is organized such that files contained on the volume can be retrieved by their names. Directories are used to map names to the physical locations on the volume where the referenced file is stored. The Initvolume command establishes the basic structural framework for these file directories. A volume which has not been processed by the Initvolume command is defined as a foreign volume; such a volume cannot be processed by the RSX-11M file system.

Each owner having a unique UIC should have a directory file of all the files created under his UIC. No other user of the system can access these files unless they know the UIC and have access privileges to the file. The User File Directory command establishes this directory file for a UIC. Once the directory file is established, the owner can create, reference, or delete files specific to his UIC.

Finally, any volume to be accessed by the file system must be made known to the system via the Mount command. Note that unmounted volumes can be accessed directly by user tasks, which is often a requirement in time-critical applications.

Thus, before a filename can be used to access a file, the volume which is to contain the file must be initialized, the volume must be identified to the system via the Mount command, and a user file directory which will contain the name of the file must be created. Volume initialization and user file directory creation need only be performed once for a volume; mounting, however, must be done after each bootstrap of the system.

3.2.4 Defaults in File Specifiers

Any of the elements of the file specifier can be defaulted, except the filename*. The system uses defaults as specified below:

STRING ELEMENT	SYSTEM DEFAULT
dev:	SY:. -- The device-unit on which the system volume is mounted.
[g,m]	The default UIC specified for each terminal during system generation. The UIC default may be changed dynamically with an option of the MCR Set command.
filename	No default. Must be specified explicitly or implicitly (using an asterisk) as described in Section 3.2.6.
type	A user-meaningful file type. For example, if a list file is created, the system assigns it a file type of LST in the absence of a user-specified type. See Section 3.2.5 for a list of default file types.
version	For input files, the most recent version number. For output files, the next higher version number. An exception is the PIP file delete function, which requires an explicit version number; this feature prevents the user from inadvertently deleting the latest version of a file.
/switch	Switch defaults are established by the individual system tasks which require them.

3.2.5 Use of File Types

RSX-11M has a standard set of file types which are used when the type is omitted from a file specifier. While the user can assign his own file types, system operations are facilitated if use is made of the file types presented in the following table. These mnemonic types are used by all DEC-supplied software to reflect actual file contents.

FILE TYPE	MEANING
CMD	A file containing a list of task or MCR commands (indirect file).
DAT	A data file, as opposed to a program file.

* The MCR Boot command is the one filename-requirement exception. See Boot command in Section 4.3.2 for the default.

DIR	A directory file, for example, a user file directory.
FTN	A FORTRAN language source program.
LST	A listing file.
MAC	A MACRO-11 source program.
MAP	A Task Builder memory allocation file.
MLB	A user macro library.
OBJ	An object program (output from MACRO-11 or FORTRAN).
OLB	An object module library.
SML	The system macro library.
TSK	A task image.

The defaults used by DEC-supplied software appear in the manual(s) describing individual software components.

3.2.6 Asterisk Convention (Wildcards)

In addition to relying on defaults in the file specifier, the user also can place an asterisk (*) in the specifier to indicate that the content of that element of the string will be satisfied by all existing values. By permitting an "any value" indicator within a specifier, the user can affect more than one file at a time.

To illustrate this capability, the Peripheral Interchange Program (PIP) supplied by DEC with RSX-11M will be used. PIP performs a number of functions, one of which is deleting files. To delete the file PROG.MAC;1, supply the following command string to PIP:

```
PIP>PROG.MAC;1/DE
```

The string:

```
PIP>
```

Is PIP's console prompting string; the notation /DE is PIP's delete switch.

Assume that the programmer has a requirement to delete the files PROG.MAC;1, PROG.OBJ;1, and PROG.TSK;1, and that these files are the only existing PROG version 1 files. Deletion can be accomplished using the asterisk convention as follows:

```
PIP>PROG.*;1/DE
```

The effect of the asterisk is to cause PIP to delete all PROG version 1 files, regardless of type. Thus, the asterisk can represent any value; this characteristic is the source of the terminology "wildcard specifier."

An asterisk can be placed in any portion of the file specifier except the device indicator. The device indicator must be specified or defaulted to SY:.

3.2.7 Examples of Filename Command Strings

In the following examples, the three letters followed by >, are standard prompts for the respective system programs.

Example 1:

Assemble the MACRO-11 source file CRGPT.MAC and create the object output file CRGPT.OBJ. Both files are on DK0: under UIC [200,200].

```
MAC>DK0:[200,200]CRGPT.OBJ=DK0:[200,200]CRGPT.MAC
```

Example 2:

Delete the file SBG.OBJ;5. The output filename string is omitted because it is not applicable. The input device is SY:, and the [200,200] UIC is defaulted.

```
PIP>SBG.OBJ;5/DE
```

The UIC [200,200] in this example will be selected if, and only if:

1. A /UIC switch has been specified via PIP, or
2. PIP is running under UIC [200,200].

Example 3:

Task build the object file CRGPT.OBJ. The output will be named CRGPT.TSK because TSK is the default file type for the Task Builder's output file; device and UIC are also defaulted.

```
TKB>CRGPT=CRGPT.OBJ
TKB>//
```

The default UIC [200,200] in this example will be selected if, and only if, TKBi is running under UIC [200,200].

Example 4:

Delete all files with the name CRGPT, regardless of type or version; SY: and [200,200] are defaulted.

```
PIP>CRGPT *;*/DE
```

The default UIC [200,200] in this example will be selected if, and only if;

1. A /UIC switch has been given in PIP, or
2. PIP is running under UIC [200,200].

CHAPTER 4

MCR COMMANDS

4.1 COMMAND SUMMARY

This chapter begins with a list, by category, of all MCR commands. The list is identical to the order in which the commands are discussed in detail. The page number for the command is in square brackets immediately preceding the functional description of the command. This command summary provides a compact overview of the facilities available from an operator console.

Initialization Commands

ASN**	[4-8]	Define or delete a logical device assignment. List current assignments on user's terminal.
BOOT*	[4-10]	Bootstrap a new system into memory and transfer control to it.
DMOUNT*	[4-12]	Delete the volume control block (VCB) and declare volume off-line.
INITVOLUME*	[4-14]	Initialize an RSX-11M Files-11-structured volume.
INSTALL**	[4-22]	Install a task in the system.
MOUNT*	[4-28]	Create the volume control block (VCB) and declare volume on-line for access by the file system.
SET**	[4-33]	Set system and terminal characteristics; redefine partitions; display terminal and partition characteristics on entering terminal.
TIME**	[4-43]	Enter time and/or date into the system; display the time and date on entering terminal.
UFD	[4-44]	Create a user file directory (UFD) file and enter its filename into the volume's master file directory (MFD) file.

* Commands with a single asterisk are privileged; the asterisk is not part of the syntax.

** Certain options are privileged on commands with double asterisks.

Informational Commands

DEVICES*	[4-47]	Display the list of peripheral devices recognized by the system on the entering terminal.
LUNS*	[4-48]	Display the list of LUN assignments for an indicated task on the entering terminal.
PARTITIONS*	[4-49]	Display the list of partition definitions on the entering terminal.
TASKLIST*	[4-5Ø]	Display the System Task List on CL:.

Task Control Commands

ABORT***	[4-51]	Terminate execution of a running task.
ALTER*	[4-54]	Alter the priority of a task.
CANCEL***	[4-55]	Cancel time-based initiation requests for a task (no effect on current execution).
FIX*	[4-56]	Fix a task in memory (task becomes memory resident).
REASSIGN*	[4-57]	Change LUN assignment.
REDIRECT*	[4-58]	Redirect all I/O requests from one physical device to another.
REMOVE*	[4-6Ø]	Remove a task from the system.
RESUME***	[4-61]	Resume execution of a suspended task.
RUN	[4-62]	Schedule a task's activation. The task may run immediately, after a delay, or in synchronization with the system clock. Periodic rescheduling is optional.
UNFIX*	[4-67]	Unfix a task from memory.

* Commands with a single asterisk are privileged; the asterisk is not part of the syntax.

*** Commands with triple asterisks are privileged if the subject tasks were not started from the terminal from which the command was issued.

System Maintenance Commands

ACT*	[4-68]	Display names of active tasks on the entering terminal.
ATL*	[4-69]	Display name and status information for active tasks on the entering terminal.
BRK	[4-71]	Pass control to the Executive Debugging Tool (XDT). Valid for systems which have XDT generated into them.
OPEN*	[4-72]	Display the contents of a memory location for examination or modification on the entering terminal.
SAVE*	[4-75]	Save the image of memory in the file from which the system was booted.
TAL*	[4-77]	Display names and status of all tasks in the system on the entering terminal.

* Commands with a single asterisk are privileged; the asterisk is not part of the syntax.

4.2 MONITOR CONSOLE INTERFACE

The operator communicates with the RSX-11M system via one of the following terminals:

Teletype* Models ASR/KSR-33 or ASR/KSR-35;
LA30 or LA36 DECwriter, or
VT05 or VT50 Display.

MCR (Monitor Console Routines) is the interface between the terminal and the RSX-11M system. MCR input is initiated by entering unsolicited input on an unattached terminal which has displayed the default prompt, or by typing CTRL/C. If CTRL/C is entered, the system responds by displaying

```
MCR>
```

on the terminal, indicating that it will now accept an MCR command.

When MCR is ready for another command, it displays the default prompt:

```
>
```

All examples show the default prompt.

4.2.1 Command Syntax

It is not necessary to type the entire command name when submitting a command. MCR requires only the first three letters of a command, followed by the command parameters. If parameters exist, they must be preceded by at least one blank or tab.

The following example shows how the Time command may be specified. Note that brackets are used in the example solely to indicate that the "E" is optional; the square brackets are not actually entered as part of the command name.

```
>TIM[E]<CR>**
14:00:04 22-AUG-75
```

Thus, either

```
>TIM<CR>
```

or

```
>TIME<CR>
```

is acceptable.

---

* Teletype is a registered trademark of the Teletype Corporation.

** The angle brackets are used to denote that the enclosed values are not taken literally as part of the command syntax. For example, <CR> indicates that a carriage return terminates the input line.

4.2.2 Keywords

Some commands use keywords which generally apply to a command argument. A keyword is similar in function to the switches described in Section 3.2.2. A keyword consists of a slash (/), followed by an ASCII identification, and optionally followed by an equal sign (=) and the value of the keyword, as follows:

```
/Keyword=value
```

Keywords can be entered in any order. As an example of keyword usage, the Install command requires a filename argument specifying the task to be installed. Keywords can be appended to the filename. One such keyword, /TASK, can specify the name under which the task is to be installed. Thus,

```
>INS JIM.TSK/TASK=SUPER
```

will cause the task contained in the file named JIM.TSK to be installed with a name of SUPER. Keywords are command specific and are defined with each command.

An example of a keyword without a value is the UNLOCK switch of the Mount command

```
>MOU DK0:/UNL
```

which specifies that disk unit 0 is to be mounted unlocked.

4.2.3 Comments

MCR will treat a line of text as a comment if the first character in the line is a semicolon (;). In addition, the exclamation mark (!) may be used to delimit comments in a command. The first exclamation mark starts the comment and the next exclamation mark or end of line terminates the comment. All text between the two exclamation marks is ignored. For example:

```
>;  THIS LINE IS A COMMENT

>TAS !THIS IS A COMMENT STRING!
```

This feature is especially useful to comment commands in MCR indirect files which are described in Section 5.1.

4.3 OPERATOR COMMANDS

Command names are at least three characters in length. MCR accepts the first three characters of a command and then searches for a blank, tab, carriage return, or ESCape character. Therefore, embedded blanks are not allowed in a command name. If a command is entered incorrectly, an error message will be displayed at the entering terminal.

Syntactical descriptions of commands and messages described in this chapter observe the following notational conventions:

1. Lower case indicates a variable whose actual value is determined when the command is entered or the message is issued. For example, the value of taskname depends on the name of the task associated with the command or message.

2. Brackets [] enclose optional items. A syntactical element enclosed in brackets (e.g., [dt]) may or may not be included in the command, at the user's option. The one exception to this rule is the syntax for the specification of a UIC. The format for a UIC is [g,m], where the square brackets are required syntactical elements.

3. Unless explicitly qualified, all numeric values required in a command may be entered as decimal or octal. Decimal values are indicated by a trailing period; octal values are indicated by the absence of a trailing period. Thus,

   255.

   and

   377

   have the same value. (The value 255. is base 10; the value 377 is base 8.)

4.3.1 Command Description Format

Each command is described using the following subheadings; those that do not apply are omitted.

COMMAND NAME

This section is headed by the command name in capital letters. On the right margin, the minimal MCR command mnemonic is shown. If the command can be executed only from a privileged terminal, the word "privileged" appears in parentheses following the command name.

Following the heading, the command's intent within the system is described.

Format:

The command format is given, and all parameters are described.

Examples:

Example uses are shown.

Command Related Error Messages:

Error messages specific to the command appear in this section. The command may also produce messages listed in Chapter 6 (common error messages), but these are not listed here.

Notes:

A list of special considerations that may prove helpful in assisting the programmer in the proper use of the command.

4.3.2 Initialization Commands

ASSIGN (some privileged options)

ASN

ASN is used to define, delete, or display logical device assignments. Logical device assignments are a convenient way to associate logical names with physical devices. There are two types of logical device assignments: local and global. Local assignments apply only to commands and tasks initiated from the terminal from which the assignment was made. Each terminal may have its own set of asssignments. Global assignments apply to all commands or tasks unless a logical name is defined both as global and local, in which case, the local assignment overrides the global assignment. Logical device names may be the same as physical device names, or they may be purely random as long as they follow the syntax for device names. This syntax requires a 2-character ASCII device name followed by an optional 1- or 2-digit octal unit number. When referring to a logical device, the name must be terminated by a colon.

Format:

ASN ppnn:=llnn:[/GBL]

Assign logical name llnn to physical device ppnn. If /GBL is specified, the assignment is global. A terminal must be privileged to define or delete global assignments.

ASN

Display local logical assignments.

ASN /GBL

Display all local and global logical assignments for all terminals in the system.

ASN =llnn:[/GBL]

Delete the local device assignment of logical name llnn. If /GBL is specified, the global assignment is deleted.

ASN =[/GBL]

Delete all local logical assignments. If /GBL is specified, all global assignments are deleted.

where:

pp - Physical device name

nn - Unit number

ll - Logical device name

Example:

```
>ASN   DP:=US1:      !Define US1:
>ASN   DT:=US2:      !Define US2:
>ASN   DK:=GB0:/GBL !Define global logical device name GB0:
```

```
>ASN                    !List local assignments
US2:    DT0:    LOCAL TI - TT0:
US1:    DP0:    LOCAL TI - TT0:
>ASN    /GBL            !List all assignments
US2:    DT0:    LOCAL TI - TT0:
US1:    DP0:    LOCAL TI - TT0:
GB0:    DK0:    GLOBAL
>ASN    =US2:           !Eliminate local logical device name US2:
>ASN    =/GBL           !Eliminate all global assignments
>ASN    =US1:           !Eliminate local logical device name US1:
```

Command Error Messages:

ASN -- LOGICAL DEVICE NOT IN SYSTEM

The specified logical device name has not been defined previously and therefore cannot be found in the logical device assignment table for the terminal.

ASN -- PSEUDO DEVICE ASSIGNMENT ERROR

A logical device name may not be assigned to a pseudo device.

BOOT (privileged)

BOO

Boot is used to bootstrap a system which exists as a task image file on a file-structured volume. It provides a convenient means for terminating one system and initiating another, especially on minimum hardware configurations. For example, Boot can be used for terminating a real-time system and initiating a program development system.

Format:

BOO[T] [filespec]

where:

filespec - The standard RSX-11M file specifier of the form

dev:[group,member]filename.type;ver

that specifies the file from which a new system is to be loaded.

Defaults applied to the file specifier are:

dev:	- The system device SY:
[group,member]	- The UIC under which BOO is running
filename	- RSX11M
type	- SYS
version	- Latest version

Example:

```
>BOO TEST
```

Bootstrap from the system device the system contained in the file TEST.SYS.

Command Error Messages:

BOO -- DPB ERROR

A bad DPB was created by MCR. This error indicates that the system itself has faulted. If the error persists, consult your local DEC software specialist.

BOO -- FILE NOT CONTIGUOUS

An attempt is being made to load a system from a non-contiguous file. System images must be contiguous.

BOO -- INVALID LOAD DEVICE

Boot has detected a device which is invalid as a system residence device.

BOO -- LABEL BLOCK READ ERROR

Boot cannot read the label block of the system image.

BOO -- NO TRANSFER ADDRESS

Boot could not find a transfer address in a virgin system image (result of a task build).

BOO -- NOT SYSTEM IMAGE

Boot has determined that the file is not a system image.

Notes:

1. The system in operation at the time of Boot command execution is immediately terminated.

2. The Boot command is the only command with an optional filename specification.

3. Bootable systems are special task image files that have no task headers. Examples of such systems are RSX-11M, RSX-11S, PRESRV.

DMOUNT (privileged)

DMO

Dismount declares that the volume or communications channel is logically off-line. After a dismount operation, the device cannot be accessed by the associated Ancillary Control Processor (ACP). For a Files-11 volume, a request is placed in the file system queue to delete the volume control block (VCB), and the volume is marked for dismount so that no further files can be accessed on the volume. The VCB is a main memory resident control block that controls volume access. The command is completed when the VCB is actually deleted. The VCB deletion does not occur until all accessed files on the volume have been de-accessed. When the de-accessing process is complete, a message is issued, indicating that the dismount operation is complete. For a network device, the communication volume control block is immediately deleted, and all further processing on the channel is terminated.

Format:

DMO[UNT] dev:[volume-label]

where:

dev: - Device-unit of the volume to be dismounted.

Default: none; must be specified.

volume-label - The Files-11 volume label may be up to twelve characters in length. It is used to validate that the proper volume is being dismounted. A volume label may not be specified for a communications channel.

Default: Twelve nulls (null volume label).

Examples:

```
>DMO DK1:RVOLUME
```

This command causes the VCB for DK1: to be deleted.

```
>DMO XU:
```

This command causes the network ACP to terminate communication over the channel connected to device XU:

Command Error Messages:

DMO -- ALREADY MARKED FOR DISMOUNT

The device-unit has already been requested to be dismounted and is in the process of waiting for all accesses to the volume to complete.

DMO -- HOME BLOCK CHECKSUM ERROR

The checksum in the home block and the calculated checksum do not agree. This message is probably caused by an I/O error.

DMO -- HOME BLOCK I/O ERROR

An error was detected in updating the volume file sequence number in the volume home block.

DMO -- NOT MOUNTABLE DEVICE

The device specified is not a mountable device and therefore cannot be dismounted.

DMO -- NOT MOUNTED

The device specified is not currently mounted.

DMO -- WRONG VOLUME

The volume label and the label specified in the command do not match.

Notes:

1. The message

*** ddnn: DISMOUNT COMPLETE

is issued to CO: when the dismount has been completed either by the file system or the network ACP. There may be a considerable delay between the issuance of the command and the printing of this message if there are a number of I/O requests pending and/or files accessed on the volume.

INITVOLUME (privileged) INI

The function of the Initialize Volume command is to produce a Files-11-structured volume. In particular, it initializes the volume (all existing files are destroyed), writes a dummy bootstrap and a home block, and builds the directory structures. The details of the Files-11 structure are discussed in the RSX-11 I/O Operations Reference Manual.

Format:

INI[TVOLUME] dev:[volume-label][/keyword][/keyword]...

Keywords:

```
/BAD  = [option]
/CHA  = [characteristics]*
/EXT  = block-count
/FPRO = [system,owner,group,world]
/INDX = index-file-position
/INF  = initial-index-file-size
/MXF  = file-count
/PRO  = [system,owner,group,world]*
/UIC  = [group,member]*
/WIN  = retrieval-pointer-count
```

where:

dev: - Device-unit of the volume to be initialized.

Default: none; must be specified.

volume-label - The volume label may be up to twelve characters in length. It is used to identify the volume.

Default: twelve nulls.

BAD - This keyword indicates that bad block processing is to be included in the volume initialization. The result of this will be that bad blocks on the volume will be marked as in-use and will not be allocated to files. The options are:

[MAN] - Accept block specified from terminal.

[AUTO] - Read bad block file on volume created by diagnostic routine (BAD).

[AUTO,MAN] - Read bad block file and, when done, accept blocks specified from terminal.

* The asterisked keywords are included for RSX-11D compatibility. Thus, volumes built on RSX-11M can be mounted on an RSX-11D system.

The brackets are required syntax. If MAN is specified, the program will prompt for bad blocks with:

INI>BAD=

Bad blocks may be entered in two formats.

nnnnn - A single block

nnnnn,mmmm - A contiguous series of mmmm blocks beginning at nnnn. A null line (carriage return) terminates bad block input.

CHA - The characteristics words are optional ASCII strings separated by commas. The square brackets are required syntax. For the Initvolume command, two characteristics are defined:

ATCH - ATTACH/DETACH (device can be used by one task or group exclusively); included for RSX-11D compatibility. Unused by RSX-11M.

Default: no Attach/Detach.

DCF - Device Control Functions are permitted; included for RSX-11D compatibility. Unused by RSX-11M.

Default: no DCF.

EXT - The default number of blocks by which a file exhausting its space allotment is to be extended.

Default: five blocks.

FPRO - Keyword for default file protection. Access codes consist of four, 4-code groups in the Access Rights word, as follows:

R = Read
W = Write
E = Extend
D = Delete

In each instance, the absence of the code means that the access right is denied the user. The square brackets are required syntax.

Default: [RWED,RWED,RWED,R].

NOTE

Protection code subparameters (system,owner,group,world) are positional. That is, the location of the word in the parameter string defines the user to whom the code applies. The order is:

system,owner,group,world

The order of appearance of the characters R, W, E, and D is fixed.

R, if desired, must be first.
W, if desired, is next.
E, if desired, is next.
D, if desired, is last.

Thus, RWE and RE are acceptable, while WR and DEWR are not.

INDX - Index file logical block number. This can be used to force the index file, master file directory (MFD), and storage allocation file to a specific volume location, usually for minimizing access time. Four possibilities are available:

BEG - Place index file at the beginning of the volume.

MID - Place index file at the middle of the volume.

END - Place index file at the end of the volume.

BLK:nnn - Place index file at the block number specified.

Default: middle of volume.

INF - Initial index file size. This is the number of file headers to allow space for in the first index file allocation.

Default: 16 files.

MXF - The maximum number of files (file headers in the volume's index file) permitted on the volume; this number must be less than or equal to the number specified for the device in the table below:

	Maximum
DECtape	144
RK Disk	1200
RF Disk	256
RS03 Disk	256

RS04 Disk	512
RP02 Disk	10000
RP03 Disk	20000
RP04 Disk	42949

Defaults

DECtape	72
RK Disk	600
RF Disk	128
RS03 Disk	128
RS04 Disk	256
RP02 Disk	5000
RP03 Disk	10000
RP04 Disk	21474

NOTE

A Files-11 volume requires five files to create the on-disk structure (See Appendix E of the RSX-11 I/O Operations Reference Manual). Thus, MXF must be computed by the maximum number of user UFD's and files plus the 5 required for the Files-11 structure.

PRO* - Keyword for volume access rights parameter. Access codes consist of four, 4-code groups in the Access Rights word, as follows:

R = Read
W = Write
E = Extend
D = Delete

In each instance, the absence of the code means that the access right is denied the user. The square brackets are required syntax.

Default: [RWED,RWED,RWED,RWED].

NOTE

Protection code subparameters (system,owner,group,world) are positional. That is, the location of the word in the parameter string defines the user to whom the code applies. The order is:

system,owner,group,world

The order of appearance of the characters R, W, E, and D is fixed.

R, if desired, must be first.
W, if desired, is next.
E, if desired, is next.
D, if desired, is last.

Thus, RWE and RE are acceptable, while WR and DEWR are not.

UIC* - Keyword for UIC parameter specifying the owner of the volume. Legal group and member numbers range from 1 to 377(8). The square brackets are required syntax.

Default: [1,1]

WIN - The number of mapping pointers to be allocated for file windows. A file window is made up of a number of mapping pointers and is stored in memory when the file is opened. (See Appendix F in the RSX-11 I/O Operations Reference Manual for a description of a mapping pointer.)

Default: 7.

Example:

```
>INI DK1:ICTSVOL2/UIC=[2,5]/INDX=BEG/FPRO=[RWED,RWE,RW,R]<CR>
```

The Initvolume parameters in the example above are:

DK1: - Device-unit

ICTSVOL2 - Volume-label

/INDX=BEG - Index file location, forced to beginning of volume

/FPRO - Default file protection, specified by group as:

system:	Read,Write,Extend,Delete
owner:	Read,Write,Extend
group:	Read,Write
world:	Read-only

/UIC - User identification code: group=2, member=5

* RSX-11M does not protect a volume at the UIC level. It does, however, prevent direct access to MOUnted volumes by non-privileged tasks. This parameter is included to permit volume interchangeability with RSX-11D.

Command Error Messages:

INI -- ALLOCATION FOR SYS FILE EXCEEDS VOLUME LIMIT

Unable to allocate system file from specified block due to intermediate bad blocks or end of volume.

INI -- BAD BLOCK FILE FULL - TASK ABORTED

The disk has more than 102 bad regions on it.

INI -- BAD BLOCK HEADER I/O ERROR

An error was detected in writing out the bad block file header.

INI -- BLOCK(S) EXCEED VOLUME LIMIT

Specified block (or blocks) exceed(s) physical size of volume.

INI -- BOOT BLOCK WRITE ERROR

An error was detected in writing out the volume boot block.

INI -- CHECKING DDnn

Automatic bad block specification is proceeding using the bad block file provided by the Bad Block Locator utility program.

INI -- CHECKPOINT FILE HEADER I/O ERROR

An error was detected in writing out the checkpoint file header.

INI -- DATA ERROR

Too large a bad block number or contiguous region was specified.

INI -- DUPLICATE BLOCK(S) FOUND

A block which has been defined as bad is being defined as bad a second time.

INI -- DEVICE NOT INIT-ABLE

The device is not a supported Files-11 device.

INI -- DEVICE NOT READY

The specified device is not ready.

INI -- DEVICE WRITE LOCKED

The specified device is hardware write-locked.

INI -- FILE CORRUPT - DATA IGNORED

Bad block file contains bad data.

INI -- HOME BLOCK ALLOCATE WRITE ERROR

In overwriting a bad home block area, a write error occurred.

INI -- HOME BLOCK WRITE ERROR

An error was detected in writing out the volume home block.

INI -- ILLEGAL KEYWORD VALUE

A value entered for a keyword exceeds its limits.

INI -- INDEX FILE BIT MAP I/O ERROR

An error was detected in writing out the index file bit map.

INI -- INDEX FILE HEADER I/O ERROR

An error was detected in writing out the index file header.

INI -- MFD FILE HEADER I/O ERROR

An error was detected in writing out the master file directory file header.

INI -- MFD WRITE ERROR

An error was detected in writing out a block in the MFD.

INI -- NO BAD BLOCK DATA FOUND

Although automatic bad block specification was selected, no bad block file was found on the volume.

INI -- NOT FILES-11 DEVICE

The system does not support Files-11 on the specified device.

INI -- NULL FILE HEADER I/O ERROR

An error was detected in writing out null file headers to the index file.

INI -- STORAGE BITMAP FILE HEADER I/O ERROR

An error was detected in writing out the storage allocation file header.

INI -- VOLUME MOUNTED

Mounted volumes may not be initialized.

Notes:

1. The Initvolume command is closely related to the file system. The parameters are listed and their explanations are necessarily sketchy. Selecting values for the parameters requires an in-depth knowledge of the file system. Refer to both the RSX-11 I/O Operations Reference Manual and the RSX-11M System Generation Manual.

INSTALL (privileged) — INS

The Install command is used to make tasks known to the system and to load resident libraries in common partitions. The filespec parameter is the only required argument. The keywords are optional arguments for parameters that were either not specified at task-build time, or are to be overridden during the execution of the Install command. The process of task installation results in a main memory resident task control block (TCB) being constructed. Once installed, a task may be acted upon by specifying its name in an appropriate command directive.

Format:

```
INS[TALL] filespec[/keyword][/keyword]...
```

Keywords:

```
/CKP  = option
/INC  = size
/PAR  = pname
/PMD  = option
/PRI  = number
/TASK = taskname
/UIC  = [group,member]
```

where:

filespec - The file specifier in the form:

```
dev:[group,member]filename.type;version
```

The standard defaults for file specifiers apply.

Default: TSK

CKP - Checkpoint option. If CKP=YES (default for checkpointable tasks), checkpointing of a checkpointable task will be allowed. If CKP=NO, checkpointing will be disabled for the task.

INC - The number of additional words of address space to be allocated to a task which is to run in a system-controlled partition. The extension size is usually expressed in octal but may also be expressed in decimal by appending a period. This keyword will override the EXTTSK parameter used at task-build time and is used for tasks which determine the size of the partition in which they are running in order to decide how much dynamic space they may allocate to symbol tables or buffers. Tasks such as MACRO-11, TKB, PIP, VFY, AT. use the space from the end of the task image to the end of their address space as dynamic memory. In effect, the INC keyword defines the number of words in this dynamic region.

Default: size specified at task-build time, or zero.

PAR - A partition name which overrides the partition specified at task-build time. In the unmapped system, the partition specified must have the same base address as the partition for which the task was built.

Default: TKB uses the name GEN. (See the RSX-11M System Generation Manual for explanation of multiuser system defaults.)

PMD - Post-Mortem Dump option. If PMD=YES, a post-mortem dump will be requested for the task if it aborts due to an SST error condition. If PMD=NO (the default from a task-build), no dump will be requested at SST abort time. To generate a dump, the post mortem dump task PMD... must be installed in the system. This keyword will override the /PM switch specified at task-build time.

Default: specified at task-build time.

PRI - Priority number. The value range is 1 through 250(10). Number conventions apply.

Default: 50. (decimal).

TASK - Taskname. The name by which the task is to be referred.

Default: first six characters of the filename or the name specified at task-build time. (See the RSX-11M System Generation Manual for explanation of multiuser system defaults.)

UIC - [group,member]. The square brackets are required syntax. This UIC is the default for the task. It can be overridden in the Run command. The task UIC determines into what file protection class (system,owner,group,world) a task belongs, and thus directly influences file access. Note that file protection may differ from file to file.

Default: default from task build. This default applies to tasks started by a Run directive or command. Tasks started by typing the name as a command, e.g., PIP, TKB, etc., run under the default UIC of the terminal from which the command was issued. This UIC is defined by the Set command.

Examples:

```
>INSTALL SCAN
```

Install task SCAN. Priority and UIC defaulted.

```
>INSTALL DK1:[1,111]SCAN.TSK;4/PRI=103
```

Install task SCAN, file type TSK, version 4 on device DK1:, group number 1, member number 111, with a priority of 103(8).

```
>INS MAC/INC=4096./PAR=SYSCTL
```

Install task MAC in the system-controlled partition SYSCTL and increase the task size by 4096 decimal words.

Command Error Messages:

INS -- BASE ADDRESS MUST BE ON 4K BOUNDARY

The base virtual address of the task is not on a 4K boundary. Applicable only to mapped systems.

INS -- BASE MISMATCH COMMON BLOCK <common-name>

The base address of the partition does not match that of the common block. Applicable only to unmapped systems.

INS -- CHECKPOINT AREA TOO SMALL

The area allocated for checkpointing the task is smaller than the partition into which it is being installed.

INS -- COMMON BLOCK IS TASK PARTITION <common-name>

A task's request for access to a common block is rejected because the partition requested is a task partition.

INS -- COMMON BLOCK NOT LOADED <common-name>

The specified common block is linked to the task but has not been installed into the system. Install the common block, then install the task.

INS -- COMMON BLOCK OCCUPIED

An attempt has been made to load a common block which is already occupied.

INS -- COMMON BLOCK PARAMETER MISMATCH <common-name>

Parameters of a common block do not match those in the task's label block.

INS -- FILE NOT CONTIGUOUS

An attempt is being made to install a task from a non-contiguous file. Task images may exist only in contiguous files.

INS -- FILE NOT TASK IMAGE

Data in the label block is not correct, indicating that the file is not a task image.

INS -- ILLEGAL DEVICE ddnn:

Device specified at task build by the assign keyword options does not exist in the system. This is a warning message; the task will be installed.

INS -- ILLEGAL DEVICE/VOLUME

The device specified is not a valid task residence device.

INS -- ILLEGAL FIRST APR

A privileged task, built to run using APR 4, 5, or 6 as its base, is too large. Using APR4, the task is larger than 12K; using APR5, the task is larger than 8K; using APR6, the task is larger than 4K. Applies only to mapped systems.

INS -- ILLEGAL PRIORITY

The value of the priority specified in the command is out of range (i.e., not 1 to 250(10), inclusive).

INS -- LENGTH MISMATCH COMMON BLOCK <blockname>

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. A task's label block data must match system data for that task before it can be installed.

INS -- NOT ENOUGH APRS FOR TASK IMAGE

The Task Builder allows the user to specify the virtual base address of a task image as a multiple of 4K. Privileged tasks start at virtual 100000(8) to be able to map the first 16K of the Executive at the same time as the user task. If the virtual base address is set too high, the task image may not be able to be mapped with the remaining mapping registers. Applicable to mapped systems only.

INS -- NO ROOM AVAILABLE IN STD FOR NEW TASK

This message means that there is no dynamic memory available to build the system task directory (STD) entry (task control block); therefore, no task can be installed.

INS -- PARTITION NOT COMMON

A partition specified for a common library is not defined as a common partition.

INS -- PRIVILEGED TASK LARGER THAN 12K

Privileged tasks have a maximum size of 12K.

INS -- SPECIFIED PARTITION FOR COMMON BLOCK

A task is being installed into a common block.

INS -- SPECIFIED PARTITION TOO SMALL

The task being installed is larger than the partition into which it is being installed.

INS -- TASK AND PARTITION BASES MISMATCH

The base of the partition does not match that of the task being installed. Applies only to unmapped systems.

INS -- TASK IMAGE I/O ERROR

Install cannot read the task image file or it is unable to rewrite the task image header. (Device is write locked.)

INS -- TASK IMAGE CURRENTLY INSTALLED

The requested task image, which is checkpointable, is already installed. Checkpointable tasks may be singly installed; other tasks may be multiply installed.

INS -- TASK NAME ALREADY IN USE

An attempt has been made to install a task with the same name as one already in the system.

INS -- TOO MANY COMMON BLOCK REQUESTS

A task is limited to three common block references.

INS -- TOO MANY LUNS

A task has requested more than 255(10) LUNs to be assigned.

INS -- UNDEFINED COMMON BLOCK <blockname>

A task references a common block that is not defined in the system. Usually, this indicates that the task was built for another system.

Notes:

1. Once a task is installed, its physical location is known to the Executive by virtue of the fact that its TCB contains the address of the load device unit control block and the logical block number of the image. Thus, to load the task, no searching is needed, and the task can be read in one transfer. However, the user is not prevented from deleting the disk file that the task resides in. If the file is deleted, subsequent file system activity could reallocate the disk blocks to another file. In this case, the task will probably abort on task load.

MOUNT (privileged)

MOU

The Mount command logically connects devices to ancillary control processors (ACPs). For Files-11 volumes, the Mount command creates the Volume Control Block (VCB) and declares that the volume is logically on-line for access by the file system. The VCB is allocated in dynamic memory and controls access to the volume. Details are specified in the RSX-11 I/O Operations Reference Manual. For network channels, the Mount command informs the network ACP that the device is available for use in the network. It allocates a communication Volume Control Block and issues a Mount request to the ACP.

Format:

```
MOU[NT] dev:[volume-label][/keyword][/keyword]...
```

Keywords:

```
/ACP    = taskname
/EXT    = block-count*
/FPRO   = [system,owner,group,world]*
/LRU    = least recently used directory count*
/PRTCL  = network protocol name**
/RCK    = redundancy check algorithm name**
/TEL    = telephone number for dial up link**
/UIC    = user id code*
/UNL      (no value is associated with UNL)*
/VI       (no value associated with VI)*
/WIN    = retrieval-pointer-count*
```

Mount keywords for Files-11 devices override corresponding values in the volume's home block.

where:

dev: - Device-unit on which the volume is to be mounted.

Default: none; must be specified.

volume-label - The Files-11 volume label may be up to twelve characters in length. It is used to ensure that the correct volume is being mounted. A volume label may not be specified for network channels.

Default: twelve nulls; no label check is performed.

ACP - The task name of an Ancillary Control Processor designed to process this volume.

* Keyword valid for Files-11 devices only.
** Keyword valid for network devices only.

Default: depends on device type, as indicated below:

For Files-11 volumes, the ACP name is F11ACP
For network channels, the ACP name is NT....

EXT - The number of blocks by which a file which exhausts its space allocation is to be extended.

Default: taken from the home block.

FPRO - Keyword for default file protection. Access codes consist of four, 4-code groups in the Access Rights word, as follows:

R = Read
W = Write
E = Extend
D = Delete

In each instance, the absence of the code means that the access right is denied the user. The square brackets are required syntax.

Default: values taken from volume home block.

NOTE

Protection code subparameters (system,owner,group,world) are positional. That is, the location of the word in the parameter string defines the user to whom the code applies. The order is:

system,owner,group,world

The order of appearance of the characters R, W, E, and D is fixed.

R, if desired, must be first.
W, if desired, is next.
E, if desired, is next.
D, if desired, is last.

Thus, RWE and RE are acceptable, while WR and DEWR are not.

LRU - The number of directory File Control Blocks (FCB) kept in memory by the ACP per volume. The more FCB's kept in memory, the faster a file can be found. If the file is in a directory for which the FCB is resident, the overhead required to open the directory is bypassed and thus the file is found more quickly.

Default: 3.

PRTCL - The name of the network line protocol.

Default: DDCMP

RCK - The name of the redundancy checking routine.

Default: CRC16

TEL - The telephone number of a line to a remote node. Up to 12 digits may be specified.

Default: null (Note: dial-out not supported in DECNET Release 1)

UIC - User identification code - [group,member].

UNL - Specify the volume index file as unlocked. When locked, the index file is read only; when unlocked, it is read/write.

Default: locked.

VI - Display the volume information on the entering terminal. The information displayed is that under which the volume was mounted. The volume information format is:

```
DEVICE = ddnn:
TYPE   = type
LABEL  = volume label
UIC    = [nnn,nnn]
ACCESS = [system,owner,group,world]
CHARAC = [ ]
```

WIN - The number of mapping pointers to be allocated for file windows.

Default: taken from home block.

Examples:

```
>MOU DK1:
```

Mount the volume on device-unit DK1:. (In this example, the optional parameters have not been specified; therefore, the parameter values in the volume's home block are used.) No label checking is performed. No volume information is displayed.

```
>MOU DK1:SYS004
```

Mount the volume labeled SYS004 on DK1:

```
>MOU XP:/TEL=16176468600
```

Mount the DP11 link and dial the remote node whose telephone number is 16176468600.

Command Error Messages:

MOU -- ACP NOT IN SYSTEM

Task specified as ACP or default ACP is not installed in the system.

MOU -- ALREADY MOUNTED

The network device-unit specified has already been mounted.

MOU -- DEVICE ATTACHED

The specified device-unit is currently attached by a task and may not be mounted.

MOU -- DEVICE NOT DEFINED IN NETWORK

The specified device-unit has not been defined in the current network topology and therefore may not be mounted.

MOU -- DEVICE OFFLINE

The specified device, although generated into the system, is not physically present in the host configuration.

MOU -- FILE HEADER READ ERROR

Mount cannot read either the index file header or the storage allocation file.

MOU -- HOME BLOCK READ ERROR

An I/O error was detected in trying to read the home block. This usually indicates that the volume is not ready. Wait until it is ready and reissue the command.

MOU -- NT DEVICE NOT MOUNTED

The device NT: must always be mounted before any other device in a network.

MOU -- NOT MOUNTABLE DEVICE

The device specified is not supported as a Files-11 device or a network device.

MOU -- OTHER VOLUME MOUNTED

An attempt has been made to mount a volume on a device which already has a mounted volume.

MOU -- RETRIEVAL POINTERS WRONG FORMAT

The index file retrieval pointers are not in correct format for RSX-11M.

MOU -- STORAGE BIT MAP FILE READ ERROR

An I/O error was encountered while reading the storage allocation file.

MOU -- TASK NOT ACP

The task specified as an ACP does not have the characteristics of an ACP.

MOU -- WRONG VOLUME

The volume label and the label specified in the command do not match.

MOU -- VOLUME STRUCTURE NOT SUPPORTED

RSX-11M does not support the Files-11 structure level of the volume being mounted.

SET (some privileged options) | SET |

The Set command makes it possible to dynamically alter a number of system-wide parameters. Some of the options are privileged. In particular, SET enables the user to:

1. Establish the default buffer size for a device;*
2. Establish terminal device characteristics;
3. Establish a terminal as privileged;*
4. Characterize a terminal as a slave terminal (a terminal from which the system will not accept unsolicited input);
5. Establish a default UIC for a terminal;**
6. Create partitions or subpartitions;*
7. Add space to the dynamic core pool, and*
8. Cause all writes to an RK05 disk to be automatically followed by a write check to ensure that the data was written correctly.

Format:

SET /Keyword=values

Only one keyword per command is permitted. All of the keywords except BUF, POOL, and UIC may be prefixed by NO to negate the effect of the keyword. Specifying NO on BUF, POOL, or UIC produces a syntax error.

Keywords for setting device characteristics are:

```
/BUF    = dev:[size]
/LA30S  [= dev:]
/LOWER  [= dev:]
/PRIV   [= dev:]
/SLAVE  [= dev:]
/SPEED  = dev:[recv:xmit]
/UIC    [= [group,member][:]dev:]]
/UIC    [=dev:]
/VT05B  [= dev:]
```

* Privileged terminal status required.
** Privileged terminal status required to change default UIC for other terminal.

The effects of these keywords are as follows:

BUF - Set or display the default buffer size of the specified device. If the size is not entered, this keyword results in the current size being displayed. BUF is particularly useful for defining line printer width (80 or 132 columns).

LA30S - Establishes the specified terminal device as an LA30S. If dev: is omitted, all terminals which are defined as LA30S are displayed.

NOLA30S - Resets the terminal to other than an LA30S. If dev: is omitted, all non-LA30S terminals will be displayed.

LOWER - Specifies that lower-case characters are not to be converted to upper case when received from the specified terminal. If dev: is omitted, all terminals for which lower-case characters are not converted are displayed.

NOLOWER - Resets the terminal characteristics so that lower-case characters received on input will be automatically converted to upper case and echoed as upper case. Absence of dev: results in the display of all terminals for which conversion to upper case is specified.

PRIV - Sets the specified terminal to privileged status. If dev: is omitted, MCR will display all privileged terminals.

NOPRIV - Sets the identified terminal to non-privileged status. Absence of dev: results in a display of all non-privileged terminals.

SLAVE - Establishes the specified terminal as one which can only enter data if it is solicited from a task; thus, unsolicited input is always rejected. Absence of dev: will result in the display of all terminals currently classified as slaves.

NOSLAVE - Resets slave status. Absence of dev: results in the displays of all non-slave terminals.

SPEED - Establishes the receive and transmit baud rate for terminals attached to the system through a DH11 multiplexer. The argument "recv" is the receive baud rate for the DH11, and "xmit" is the transmit baud rate for the DH11. Both must be specified in setting the speed. If neither is specified, the current settings will be displayed. Valid baud rates are: 0, 110, 150, 200, 300, 600, 1200, 1800, 2400, 4800, 9600, A (user-specified baud rate A), and B (user-specified baud rate B).

VT05B - Sets the specified terminal as a VT05B (a 2400 baud VT05). If dev: is omitted, all terminals classified as VT05B will be displayed.

NOVT05B - Resets the terminal to other than a VT05B. If dev: is omitted, all terminals classified as non-VT05B terminals will be displayed.

UIC - The specified UIC is established as the default, which is used implicitly for the entering terminal unless dev: is specified, in which case, the default UIC is set for that terminal. Setting the UIC explicitly is a privileged command. All tasks run from the terminal are run with the UIC of the terminal, unless explicitly overridden with the UIC switch in the Run command. Further, external* MCR function tasks are requested with this UIC. If a UIC is not specified, the current UIC will be displayed. If only dev: is specified, the UIC for the specified terminal will be displayed, provided that the terminal from which the command was issued is privileged.

Notes:

1. The status of all terminals may be displayed at any terminal.

2. Privileged status may only be altered from a privileged terminal.

3. LA30S, SLAVE, and VT05B status may be altered from a privileged terminal, or the requesting terminal may alter its own status.

4. Buffer size may be altered only from a privileged terminal.

5. The UIC of a terminal may be altered from the same terminal. The UIC of any terminal may be altered from a privileged terminal.

6. Improper usage of the Set command from a privileged terminal can result in a situation in which privileged operator input will not be accepted. There are two situations which must be avoided, arising when one of the following conditions is in effect:

   a. There are no non-slave, privileged terminals in the system. Non-privileged terminals cannot enter privileged commands. Slaved terminals cannot enter unsolicited input to MCR.

   b. The buffer length of all non-slave, privileged terminals is shorter than 14 bytes. The terminal buffer is too short to enter many important commands required to continue system operation or to alter the status of other terminals.

* These are the MCR Mount, Install, Dismount, and UFD commands (which run as tasks) and all system-supplied software.

Keywords for altering memory allocation are:

```
/MAIN    = pname[:base:size:type]

/NOMAIN  = pname

/SUB     = mname:pname[:base:size]

/NOSUB   = mname:pname

/POOL    [=top]
```

These keywords are used as follows:

MAIN - This keyword is used to establish a main partition.

pname = The 1- to 6-character alphanumeric partition name.

:base = Physical base address of the partition specified as a number of 64-byte blocks.

:size = Size of the partition specified as a number of 64-byte blocks.

:type = TASK for a user-controlled task partition: SYS for a system-controlled task partition, COM for a common partition, or DEV for a common partition mapping into the device registers. User-controlled and system-controlled task partitions are allocated for user tasks. Common partitions are used for resident libraries and common blocks. Device partitions are used by non-privileged tasks to access device registers in the external page, such as those for the UDC and ICS/ICR-11.

Base and size may be entered in any of the following formats; nnnn is a number to which a multiplication factor is applied to determine the value used.

Format		Calculated Value
nnnn	-	Octal (nnnn*100)
nnnn.	-	Decimal (nnnn.*64.)
nnnK	-	Octal K (nnn*4000)
nnn.K	-	Decimal K (nnn.*2048.)

Valid ranges of calculated values are:

Octal	0<=base<177777	0<size<2000
Decimal	0.<=base<65535	0.<size<1024.
Octal K	0K<=base<3777K	0K<size<40K
Decimal K	0.K<=base<2047.K	0.K<size<32.K

For system-controlled partitions, size may range from 0<size<1024.K or any valid representation of these limits.

Example:

Any of the following inputs for size will allocate a 2048. byte partition.

```
40
32.
1K
1.K
```

If base, size, and type are not specified, MCR displays the values of these parameters for the named partition. If the named partition is a main partition, the parameters of all its subpartitions will also be displayed.

NOMAIN - The NOMAIN keyword eliminates the named partition from the system.

SUB - This keyword is used to establish a subpartition. Subpartitions may only be established in user-controlled main partitions.

mname = The 1- to 6-character main partition name. The subpartition being defined will become a subpartition in the named main partition.

:pname = The 1- to 6-character subpartition name. The subpartition being defined is a subpartition of the main partition specified in mname.

:base = Physical base address of the subpartition, specified as a number of 64-byte blocks.

:size = Size of the subpartition, specified as a number of 64-byte blocks.

Base and size may be entered in any of the following formats; nnnn is a number to which a multiplication factor is applied to determine the value used.

Format		Calculated Value
nnnn	-	Octal (nnnn*100)
nnnn.	-	Decimal (nnnn.*64.)
nnnK	-	Octal K (nnn*4000)
nnn.K	-	Decimal K (nnn.*2048.)

Valid ranges of values are:

Octal	0<=base<10000	0<size<2000
Decimal	0.<=base<4096.	0.<size<1024.
Octal K	0K<=base<200K	0K<size<40K
Decimal K	0.K<=base<128.K	0.K<size<32.K

Example:

Any of the following inputs for size will allocate a 2048. byte partition.

```
40
32.
1K
1.K
```

If base and size are omitted, the current values will be displayed for the named subpartition.

NOSUB - Eliminates the named subpartition from the system.

POOL - This keyword is used to increase the size of the dynamic storage region.

top - First location in memory to be used for user partitions. This is given in number of 64-byte blocks. It can be given in octal or decimal. If top is not given, the system will print out the current top of memory, the size of the longest block of pool space in words, and the total number of words in the pool.

The format is:

```
POOL = top:max:total
```

where top is given in units of 64-byte blocks. Max and total are in decimal words.

Keyword for specifying write check on writes to RK05 disks:

```
/WCHK[=dev:]
/NOWCHK[=DEV:]
```

WCHK - Specifies the RK05 disk which is to have all writes followed by a write check. This feature provides high reliability for data transfers to the RK05 disk. If dev: is omitted, all RK05 disk drives with write check enabled will be displayed.

NOWCHK - Specifies that write check is to be disabled for the specified device. If dev: is not specified, the RK05 disks which have write check disabled will be displayed.

Command Error Messages:

SET -- ALIGNMENT ERROR

An attempt has been made to create a partition but the base address or size conflicts with existing partitions or physical memory size.

SET -- DEVICE NOT DH11

An attempt has been made to set the baud rate for a terminal which is not attached to a DH11 multiplexer.

SET -- DEVICE NOT TERMINAL

An attempt has been made to set terminal characteristics for a non-terminal device.

SET -- NON EXISTENT MEMORY

An attempt has been made to define a partition in non-existent memory.

SET -- PARTITION ALREADY EXISTS

An attempt has been made to define a partition with a name that is already in use.

SET -- SPACE USED

An attempt has been made to create a partition or subpartition in a storage area already occupied.

SET -- TASK INSTALLED IN PARTITION

An attempt has been made to eliminate a main partition or subpartition containing installed tasks.

SET -- TOO MANY SUBPARTITIONS

A main partition is limited to a maximum of seven subpartitions.

SET -- UNKNOWN MAIN PARTITION

An attempt has been made to define a subpartition of a non-existent main partition.

SET -- WRITE CHECK NOT SUPPORTED

An attempt has been made to enable write check on a device which is not an RK05. Write check is currently supported for RK05 disks only.

Notes:

1. If a task is currently installed in a partition and an attempt is made to eliminate the partition (/NOMAIN, /NOSUB), the command will be rejected.

2. If a main partition is eliminated, all its subpartitions are also eliminated.

3. If a subpartition is being eliminated, only the specified subpartition will be eliminated. A main partition that has subpartitions with tasks installed will not be allowed to be eliminated until those tasks are removed.

4. When defining a partition (main or sub) the name must not already be defined as a partition. In addition, main partitions may not overlap any other main partition, nor may subpartitions of a main partition overlap any other subpartitions of the main partition.

5. The numeric value convention applies to all the Set command numeric values.

6. All partitions, except LDR, are considered user partitions by the POOL command. Thus, SYSPAR is included as a user partition.

7. Once space has been allocated to the dynamic storage region, it can never be recovered for use in partitions.

8. System-controlled partitions may not have subpartitions explicitly defined by the Set command. An attempt to do so will be rejected. The Executive defines subpartitions of the system-controlled partition dynamically as needed for tasks installed in the system-controlled partition.

Examples:

```
>SET /BUF=LP0:
BUF=LP0:132.
```

Display the current buffer size of LP0:

```
>SET /BUF=TT1:40.
```

Set TT1: buffer size to 40(10).

```
>SET /LA30S=TT2:
>SET /LA30S
LA30S=TT0:
LA30S=TT2:
```

Set TT2: as an LA30S.
Display all LA30S terminals.

```
>SET /NOLA30S=TT0:
```

Set TT0: to other than an LA30S terminal.

```
>SET /PRIV=TT0:
```

Set TT0: to a privileged terminal.

```
>SET /PRIV
PRIV=TT0:
PRIV=TT1:
```

Display all privileged terminals.

```
>SET /SLAVE=TT3:
```

Set TT3: to a slave terminal.

```
>SET /VT05B
VT05B=TT4:
VT05B=TT5:
VT05B=TT6:
```

Display all VT05B terminals.

```
>SET /MAIN=SYSPAR:420:140:TASK
```

Define a main partition called SYSPAR as a user-controlled partition whose base address is 42000(8) and whose length is 14000(8) bytes.

```
>SET /SUB=SYSPAR:MCRPAR:445:56
```

Define a subpartition called MCRPAR of main partition SYSPAR whose base address is 44500(8) and whose length is 5600(8) bytes.

```
>SET /MAIN=SYSPAR
MAIN=SYSPAR:0420:0140:TASK
SUB=SYSPAR:MCRPAR:0445:0056
```

Display the parameters of the main partition SYSPAR and all of its subpartitions.

```
>SET /NOSUB=SYSPAR:MCRPAR
```

Eliminate subpartition MCRPAR.

```
>SET /NOMAIN=SYSPAR
```

Eliminate main partition SYSPAR.

```
SET /MAIN=GEN:1000:6000:SYS
```

Define a main partition called GEN as a system-controlled partition whose base address is 100000(8) and whose length is 600000(8) bytes.

```
>SET /POOL=420
```

Set the beginning of user partitions to 42000(10).

```
>SET /POOL

POOL=0420:150:420.
```

The first location for user partitions is 42000; the longest free block is 150. words, and the total of all pool space is 420. words.

```
SET /WCHK=DK1:
```

Enable write check on DK1:.

TIME (some privileged options)

TIM

The Time command makes it possible to:

Set the current time of day (privileged);

Set the current date (privileged), or

Display the current time and date.

Format:

TIM[E] [hrs:mins[:secs]] [mnth/day/year]

where:

hrs = Hours (range 0-23)

mins = Minutes (range 0-59)

secs = Seconds (range 0-59) (optional)

mnth = Month (range 1-12)

day = Day (range 1-31)

year = Year (range 0-99). Year is relative to 1900; thus, 1975 is entered as 75.

Examples:

```
>TIM
10:23:31 03-May-75
```

Display the current time and date.

```
>TIM 1:30 4/7/75
```

Set the time to 01:30:00 and the date to 07-Apr-75

Notes:

1. If neither time nor date is specified, the current time and date will be displayed on the entering terminal.

2. If the time and date are specified, the clock and calendar will be set; if only the time is specified, the clock will be set; if only the date is specified, the calendar will be set.

3. The order of time and date in the command is not significant. Either order will be accepted.

4. All numeric values are decimal. No terminating period is necessary (or permitted).

USER FILE DIRECTORY

UFD

This MCR command creates a user file directory file (UFD) on a Files-11 volume and enters its name into the master file directory file (MFD). Each user or group of users desiring private protection and file access by name requires a UFD. The volume must have been subjected to Initvolume and Mount commands before UFDs may be defined. Once a volume has been initialized and mounted, UFDs may be added at any time.

Format:

```
UFD dev:[volume-label][group,member][/Keyword][/Keyword]
```

Keywords:

```
/ALLOC = number-of-entries

/PRO   = [system,owner,group,world]
```

where:

dev: - Device-unit containing the volume on which the UFD being created will reside.

Default: none; must be specified.

volume-label - If specified, the label is compared with the label on the volume. If the names match, a UFD can be allocated. If they do not match, the command is rejected.

DEFAULT: NULLS; NO CHECK IS MADE.

[group,member] - The UIC for the UFD. The UIC establishes the owner of the UFD. The square brackets are required syntax.

Default: none; must be specified.

ALLOC - Number of directory entries for which space is to be allocated. The number provided is rounded up to the next multiple of 32(10).

Default: 32(10)

PRO - Establishes access rights for the directory file. Access codes consist of four, 4-code groups in the Access Rights word, as follows:

R = Read
W = Write
E = Extend
D = Delete

In each instance, the absence of the code means that the access right is denied the user class.

Defaults: [RWED,RWED,RWED,RWED]

NOTE

Protection code subparameters (system,owner,group,world) are positional. That is, the location of the word in the parameter string defines the user to whom the code applies. The order is:

system,owner,group,world

The order of appearance of the characters R, W, E, and D is fixed.

R, if desired, must be first.
W, if desired, is next.
E, if desired, is next.
D, if desired, is last.

Thus, RWE and RE are acceptable, while WR and DEWR are not.

Example:

>UFD DK1:[1,1]

Create a user file directory for the UIC [1,1] on DK1:. The name and file number of the UFD file are entered into the MFD. The name of the UFD is 001001.DIR;1, and its owner is UIC [1,1].

Command Error Messages:

UFD -- CAN'T READ MCR COMMAND BUFFER

UFD was initiated by a Run command rather than as an MCR command.

UFD -- DIRECTORY ALREADY EXISTS

The requested UFD already exists on the volume.

UFD -- FAILED TO CREATE DIRECTORY

No space exists on the volume or an I/O error has occurred.

UFD -- FAILED TO ENTER IN MFD

No room in the MFD, or no room on the volume, or an I/O error has occurred on the volume.

UFD -- HOME BLOCK I/O ERROR

An error was detected in reading the volume home block.

UFD -- NOT FILES-11 DEVICE

The device on which the UFD is to be created is not a Files-11 device and therefore cannot support UFDs.

UFD -- WRITE ATTRIBUTES FAILURE

An error was encountered while writing the attributes of either the MFD or the newly created UFD.

UFD -- WRONG VOLUME

The volume label and the label specified in the command do not match.

UFD -- VOLUME NOT MOUNTED

Volume on which a UFD is to be created must be mounted before accessing the Files-11 structure.

Notes:

1. UFD is valid only for mounted volumes.

4.3.3 Informational Commands

DEVICES (privileged)

DEV

The Devices command displays the symbolic names of all device-units known to the system on the entering terminal.

Format:

```
DEV[ICES]
```

Example:

```
>DEV
DK0:    MOUNTED
DK1:    MOUNTED
DF0:    MARKED FOR DISMOUNT
DT1:
DT2:
DT3:    OFFLINE
LP0:
TT0:
TT1:
TI0:
CO0:    TT0:
CL0:    LP0:
SY0:    DK0:
```

Notes:

1. MOUNTED is an optional message to indicate that the device is mounted.

2. MARKED FOR DISMOUNT is an optional message to indicate that a mountable device has been requested to be dismounted, but the dismount operation has not yet been completed.

3. OFFLINE is an optional message which indicates that, although the system tables contain entries for this device, the host configuration does not contain the related device.

4. A device name in the second column is the device to which the corresponding device in the first column has been redirected.

LOGICAL UNIT NUMBERS (privileged)

LUN

The Logical Unit Numbers command is used when the operator needs to determine which physical devices a task has statically assigned. A list of physical device units and corresponding logical unit numbers is displayed on the entering terminal. It indicates which devices were assigned to the specified task.

Format:

LUN[S] taskname

Example:

```
>LUN XKE
TI0:  1
TI0:  2
CL0:  3
TT3:  4
TT3:  5
DK0:  6
TT3:  7
```

The display shows static assignments for LUNs 3, 4, 5, 6, and 7, as recorded in the task image file header; no other LUNs are statically assigned to the task XKE.

Command Error Messages:

LUN -- NO LUNS

The task which is the argument of the LUNs command does not have any logical units. This is not an error message, but rather, an indication that there are no assignments to display.

Notes:

1. After a task is in operation, the LUNs command display does not necessarily reflect the actual assignments in effect. Assign LUN Exucutive directives issued by the task may have changed the static assignments.

PARTITION DEFINITIONS (privileged)

PAR

The Partition Definitions command displays on the entering terminal a description of each memory partition in the system.

The following is displayed:

1. Partition name;
2. Partition base address (octal);
3. Partition size (octal);
4. Partition kind, main partition or subpartition, and
5. Partition type, TASK (user-controlled), COM (common), DEV (device registers), or SYS (system-controlled).

Format:

PAR[TITIONS]

Example:

```
>PAR
LDR     023544  001000  MAIN  TASK
SYSPAR  030100  007700  MAIN  TASK
TKNPAR  030100  000400  SUB   TASK
GEN     040000  100000  MAIN  SYS
COMPAR  140000  020000  MAIN  COM
DEVPAR  160000  020000  MAIN  DEV
```

Notes:

1. The display contains the name, base, size, kind, and type of partition in columns 1 through 5, respectively.

2. Partitions of type TASK are user-controlled task partitions. COM partitions are used for re-entrant libraries and common data areas. DEV partitions are used to allow tasks to communicate with specific device registers, such as the UDC and ICS/ICR-11 industrial control subsystems. SYS partitions are system-controlled task partitions.

TASKLIST (privileged)

TAS

The Task List command displays a description of each task on the pseudo device CL:. The display contains:

1. Task name;
2. Task version identification;
3. Partition name;
4. Task priority;
5. Size of task in bytes (octal);
6. Load device identification;
7. Disk address logical block number (octal), and
8. Task memory state.

Format:

```
TAS[KLIST]
```

Example:

```
>TAS
. LDR.  0001M LDR     248. 010600 DK0: -00000010572
.TKTN.  0001M GEN     249. 002200 DK0: -00000011112  FIXED
...MCR  0001M MCR     221. 002200 DK0: -00000010511  CHECKPOINTED
...MOU  0001M PAR3K   221. 001400 DK0: -00000011255
```

Notes:

1. The display contains, in columnar form, from left to right: task name, task version identification, partition name, task priority, task size, load device identification, logical block number on the load device, and task memory state.

2. FIXED indicates that the task is fixed in memory.

3. CHECKPOINTED indicates that the task has been swapped out of memory to make room for a higher priority task.

4.3.4 Task Control Commands

ABORT (some options privileged) — ABO

The Abort command allows a console operator to terminate the execution of an indicated task. In the example below, the task, SCAN, is terminated immediately.

Format:

```
ABO[RT] taskname
```

Example:

```
>ABORT SCAN
TASK "SCAN  " TERMINATED
ABORTED VIA DIRECTED or MCR

     PC=076400
     PS=000030
     R0=000000
     R1=004230
     R2=074350
     R3=000020
     R4=000000
     R5=076000
     SP=076114
```

Command Error Messages:

When Abort is issued, MCR will cause the Task Termination Notification routine (TKTN) to run. It is TKTN that produces the termination display shown in the above example. Thus, messages following an Abort command may come from the command itself or from TKTN. These two possible message origins are presented separately below.

Message From Abort command:

ABO -- TASK NOT ACTIVE

The specified task is not currently active.

Messages from Task Termination Notification Routine (TKTN):

The Task Termination Notification routine is used to display all task aborts, whether from the MCR Abort command or some other source. The display consists of:

```
TASK "<taskname>" TERMINATED
<abort cause>
```

The display of the abort cause is followed by a list of the task's registers at the time of the abort. Listed below are the abort causes.

Abort Cause Messages:

ABORTED BY DIRECTIVE or MCR

Either MCR or an Executive directive issued by another task has caused the task to be aborted.

11 40 F.P. EXCEPTION

The task encountered a floating point exception while executing on a PDP11/40 and no SST routine was specified to process the trap.

AST ABORT. BAD STACK

AN AST cannot be effected because the AST parameters cannot be pushed onto the task's stack.

CHECKPOINT FAILURE. READ ERROR.

Task could not be read back into memory after being checkpointed.

IOT EXECUTION

The task executed an IOT instruction and no SST routine was specified to process the trap.

LOAD FAILURE. READ ERROR

Task could not be loaded.

MEMORY PROTECT VIOLATION

The task encountered a memory protect violation and no SST routine was specified to process the trap.

NON RSX EMT EXECUTION

The task executed an EMT instruction with an argument other than 377(8) and no SST routine was specified to process the trap.

ODD ADDRESS OR OTHER TRAP FOUR

The task executed a word instruction with an odd address or referenced a non-existent memory location in an unmapped system and no SST routine was specified to process the trap.

PARITY ERROR

A parity error occurred while executing the task. The task is fixed in memory so that the memory will not be re-used by another task.

RESERVED INST EXECUTION

The task has executed an illegal instruction and no SST routine was specified to process the trap.

SST ABORT. BAD STACK

An SST cannot be effected because the SST parameters cannot be pushed onto the task's stack, or a stack overflow has been detected in an unmapped system, as indicated by a nonzero value in the header guard word.

TASK EXIT WITH OUTSTANDING IO

Tasks should terminate all I/O operations before exiting. The system does, however, clean-up all outstanding I/O.

T BIT TRAP OR BPT EXECUTION

The task has either set the T bit in the Processor Status Word or executed a Breakpoint Trap instruction and no SST routine was specified to process the trap.

TRAP EXECUTION

The task has executed a Trap instruction and no SST routine was specified to process the trap.

Notes:

1. Aborting a task causes an orderly, but forced, termination of the specified task. It:

   a. Performs I/O rundown; I/O for all non-file-structured devices is cancelled. I/O for file-stuctured devices is allowed to complete and then the files are deaccessed. All attached devices are detached. I/O rundown may take a considerable amount of time for tasks connected to a network node.

   b. Requests the Task Termination Notification routine (TKTN) to display a message on the aborted task's TI: terminal. The display describes the reason for the termination and the task's registers at termination. If the task is checkpointed at the time TKTN is called to process the abort, the registers will not be printed. For tasks whose names begin with "..." and which were aborted via the ABO command, the registers will not be displayed.

   c. Releases the task's partition if the task is not fixed.

2. An Abort command directed to a task not initiated from the entering terminal may only be issued from privileged terminals.

ALTER (privileged)

ALT

This command allows the operator to change the priority of an installed task.

Format:

ALT[ER] taskname /keyword=value

Keyword:

/PRI = priority

where:

taskname - Name of task whose priority is to be altered.

priority - New priority to be assigned to the specified task.

Example:

>ALT TEST /PRI=250

Alter the priority of task TEST to 250(8).

Command Error Messages:

ALT -- TASK BEING FIXED

An attempt was made to change the priority of a task which is in the process of being fixed.

Notes:

1. A task that is active (not dormant) or being fixed in memory cannot have its priority altered.

2. Priority must be less than or equal to 250(10) and greater than 0. The numeric conventions apply.

CANCEL (some privileged options) | CAN |

The Cancel command allows the operator to cancel time-based initiation requests for a task. These requests result from a Run directive or any of the time-synchronized variations of the MCR Run command.

Format:

CAN[CEL] taskname

Example:

>CAN XKE

Cancel all periodic rescheduling and time-based initiation requests for task XKE.

Notes:

1. A Cancel command directed to a task not initiated from the entering terminal may only be issued from privileged terminals.

2. Cancel does not affect the current execution of the task if it is active. Only the time-based schedule requests still in the queue are removed.

FIX-IN-MEMORY (privileged)

FIX

The Fix-in-Memory command allows the operator to force a task to be loaded and locked into its partition. Subsequent requests for the task to be run will be serviced more quickly, since the task will be memory resident and will not have to wait for loading from the disk.

Format:

FIX taskname

Example:

>FIX XKE

FIX task XKE in its partition.

Command Error Messages:

FIX -- PARTITION BUSY

Partition in which task is to be fixed is currently occupied, so task cannot be fixed.

FIX -- TASK BEING FIXED

The subject task is already in the process of being fixed.

FIX -- TASK ALREADY FIXED

The specified task is already fixed in memory.

FIX -- TASK CHECKPOINTABLE

A checkpointable task cannot be fixed in memory.

Notes:

1. Fixed tasks remain physically in memory even after they exit, and, thus, they need not be reloaded when requested to run. Their ability to respond to realtime demands is, therefore, increased. Only an Unfix command can place the task in a state which will permit subsequent freeing of the occupied memory.

2. A task is fixed upon availability of the partition in which it is to be fixed. Thus, the task may not be fixed until the partition is available.

3. Checkpointable tasks are not fixable.

4. Active tasks may not be fixed.

5. A task must be installed before it can be fixed.

REASSIGN (privileged) REA

The Reassign command allows the operator to reassign a task's Logical Unit Numbers (LUNs) from one physical device-unit to another. The reassignments are made in the disk image only.

Format:

REA[SSIGN] taskname lun nud:

where:

lun = Logical unit number
nud = New device-unit.

Examples:

>REA JOE 3 TT0:

Reassign LUN 3 of task JOE to device TT0:.

>REA BILL 2 TT1:

Reassign LUN 2 of task BILL to device TT1:.

Command Error Messages:

REA -- LUN OUT OF RANGE

An attempt has been made to reassign a LUN which is greater than the maximum number of LUNs allocated during task build.

Notes:

1. Reassign performs re-assignments in the indicated task's disk image; only the static device assignments are altered.

2. Reassign does not affect a memory-resident task. Hence, Reassign has no effect on the assignments of a currently executing task, nor on a task that is fixed in memory. The Redirect command (see next page) is used for dynamic device reconfiguration.

REDIRECT (privileged) RED

The Redirect command allows the operator to redirect all I/O requests directed to a physical device-unit to another physical device-unit. The Redirect command is especially useful if I/O units required for a task are inoperable.

Format:

RED[IRECT] nud:=old:

where:

nud: New device-unit; this is the redirected-to device;

= Equal sign notation for the redirect action; and,

old: Old device-unit; this is the redirected-from device.

Examples:

>RED TT3:=TT6:

Redirect all I/O requests for device TT6: to device TT3:.

>RED TT:=LP:

Redirect all I/O requests for device LP0: to device TT0:.

Command Error Messages:

RED -- CIRCULAR REDIRECT ERROR

An attempt has been made to redirect a device such that a circular list of redirects would result.

RED -- DEVICE NOT REDIRECTABLE

The specified old device may not be redirected.

RED -- NEW DEVICE NOT KNOWN TO SYSTEM

The new device in the Redirect command is not known to the system (does not exist in the device tables).

RED -- OLD DEVICE ATTACHED

An attempt has been made to redirect an attached device.

RED -- OLD DEVICE MOUNTED

An attempt has been made to redirect a mounted device.

RED -- OLD DEVICE NOT KNOWN TO SYSTEM

An attempt has been made to redirect an unknown device (the device does not exist in the device tables).

RED -- PSEUDO DEVICE REDIRECT ERROR

An attempt has been made to redirect a pseudo device to another pseudo device.

RED -- TI REDIRECT ERROR

The pseudo device TI: may not be redirected.

Notes:

1. Redirect does not affect any I/O requests already in the I/O queue.
2. The device TI: may not be redirected.
3. A pseudo device may not be redirected to another pseudo device.
4. A mounted volume may not be redirected.
5. An attached device may not be redirected.

REMOVE (privileged) REM

The Remove command allows the operator to delete a task name from the System Task Directory (STD), actually removing the task from the system. All that remains in the system is the task image file.

Format:

REM[OVE] taskname

Example:

>REM SCAN

Remove task named SCAN from the system. The task image file remains.

Notes:

1. The process of removing a task from the system makes a task unknown to the system. It is the complement of Install.
2. To remove a task that is in execution, first abort the task, then remove it.
3. If a task that is the object of a Remove command is fixed, it is automatically unfixed and then removed.
4. Remove automatically cancels all time-based schedule requests for the specified task.
5. Remove automatically unlinks a task from all ICS/ICR interrupts.

RESUME (some privileged options) | RES |

The Resume command allows the operator to continue the execution of a previously suspended task.

Format:

RES[UME] taskname

Example:

>RES XKE

Resume task XKE.

Command Error Messages:

RES -- TASK NOT SUSPENDED

The task used as the argument of the Resume command is not currently suspended.

Notes:

1. A task can only be suspended as a result of a Suspend directive issued by the running task to suspend itself; a task cannot suspend any task other than itself. The Resume command allows such a suspended task to proceed.

2. A Resume command directed to a task not initiated from the entering terminal may only be issued from privileged terminals.

RUN

RUN

Run is the command which initiates the execution of a task. With Run a task may be:

Started immediately;

Started a time increment from now;

Started a time increment from clock unit synchronization;

Started at an absolute time of day, or

Installed, immediately run, and removed on exit.

All of these options, except install-run-and-remove, are available with or without optional rescheduling.

Run has five format variations, as described below.

Formats:

1. Run Immediate.

   RUN taskname [/Keyword][/Keyword]

   Keywords:

   /RSI = nnnnu

   /UIC = [group,member]

   where:

   taskname = A 1- to 6-character taskname.

   RSI = The reschedule interval. The format is the same as for dtime detailed below under format 2. The reschedule interval specifies how often the task is to be rerun. Thus, each time the specified interval of time lapses, an initiation request is made for the specified task.

   Default: no rescheduling.

   UIC = Keyword for the UIC parameter. Legal group and member numbers range from 1 through 377(8). The square brackets are required syntax. This is the UIC under which the task will be requested. The UIC determines which files the task may access.

   Default: the value established for the terminal from which the Run command is entered.

2. Run a time increment from now.

   RUN taskname dtime [/Keyword][/Keyword]

Keywords:

/RSI = nnnnu

/UIC = [group,member]

where:

taskname = A 1- to 6-character taskname.

dtime = The time at which the task will be initiated in time units from command issuance. It is of the form nnnnu.

A time parameter consists of two fields:

a. A magnitude field, and

b. A units field.

where:

nnnn = The magnitude
u = Units, with:

T for ticks;
S for seconds;
M for minutes, and
H for hours.

The legal value of the magnitude is related to the value of the units field, which is encoded as:

T = Ticks. A tick is a clock interrupt and the rate at which interrupts occur depends on the type of clock installed on the system.

For a line frequency clock, the tick rate is either 50 or 60 per second, corresponding to the line frequency.

For a programmable clock, a maximum of 1000 ticks per second is available. (Frequency is selectable at system generation.)

S = Seconds.

M = Minutes.

H = Hours.

The magnitude is the number of units to be clocked, but the magnitude value cannot exceed 24 hours in the specified units.

Units = T. Any positive value is valid (maximum of 15 bits).

Units = S. Any positive value is valid (maximum of 15 bits).

Units = M. The maximum magnitude is 1440(10).

Units = H. The maximum magnitude is 24(10).

RSI = See Format 1, above.

UIC = See Format 1, above.

3. Run a time increment from clock unit synchronization.

The task start time is determined by first waiting for the next time unit's occurrence (hour, minute, second, tick) and then waiting for the specified time increment to elapse.

Run taskname sync [dtime][/Keyword][/Keyword]

Keywords:

/RSI = nnnnu

/UIC = [group,member]

where:

taskname = A 1- to 6-character taskname.

sync = H = synchronize on the next hour;
M = synchronize on the next minute;
S = synchronize on the next second, and
T = synchronize on the next tick.

dtime = As defined in Format 2, above; dtime, if present, is added to the synchronization unit to produce the actual run time for running the task.

RSI = See Format 1, above.

UIC = See Format 1, above.

4. Run at an absolute time of day.

RUN taskname atime [/Keyword][/Keyword]

Keywords:

/RSI = nnnnu

/UIC = [group,member]

where:

taskname = A 1- to 6-character taskname.

atime = Absolute time of day when this task will be initiated. Input format is hh:mm:ss.

where:

hh = hours (decimal assumed);
mm = minutes (decimal assumed), and
ss = seconds (decimal assumed).

RSI = See Format 1, above.

UIC = See Format 1, above.

5. Install, run immediately, and remove on exit.

When Run tries to start a task as a type 1 request, it searches the system task directory (STD) for the task; if it finds the entry, it proceeds to run it. However, if the task is not found, it will attempt to install a task whose filename is that given as the task name. The directory under which it looks for this file is determined by the presence or absence of a leading "$". If no "$" is found, it looks under the UIC of the terminal requesting the RUN. If "$" is present, it looks under the system directory ([1,50] for unmapped systems, and under [1,54] for mapped systems).

RUN [$]filename [/keyword][/keyword]...

Keywords:

/UIC = [group,member]

/PRI = Priority number. The value range is 1 through 250(10).

Default: 50. (decimal).

/PAR = Partition into which the task is to be installed.

Default: from task build.

/TASK = Task name. Name which is to be temporarily assigned to the task while it is running.

/INC = Number of words by which the task is to be increased. For system-controlled partitions only. (See description of Install command for more complete explanation.)

where:

$ = Optional - specifies that file is found under system UIC ([1,50] or [1,54]).

filename = Is the name of the task image file to be run.

Examples:

```
>RUN XKE 15M
```

RUN task XKE 15 minutes from command issuance.

```
>RUN XKE 15M/RSI=90S/UIC=[3,1]
```

RUN task XKE 15 minutes from command issuance, rescheduling it every 90 seconds, with a user identification code of [3,1].

```
>RUN $MAC
```

Run task in file MAC.TSK from the system directory. Task is automatically installed, run, and removed on exit.

Command Error Messages:

RUN -- INVALID TIME PARAMETER

A time field is incorrect.

RUN -- TASK BEING ABORTED

A request for the execution of a task has been made, but the subject task is currently being aborted either due to an unrecoverable error or as the result of an operator Abort directive.

Notes:

1. Formats 1 and 5, i.e., Run Immediate, will establish TI: as the terminal from which the task was initiated, if no /RSI is specified. Otherwise, the TI: terminal will be set to CO:.

2. A Run immediate, if specified without the /RSI option and terminated with an ESCape character instead of a carriage return, will echo the default prompt when the requested task exits and not when MCR has finished initiating the task. This provides a facility whereby an operator can determine that a task which produces no output on a terminal has exited.

3. The install, run, and remove form will establish the terminal from which the command was initiated to be the TI: terminal and will cause the default prompt to be displayed on TI: when the task exits. This form of the command is especially useful when the space available in the dynamic storage region is small.

UNFIX (privileged) UNF

The Unfix command allows the operator to free a fixed task from memory. The effect is to allow tasks waiting for the partition in which the task resides to compete for the partition. Unfix is the complement of the Fix command.

Format:

UNF[IX] taskname

Example:

>UNFIX XKE

Unfix task XKE, freeing the partition in which it resides for task competition.

Command Error Messages:

UNF -- TASK NOT FIXED

An attempt has been made to unfix a task which is not fixed.

Notes:

1. If a task exits or aborts, but is fixed, it will still remain in control of the physical memory in which it resides.

4.3.5 System Maintenance Commands

ACTIVE (privileged) | ACT |

This command will display the names of all active tasks on the requesting terminal. It is a quick way to determine the status of the system.

Format:

ACT[IVE]

Example:

```
>ACT
...MCR
F11ACP
...PIP
```

ACTIVE TASK LIST (privileged)

ATL

This command will display the name and status of all active tasks in the system on the entering terminal. It provides information useful for determining the exact status of each active task. The following information is displayed for each task:

1. Task name;
2. Task control block physical address (octal);
3. Partition name;
4. Partition base and limit physical addresses (octal);
5. Task status flags;
6. TI terminal physical device-unit;
7. I/O count (decimal);
8. Task local event flags, and
9. Task registers and Processor Status Word (memory resident tasks only).

The task status flags displayed are those from the task control block (TCB). This command will display the name of the bit if the bit is set to one at the time of the display. Zero bits are not displayed. Flag names are three characters long and correspond to the last three characters of the mnemonic defining the status bits. (See the RSX-11M System Generation Manual for a description of the TCB.) For example, FXD represents the bit TS.FXD and indicates that the task is fixed in memory. Names prefixed by a minus (-) sign indicate that the bit represents the complement of the condition. For example, -CHK indicates that the task is not checkpointable.

STATUS	TCB FLAG	DESCRIPTION
ABO	TS.ABO	Task is being aborted.
ACP	TS.ACP	Task is an Ancillary Control Processor.
AST	TS.AST	Task is processing an AST.
BFX	TS.BFX	Task is being fixed in memory.
-CHK	TS.CHK	Task is not checkpointable.
CKD	TS.CKD	Task checkpointing is disabled.
CKP	TS.CKP	Task is checkpointed.
CKR	TS.CKR	Task checkpoint request pending.
DST	TS.DST	Task ASTs are disabled.
-EXE	TS.EXE	Task is not in execution.
FXD	TS.FXD	Task is fixed in memory.
HLT	TS.HLT	Task is being terminated.
MCR	TS.MCR	Task was activated by MCR.
MSG	TS.MSG	Task aborted, waiting for TKTN message.
-PMD	TS.PMD	Task Post Mortem Dump disabled.
OUT	TS.OUT	Task is out of memory.
PRV	TS.PRV	Task is privileged.
RDN	TS.RDN	Task I/O is being run down.
REM	TS.REM	Task is to be removed on exit.
SPN	TS.SPN	Task is being suspended.
SPNA		Task was suspended prior to AST.
STP	TS.STP	Task stopped for terminal input.
WFR	TS.WFR	Task is in a "wait-for" state.
WFRA		Task was in a "wait-for" state prior to AST.

Format:

```
ATL
```

Example:

```
>ATL

...MCR 061300  SYSPAR 100000-107777 -PMD PRV
   TI - TT5:  IOC - 0.  EFLG - 000001 040000  PS - 170000 PC - 103520
   REGS 0-6  000106 000001 000004 050372 057066 061300 100372
F11ACP 062430  FCPPAR 110000-131777 -PMD ACP PRV WFR
   TI - CO0:  IOC - 0.  EFLG - 000002 040004  PS - 170000 PC - 100470
   REGS 0-6  062442 000000 000001 054054 053054 046212 100264
```

Notes:

1. If the task is not in memory (OUT flag is displayed), the contents of the PS, PC and registers will not be displayed.

BREAKPOINT TO EXECUTIVE DEBUGGING TOOL (privileged)

BRK

This command will pass control to the Executive Debugging Tool (XDT) if it has been generated into the system. If XDT is not in the system, this command is a no-op.

Format:

```
BRK
```

Example:

```
>BRK
```

Notes:

1. When XDT gains control, it will print the message

```
BE:  nnnnnn
XDT>
```

   to indicate that XDT now has control. All XDT commands are available for use in debugging the Executive or user-written drivers. To return control to MCR, enter the P command and MCR will prompt with ">".

OPEN REGISTER (privileged) | OPE

The Open command allows examination and optional modification of a word of memory. If a location within a task is opened, the task must be fixed in memory.

Format:

OPE[N] memory address [+ or -n][/Keyword]

(memory-address) (contents-of-address)/[value]line-terminator

The Open command, as shown above, is a 2-line command. The first line initiates the command. Its parameters are as follows:

memory-address	A 1- to 6-digit octal memory address.
+ or -n	One or more optional octal numbers to be added to or subtracted from the memory address.

Keywords:

/TASK = Taskname

/PAR = Partition name

/KNL (No value is associated with KNL)

TASK, PAR, and KNL are applicable only on mapped systems.

After accepting the first input line, Open finds the memory location to be accessed by either using:

1. The keyword in a mapped system; or
2. Using the specified address as the actual address in an unmapped system.

TASK results in the task partition being located; PAR results in the named partition being located; and KNL results in Executive memory (the first 32K of memory) being accessed. If no keyword is specified, the absolute address is opened.

When TASK and PAR are used, the supplied memory address (a virtual address in the partition) is modified by + or - n. The absolute address thus formed is used to display the second line shown in the command format. In particular, the virtual address (task, partition, or kernel) and the contents are displayed. The operator may then enter an optional replacement value followed by a line terminator.

The line terminator directs the subsequent action of Open. If a value is entered, it replaces the contents of the word whose address and current value are shown by the first part of the line-2 display. The slash (/) is part of the line-2 display.

Line Terminator Options:

ESC = ESCape or ALTMODE: ESC (ALTMODE) terminates acceptance of further input and is the only means of exit from the MCR Open function.

CR = Carriage return: the next sequential location is opened.

^CR = Up-arrow (circumflex on some terminals) carriage return: the previous location is opened.

*CR or @CR = Asterisk carriage return, or at-sign carriage return: the location pointed to by the contents of the opened location is opened.

Example:

```
>OPE 4+10/TASK=CYCLE
000014 060014/350<ESC>
```

In this example, the value of the virtual address of the task CYCLE is 4, while n equals 10, yielding the effective task-relative address of 000014. The system responds by printing the virtual address (000014), and the contents of the address (060014), followed by the slash. The operator responds by entering the new value 350 and the ESCape character. The value 350 replaces the previous contents of task virtual location 000014.

Command Error Messages:

OPE -- BYTE ADDRESS

The address specified as the argument to the Open command is an odd address.

The address 3000 is a legal address; however, the address 3001 is illegal.

OPE -- INPUT I/O ERROR

In attempting to read the next command, Open detected an error.

OPE -- INVALID ADDRESS

The address specified as an argument in the Open command references a nonexistent memory location, an address outside of the specified partition, or an address outside of the task's virtual address space.

OPE -- TASK NOT FIXED

An attempt is being made to open a task virtual location. Since the task is not fixed, it may or may not be in memory at the time the command is issued.

Notes:

1. In a mapped system, specifying an illegal location or accessing beyond the limits of the partition will result in an error. The user is limited to the memory region specified (TASK or PAR).

2. In an unmapped system, any location in physical memory can be accessed.

3. In a mapped system, any address of physical memory may be accessed by not specifying a keyword.

SAVE (privileged)

SAV

The Save command is used to write the main-memory resident image of an RSX-11M system into a task image file such that a hardware bootstrap or the Boot command may later be used to reload and restart it. The saved system is written into the file from which it was originally booted. This command is intended to provide the user with a facility for building development systems that have tasks already installed, thus, eliminating the need for repetitive task installation following every system bootstrap.

Format:

SAV[E] [/WB]

Keyword:

/WB Indicates that a boot block pointing to the system image is to be written out to the system device. The new boot block will point to the file saved with the issuance of the command. Thus, on the next hardware bootstrap, it will be this file that is loaded. Without the /WB switch, the file previously pointed to by the boot block remains in effect, that is, it is not overwritten.

Example:

```
>SAV
```

The current status of the system is saved on the system disk. System changes made by the Redirect or other MCR commands are also saved with the system main-memory resident image.

Command Error Messages:

SAV -- I/O OUTSTANDING

Tasks remain in the system with I/O requests waiting to complete. A system cannot be saved with I/O outstanding.

SAV -- LABEL BLOCK I/O ERROR

In saving the system image, Save writes the transfer address in the label block of the system image file. An error occurred during this write attempt.

SAV -- NOT VALID SAVE DEVICE

A system may only be saved on a directory device.

SAV -- VOLUME(S) STILL MOUNTED

A system may not be saved with volumes mounted. Dismount the volumes and retry the Save command. If necessary, use the MCR Devices command to determine which volumes are mounted.

Notes:

1. The Save command verifies that the system is inactive by making the following checks:

   a. No tasks have outstanding I/O; and

   b. No mountable devices are mounted.

   An error is reported if either of these checks fails.

2. All RSX-11M system images reside on a file-structured volume as a special form of a task image. This special format is a task image without a task header. There may be more than one system image on a volume (for example, a program development system and a production or test system).

3. This command writes out the system image to the file from which it was booted.

4. A system may be booted via the hardware bootstrap, in which case, a specified file is booted (see /WB), or it may be booted by the Boot command.

5. When the bootstrap block is written, the physical disk block address of the system image file is stored with it. However, the file can be deleted. If there is file system activity, the blocks which were allocated to the system image may subsequently be reallocated to another file. A subsequent bootstrap using the boot block will result in the loading of possibly random data.

TASK LIST - ATL FORMAT (privileged) | TAL

This command will display, on the entering terminal, the names and status of all tasks installed in the system in the format of the ATL command. See the ATL command for a description of the printout.

Format:

```
TAL
```

Example:

```
>TAL
. LDR. 052644  LDR     -000777 -EXE -CHK FXD PRV
   TI - CO0:  IOC - 0.  EFLG - 000001 000000  PS - 170340  PC - 045170
   REGS 0-6  044632 000010 052364 061332 062660 061300 000534
TKTN   061040  SYSPAR 100000-107777 -EXE -PMD -CHK OUT PRV
   TI - CO0:  IOC - 0.  EFLG - 000001 000000
RMDEMO 053600  GEN     152000-162477 -PMD -CHK PRV WFR
   TI - TT0:  IOC - 0.  EFLG - 000001 000000  PS - 170000  PC - 106566
   REGS 0-6  105772 104114 000000 10577 001601 105772 101204
...MCR 061300  SYSPAR 100000-107777 -PMD PRV
   TI - TT5:  IOC - 0.  EFLG - 000001 040000  PS - 170000  PC - 103520
   REGS 0-6  000106 000001 000004 050372 077762 061300 100372
F11ACP 062430  FCPPAR 110000-131777 -PMD ACP PRV WFR
   TI - CO0:  IOC - 0.  EFLG - 000002 040004  PS - 170000  PC - 100470
   REGS 0-6 062442 000005 000001 056770 053054 046212 100264
```

4.4 TASK NAMING CONVENTIONS

RSX-11M may be configured into a single-user system or a multiuser system. In order to run many copies of the same task image with as little impact on the user as possible, a set of naming conventions and an algorithm have been used. These conventions are such that the entire naming algorithm is transparent to the casual user of the system. This naming convention applies to all tasks which are invoked as external MCR functions (e.g., PIP, MOU, MAC, etc.). On a single-user system, these tasks are prefixed with "..." (three periods) to create the task name (e.g., PIP's task name is "...PIP"). For multiuser systems, another naming convention applies. The task name is suffixed with the unit number of the terminal from which it was invoked (e.g., PIP invoked from Terminal 2 (TT2:) is named "PIP2"). The user does not need to know this, however, since he may refer to the task by the three character command name in all commands that require a task name. Thus, to abort an external MCR function, the user types ABO xxx, where xxx is the command name.

For tasks invoked using the install, run, and remove option of the Run command, the naming is straightforward. It is always named "TTn", where n is the unit number of the terminal from which it was started. Again, the user is not required to know this, since the naming algorithm will automatically find the right task. However, by not specifying a task name, the system will automatically search for task "TTn".

The algorithm which follows applies to all MCR commands which require a task name as an argument.

1. Search for the explicitly named task; if found, operate on it.

2. If the named task is not found, insure that the task name given is three characters long. If it is not, the task cannot be found. If it is, search for "xxxn", where xxx is the 3-character name specified on the command, and n is the unit number of the terminal from which the command was issued. If xxxn is found, operate on it.

3. If xxxn is not found, search for task "...xxx". If it is found, operate on it; if not found, the task cannot be found. The TI terminal for ...xxx must be the same as the terminal from which the command was issued or the task will not be found.

4. If no task name was specified, search for task "TTn", where n is the unit number of the terminal from which the command was issued. If TTn is found, operate on it; if TTn is not found, the task cannot be found.

This algorithm, as can be seen, proceeds in an orderly way and will always find the correct task. In addition, task names used by single-user systems default correctly, since tasks of the name "xxxn" will not be in the system, and the search will go to step 3 of the algorithm. For tasks on multiuser systems running in system-controlled partitions, this algorithm also works, since it always searches for task "xxxn" before "...xxx". (See the RSX-11M System Generation Manual for a more detailed description of multiuser systems.)

Examples:

1. On terminal TT2:, the sequence of commands

```
>RUN $PIP
PIP>^C<CR>
MCR>ABO<CR>
```

   will abort task TT2 (in reality PIP), which was caused to be run by the Install-Run-and-Remove command.

2. On terminal TT0:, in a single-user system, the command sequence

```
>PIP
PIP>^C<CR>
MCR>ABO PIP
```

   will abort task "...PIP".

3. On terminal TT1:, where task PIP1 has been installed or has been dynamically created in a system-controlled partition, the sequence,

```
>PIP
PIP>^C<CR>
MCR>ABO PIP
```

   will abort task "PIP1".

4. On terminal TT1:, where task ...PIP is running, the sequence

```
>PIP
PIP>^C<CR>
MCR>ABO PIP
```

   will abort task "...PIP", since PIP1 is not in the system.

NOTE

In this case, a check is made to determine if the TI terminal for task ...PIP is the same as TT1:. If not, the task would not be found.

CHAPTER 5

INDIRECT COMMAND FILES

5.1 INDIRECT COMMAND FILES

An indirect command file is a text file containing a list of commands exclusive to, and interpretable by, a single task, usually a system-supplied component of RSX-11M, such as MACRO-11 or the Task Builder. In addition to task components, MCR has an indirect file processor (...AT.) which interprets commands and either acts on them or passes them on to MCR for processing.

Indirect files are initiated by replacing the command string required by a task with a file specifier preceded by an at sign (@).

For example, to initiate a file of MACRO-11 commands, the user would input:

```
>MAC @INPT.CMD
```

After MACRO-11 is initiated, it accesses the file INPT.CMD for its commands.

To initiate a file of MCR commands, the user would simply enter the file specifier preceded by an at (@) sign, as shown below,

```
>@filespec
```

anytime MCR will accept input.

The default file type for indirect command files is .CMD. An indirect file may contain any command interpretable by the task to which it is directed, but no others.

Indirect files for tasks other than MCR may not contain indirect file references.

MCR indirect files may reference other MCR indirect files, but the maximum nesting depth is limited to four levels of files. MCR will display on the requesting terminal every command retrieved from an MCR indirect command file.

5.2 MCR INDIRECT FILE PROCESSOR

The MCR indirect file processor task (AT.) is capable of reading a command input file and interpreting each line as either a command to be passed directly to MCR or a reguest for action by the task itself.

Commands for MCR are simply entered on a line, requiring no special prefix characters, whereas commands to the indirect command file processor always begin with a period (.). Comments are indicated by a leading exclamation point or semicolon and are printed on the entering terminal.

Commands interpreted by the indirect file processor provide the following capabilites:

Define a label (.label:);
Ask a question and wait for reply (.ASK);
Delay execution for a specified period of time (.DELAY);
Branch to a label (.GOTO);
Determine whether a task is active or not (.IFACT/.IFNACT);
Determine whether a symbol is defined or not (.IFDF/.IFNDF);
Determine whether a task is installed in the system or not (.IFINS/.IFNINS);
Determine whether a symbol is true or false (.IFT/.IFF);
Branch to a label on detecting an error (.ONERR);
Pause for operator action (.PAUSE);
Set the value of a symbol to true or false (.SETT AND .SETF);
Wait for a task to finish execution (.WAIT);
Start a task, pass a command line to it and continue (.XQT), and provide commentary ("!text" OR ";text").

5.2.1 Symbols

It is possible to define symbols which represent true or false values and which may be tested to control flow through the indirect command file. Symbols are one to six ASCII characters in length. Numbers, upper and lower case letters, and special characters are all valid. Symbols are defined by the .ASK directive or by the .SETT and .SETF directives. The first appearance of the symbol in one of these directives defines the symbol and sets it to true or false; subsequent directives may test the value or redefine it.

There is one special symbol which is always defined and is used to determine whether the system on which the user is running is mapped or unmapped. This symbol is

```
<MAPPED>
```

The brackets are required syntax. If the system is unmapped, <MAPPED> has a value of false; if the system is mapped, it has a value of true.

Example:

```
.IFT <MAPPED> SET /UIC=[1,54]
```

If the system is mapped, the Set command is passed to MCR.

5.2.2 Commands to MCR

A command is normally passed directly to MCR for processing, and the indirect file processor waits until MCR has processed it before going on to the next command in the file. However, if the first character in the line is a period (.), this signals that the command is to be interpreted by the indirect file processor and appropriate action is to be taken. The command is scanned and processed from left to right. When the final command on the line is processed and the conditions are true, the rest of the line will be passed to MCR. If the conditions are false, the rest of the line will be ignored and the next line will be processed.

Example:

```
.IFNINS ...PIP INS [1,50]PIP
```

If task ...PIP is not installed, the command INS [1,50]PIP will be passed to MCR for execution. If ...PIP was installed, the INS [1,50]PIP command would not be passed to MCR. Each command passed to MCR is also displayed on the entering terminal as a log of executed commands.

5.2.3 Switches

The indirect file processor accepts two switches: /TR and /DE.

/TR Indicates that a trace of each line processed is to be provided on the entering terminal. This function is useful for debugging an indirect command file. Each command line, including those that begin with a period (directive), is displayed. The period on the first directive in the line is changed to an exclamation mark (comment) and displayed. If the command causes some action to occur, the next line printed will indicate the action; usually, this line consists of the MCR commands issued as a result of the previous directive.

/DE Indicates that the indirect command file is to be deleted when processing is complete.

These switches may be prefixed with a minus sign (-) or "NO" to negate the action of the switch, e.g., /NOTR will not generate a trace. The defaults are /NOTR and /NODE.

5.2.4 Multi-Level Indirect Files

Up to four levels of indirect command files may be specified. Each time a new level is entered, all symbols previously defined are masked out of the symbol table, and only symbols defined in the current level are avaliable; symbols are local to the level of command file in which they are defined. When control returns to a previous level, the symbols defined in that level are available again.

5.2.5 Syntax

All directives are separated from their arguments by at least one space. MCR commands are separated from directives by at least one space.

5.2.6 Directives

5.2.6.1 Define a Label

.label:

Labels are always at the beginning of the line; they may be on a line with additional directives and/or an MCR command, or on a line by themselves. When control is passed to a line with a label, the line is processed as if the label was not there. Commands do not have to be separated from the label by a space. Only one label is permitted per line. Labels are 1 to 6 characters long and must be preceded by a period and terminated by a colon. Any valid ASCII character is allowed in a label.

Examples:

```
.100:
.HERE:
.10$:
```

5.2.6.2 Ask a Question and Wait for a Reply

.ASK

The .ASK directive will ask a question, wait for a reply, and set a specified symbol to the value of true or false, depending on the reply. If the symbol is not defined, an entry will be made in the symbol table. If the symbol is already defined, its value will be set by the answer to the question.

Format:

```
.ASK ssssss txt-strng
```

where:

ssssss = 1- to 6-character symbol which is assigned a true/false value.

txt-strng = any ASCII string of characters which must be preceded by at least one blank.

The text-string will be prefixed with an asterisk and suffixed with "? [Y/N]:" when it is displayed. Three answers are recognized by the indirect file processor:

1. Y<CR> - set symbol ssssss to true.
2. N<CR> - set symbol ssssss to false.
3. <CR> - set symbol to false. <CR> indicates carriage return.

Example:

```
.ASK A DO YOU WANT TO INSTALL PIP
```

will display

```
>* DO YOU WANT TO INSTALL PIP? [Y/N]:
```

on the entering terminal. Symbol A will be set to true or false.

5.2.6.3 Delay Execution for a Specified Period of Time

.DELAY

The .DELAY directive is used to delay further processing of the indirect file for a specified period of time.

Format:

```
.DELAY nnu
```

where:

```
nn = Number of time units to delay
 u = T - ticks;
     S - seconds;
     M - minutes, and
     H - hours.
```

The parameter nn is octal, unless terminated with a period. For example:

```
10S is 10(8) seconds
10.S is 10(10) seconds
```

When the DELAY directive is executed, the message

```
AT. -- DELAYING
```

is issued.

When the time period expires and the task resumes, the message

```
AT. -- CONTINUING
```

is issued.

Example:

The directive

```
.DELAY 20M
```

will delay processing for 20(8), or 16(10) minutes.

5.2.6.4 Branch to a Label

.GOTO

Branches from one line in an indirect file to another are accomplished with the .GOTO directive. All commands between the .GOTO directive and the specified label are ignored. Branches may go forward or backward in the file.

Format:

```
.GOTO label
```

where label is the name of the label, but without the leading period and trailing colon. The label must be preceded by at least one space.

Example:

The directive

```
.GOTO 100
```

will transfer control to the line containing the label .100:.

5.2.6.5 Logical Test (IF)

There are a number of directives which make tests; if the test is true, the rest of the command is processed. Logical tests may be combined into a compound logical test with the .AND and .OR directives.

5.2.6.5.1 Test if Symbol Is True or False

.IFT/.IFF

The value of a symbol may be tested for true or false; if the test is true, the rest of the command is processed.

Format:

```
.IFT  ssssss     If symbol ssssss is true.
.IFF  ssssss     If symbol ssssss is false.
```

The symbol must be preceded and followed by at least one blank, except at the end of a line, in which case, it need not be followed by a blank.

Example:

```
.IFT  A  .GOTO  100
.IFF  B  .GOTO  200
```

5.2.6.5.2 Test if Symbol Is Defined or Not Defined

.IFDF/.IFNDF

A test can be made to determine whether a symbol has been defined or not defined; if the test is true, the rest of the command is processed. This directive does not test the value of the symbol.

Format:

```
.IFDF  ssssss      If symbol ssssss is defined.
.IFNDF  ssssss     If symbol ssssss is not defined.
```

The symbol ssssss must be preceded and followed by at least one blank, except at the end of a line, in which case, it need not be followed by a blank.

Example:

```
.IFDF  A  .GOTO 100
.IFNDF  A  .ASK A DO YOU WANT TO SET TIME
```

5.2.6.5.3 Test if Task Is Installed or Not Installed

.IFINS/.IFNINS

A test can be made to determine whether a task is installed in the system; if the test is true, the rest of the command is processed.

Format:

```
.IFINS  tttttt      If task tttttt is installed.
.IFNINS  tttttt     If task tttttt is not installed.
```

The task name tttttt must be preceded and followed by at least one blank, except at the end of a line, in which case, the trailing blank is not required.

Example:

```
.IFINS  ...PIP  .GOTO 250
.IFNINS  ...PIP  INS [1,50]PIP
```

5.2.6.5.4 Test if Task Is Active or Not Active

.IFACT/.IFNACT

A test can be made to determine whether a task is active; if the test is true, the rest of the command is processed.

Format:

```
.IFACT  tttttt      If task tttttt is in execution.
.IFNACT  tttttt     If task tttttt is not in execution.
```

The task name tttttt must be preceded and followed by at least one blank, except at the end of a line, in which case, it need not be followed by a blank.

Example:

```
.IFACT   REPORT  .GOTO 350
.IFNACT  REPORT  RUN REPORT
```

5.2.6.5.5 Compound Tests

"If" tests may be combined via the .AND and .OR directives. In addition, an implied .AND is effected when multiple IFs appear on the same line without being separated by an .AND directive. In this case, the command is executed only if all tests are true. The compound operators .AND and .OR must be preceded and followed by at least one blank.

Examples:

```
.IFT A .AND .IFF B  .GOTO C
.IFT A .IFF B  .GOTO C          Same effect as first line
.IFT A .OR  .IFF B RUN PIP
```

5.2.6.6 Branch to Label on Detecting an Error

.ONERR

If one of the following errors is detected:

1. Undefined symbol referenced,
2. Symbol table overflow,
3. Undefined label, or
4. Syntax error,

control will be passed to the line containing the specified label. This feature provides the user with a means of gaining control to terminate command file processing in an orderly manner.

Format:

```
.ONERR  label
```

Control is passed to the line starting with ".label:" upon detecting an error. The .ONERR directive may be issued at any place in the command file. If the directive is executed, error processing will be passed to that label until a new .ONERR directive is executed. If the label specified by the .ONERR directive does not exist and an error condition has occurred, the task will exit.

Example:

```
.ONERR  100
```

5.2.6.7 Pause for Operator Action

.PAUSE

It is possible to interrupt the processing of an indirect file and wait for operator action. This is done using the .PAUSE directive. Issuing a .PAUSE directive will cause the indirect file processor task to supend itself. The operator may perform some specified operations and then cause the task to resume.

Format:

```
.PAUSE
```

When the indirect file processor task suspends itself, it displays the following message on the entering terminal:

```
AT. -- PAUSING.  TO CONTINUE TYPE "RES tttttt"
```

where:

tttttt = is the name of the task.

When the operator is ready to resume, he types:

```
>RES  tttttt
```

The task then prints out:

```
AT. -- CONTINUING
```

and it continues processing where it left off.

5.2.6.8 Set Symbol to True or False

.SETT/.SETF

A symbol may be defined or its value may be changed by use of the .SETT/.SETF directives. If the symbol is not defined, a symbol table entry is built and set to the appropriate value; if the symbol exists, the value is set appropriately.

Format:

```
.SETT  ssssss     Set symbol ssssss to true.
.SETF  ssssss     Set symbol ssssss to false.
```

The symbol ssssss must be preceded by at least one blank.

Example:

```
.SETT  X
.SETF  ABCDE
```

5.2.6.9 Wait for a Task to Finish Execution

.WAIT

Processing of the indirect command file may be suspended until a particular task has terminated by issuing the .WAIT directive.

Format:

```
.WAIT  tttttt
```

The task name tttttt must be preceded by at least one blank.

Example:

```
.WAIT  PIP
```

5.2.6.10 Initiate Parallel Task Execution

.XQT

Normally, a command is passed to MCR and the indirect file processor waits until the command is completely executed. However, it is possible to initiate a task and not wait for it to complete before executing the next indirect file command. The MCR Run command initiates tasks, and the indirect file processor will continue as soon as the Run command is processed. But the user is not able to pass a command string to the target task via the Run command. Using the .XQT directive, it is possible to start a task, pass a command string to it, and continue processing command lines in parallel with the initiated task.

Format:

```
.XQT  cmd command-string
```

where:

cmd	is the name of the task (e.g., MAC, PIP, DMO).
command-string	is the command to the task.

Using the .XQT command provides a facility to initiate parallel processing of tasks. To synchronize the execution of parallel tasks, the .WAIT directive can be used to suspend command file processing until the specified task has completed.

Example:

```
.XQT  MAC TEST,TEST=TEST
.XQT  TKB BLD,BLD=BLD
.WAIT MAC
.WAIT TKB
```

The example starts an assembly and a task build executing in parallel and then waits for the two tasks to complete.

5.2.6.11 Comments

Comments are indicated by a leading semicolon (;). Comments may also appear within paired exclamation marks. Comments are displayed on the entering terminal to provide an explanation or instructions regarding an indirect command file.

Example:

```
!  THIS IS A COMMENT!
;  THIS IS A COMMENT ALSO
```

5.2.7 Task Name References

Task name references in an indirect command file follow the same rules as for MCR. If the task was started as an external MCR task (e.g., MAC, PIP, DMO), it may be referenced by its 3-character name. Thus, the .WAIT, .IFINS, .IFACT directives need only specify the 3-character command name and the indirect file processor will find the correct task. Of course, a specific task may be referenced using its full 6-character name.

5.2.8 Example of Command File and Its Execution

Command File

```
;
; EXAMPLE OF INDIRECT COMMAND FILE PROCESSOR COMMANDS
;
.SETF PIPINS
.ASK CONT DO YOU WANT TO CONTINUE
.IFF CONT .GOTO 100
.10:.IFINS PIP .SETT PIPINS
.IFNINS PIP INS [1,54]PIP
.IFINS PIP ; PIP IS INSTALLED
.IFACT AT. ; INDIRECT FILE PROCESSOR IS ACTIVE
.IFNDF X ; X IS NOT DEFINED
.SETF X
.IFDF X ; X IS NOW DEFINED
.ONERR 50
.IFT Y .GOTO 60
.50:; Y IS NOT DEFINED - ONERR EXAMPLE
.60:.PAUSE
.DELAY 5S
.IFDF PIPINS .AND .IFDF X ; TEST OF AND
.IFDF PIPINS .OR .IFDF Z ; TEST OF OR
.ASK LOOP DO YOU WANT TO LOOP
.IFT LOOP .GOTO 10
.100:.ASK X DO YOU WANT TO GO TO SECOND LEVEL FILE
.IFT X @LEVEL2
;RETURN TO LEVEL 1 FILE
.XQT PIP
.WAIT PIP
```

Execution of Command File

```
>@ATEXMPL
>;
>; EXAMPLE OF INDIRECT COMMAND FILE PROCESSOR COMMANDS
>;
>* DO YOU WANT TO CONTINUE? [Y/N]:Y
>; PIP IS INSTALLED
>; INDIRECT FILE PROCESSOR IS ACTIVE
>; X IS NOT DEFINED
>; X IS NOW DEFINED
>
AT. -- UNDEFINED SYMBOL Y
>; Y IS NOT DEFINED - ONERR EXAMPLE
>
AT. -- PAUSING.  TO CONTINUE TYPE "RES AT.0  "
>RES AT.
AT. -- CONTINUING
>
>
AT. -- DELAYING
>
AT. -- CONTINUING
>; TEST OF AND
>; TEST OF OR
>* DO YOU WANT TO LOOP? [Y/N]:N
>* DO YOU WANT TO GO TO SECOND LEVEL FILE? [Y/N]:Y
>;
>; LEVEL 2 FILE
>; RETURN TO LEVEL 1 FILE
>PIP
PIP>
>^Z
@ <EOF>
>
```

5.2.9 Error Messages

AT. -- BAD COMMAND OR SYNTAX

A directive is unknown or the syntax of the directive is incorrect.

AT. -- BINARY OPERATOR ERROR

Incorrect use of the .AND or .OR directives.

AT. -- CONTINUING

Not an error message. Indicates that the indirect file processor is resuming execution after a .PAUSE or .DELAY directive.

AT. -- DELAYING

Not an error message. Indicates that a .DELAY directive was just executed.

AT. -- FILE READ ERROR

An error was detected in reading the indirect file. Usually due to records which are more that 80. bytes long.

AT. -- INVALID ANSWER

In response to a question from .ASK, the operator entered something other than Y, N, or null, followed by carriage return. The question will be repeated.

AT. -- MAXIMUM INDIRECT FILES EXCEEDED

An attempt has been made to reference an indirect file at a nested depth greater than four.

AT. -- PAUSING. TO CONTINUE TYPE "RES tttttt"

Not an error message. The indirect file processor just executed a .PAUSE directive.

AT. -- RECORD LARGER THAN 80. BYTES

A record larger than 80. bytes has been encountered. Maximum record size is 80. bytes.

AT. -- REDEFINING SPECIAL SYMBOL

An attempt has been made to change the value of a special symbol (i.e., those bracketed with "<" ">"). They may not be redefined.

AT. -- SYMBOL TABLE OVERFLOW ssssss

The symbol table is full and there is no space for symbol ssssss.

AT. -- UNDEFINED LABEL llllll

The label llllll specified in a .GOTO or .ONERR directive cannot be found.

AT. -- UNDEFINED SYMBOL ssssss

The symbol ssssss is being tested, but it has not been defined.

Notes:

1. The default file type for indirect command files is .CMD.
2. When the indirect file processor reaches the end-of-file at the top level file, it will display:

   >@ <EOF>

   and exit.

CHAPTER 6

MCR ERROR MESSAGES

6.1 INTRODUCTION

All messages from MCR and TKTN appear in this section in alphabetical order. Some messages include an action subheading that suggests remedial action to be taken by the operator. In messages where the required action is obvious or contained in the description of the message, the action subheading is omitted.

6.2 MESSAGES

11 40 FP. EXCEPTION

This is a TKTN message. The task has encountered a floating point exception while executing on a PDP11/40 and no SST routine was specified to handle the trap.

XXX -- ACP NOT IN SYSTEM

The specified Ancilliary Control Processor is not installed in the system.

XXX -- ALIGNMENT ERROR

An attempt has been made to create a partition but the base address or size conflicts with existing partitions or physical memory size.

XXX -- ALLOCATION FOR SYS FILE EXCEEDS VOLUME LIMIT

Unable to allocate system file from specified block due to intermediate bad blocks or end-of-volume.

XXX -- ALREADY MARKED FOR DISMOUNT

The device-unit has already been requested to be dismounted and the associated ACP is in the process of waiting for all accesses to the volume to complete.

XXX -- ALREADY MOUNTED

The network device specified has already been mounted.

AST ABORT. BAD STACK

This is a TKTN message. An AST cannot be effected because the AST parameters cannot be pushed onto the task's stack.

XXX - BAD COMMAND OR SYNTAX

A directive is unknown or the syntax of the directive is incorrect.

XXX -- BAD BLOCK FILE FULL - TASK ABORTED

The disk has more than 102 bad regions on it.

XXX -- BAD BLOCK HEADER I/O ERROR

A write error was detected in writing the bad block file header.

XXX -- BASE ADDRESS MUST BE ON 4K BOUNDARY

The base address of the task is not on a 4K boundary. Applicable only to mapped systems.

XXX -- BASE MISMATCH COMMON BLOCK <common-name>

The base address of the common block, as recorded in the task image, does not match the base address of the resident common block.

XXX -- BINARY OPERATOR ERROR

Incorrect use of the .AND or .OR indirect file processor directives.

XXX -- BLOCK(S) EXCEED VOLUME LIMIT

Specified block (or blocks) exceed(s) physical size of volume.

XXX -- BOOT BLOCK WRITE ERROR

An error was detected in writing the volume boot block.

XXX -- BYTE ADDRESS

The address specified as the argument to the Open command is an odd address. For example, the address 3000 is a legal address; however, the address 3001 is not.

XXX -- CAN'T READ MCR COMMAND BUFFER

UFD was started by a Run command rather than by an MCR command; therefore, the command cannot be parsed.

XXX -- CHECKING ddnn:

Automatic bad block specification is proceeding, using the bad block file produced by the Bad Block Locator utility program.

XXX -- CHECKPOINT AREA TOO SMALL

The area allocated for checkpointing the task is smaller than the partition into which it is being installed.

CHECKPOINT FAILURE. READ ERROR

This is a TKTN message. The task could not be read back into memory after being checkpointed.

*** ddnn: CHECKPOINT WRITE ERROR

This is a TKTN message. Write failure has occurred while attempting to checkpoint a task.

XXX -- CIRCULAR REDIRECT ERROR

An attempt has been made to redirect a device such that a circular list of redirects would result.

XXX -- COMMAND I/O ERROR

An I/O error was generated during a read to an indirect file, or an attempt by another task to obtain a command from MCR failed.

XXX -- COMMON BLOCK IS TASK PARTITION <common-name>

A task's request for access to a common block is rejected because the partition requested is a task partition.

XXX -- COMMON BLOCK NOT LOADED <common-name>

The specified common block is linked to the task but has not been loaded into the system. Install the specified common block, then install the task.

XXX -- COMMON BLOCK OCCUPIED

An attempt has been made to load a common block which is already occupied.

XXX -- COMMON BLOCK PARAMETER MISMATCH <common-name>

Parameters (partition name and PIC attributes) of a common block do not match those in the task's label block.

XXX -- CONTINUING

Not an error message. Indicates that the indirect file processor is resuming execution after a .PAUSE or .DELAY directive.

XXX -- DATA ERROR

Bad block number or contiguous region size too large as specified. Reenter properly.

XXX -- DELAYING

Not an error message. A .DELAY directive was just executed by the indirect file processor.

XXX -- DEVICE ATTACHED

The specified device is currently attached by a task and may not be mounted.

XXX -- DEVICE NOT DEFINED IN NETWORK

The specified device has not been defined in the network topology and therefore may not be mounted.

XXX -- DEVICE NOT IN SYSTEM

The device specified in the command was not generated into the system. Devices that are to be used in the system must be specified during system generation.

XXX -- DEVICE NOT MOUNTED

The device specified in the file specification is not mounted.

XXX -- DEVICE NOT READY

Specified device is not ready.

XXX -- DEVICE NOT REDIRECTABLE

The specified device may not be redirected.

XXX -- DEVICE NOT TERMINAL

An attempt has been made to set terminal characteristics for a non-terminal device.

XXX -- DEVICE OFFLINE

The specified device, although generated into the system, is not physically present in the host configuration.

XXX -- DEVICE WRITE-LOCKED

Specified device is hardware write-locked.

XXX -- DIRECTORY ALREADY EXISTS

The requested user file directory file already exists on the volume.

*** ddnn: DISMOUNT COMPLETE

This is a TKTN message. This indicates that the device requested to be dismounted is now logically disconnected from the system, i.e., all files are deaccessed and the VCB is deallocated.

XXX -- DPB ERROR

A bad DPB was created by MCR. This error indicates that the system itself has faulted. If the error persists, consult your local DEC software specialist.

XXX -- DUPLICATE BLOCK(S) FOUND

A block which has been specified as bad is being defined as bad a second time.

XXX -- FAILED TO CREATED A DIRECTORY

No space exists on the volume, or an I/O error has occurred.

XXX -- FAILED TO ENTER IN MFD

No space exists in the master file directory, or no space exists on the volume, or an I/O error has occurred on the volume.

XXX -- FATAL I/O ERROR

An input or output operation cannot be completed due to the unavailability of a device or dynamic memory, or a device error has occurred.

XXX -- filename FILE HEADER I/O ERROR

An error was detected in trying to write out the file header for the specified system file.

XXX -- FILE HEADER READ ERROR

Mount cannot read either the index file or the storage allocation file.

Action: Determine the problem; if it is a hardware error, contact Field Service.

XXX -- FILE NOT CONTIGUOUS

An attempt is being made to boot a system or install a task from a non-contiguous file. System images and task images must be contiguous.

XXX -- FILE NOT FOUND

The requested file is not in the directory on the specified volume.

XXX -- FILE NOT TASK IMAGE

Data in the label block is not correct, indicating the file is not a valid task image.

XXX -- FILE READ ERROR

An error was detected in reading an indirect file. This error is usually due to records which are more than 80. bytes long.

XXX -- HOME BLOCK ALLOCATE WRITE ERROR

In overwriting a bad home block area, a write error occurred.

XXX -- HOME BLOCK CHECKSUM ERROR

The checksum in the home block and the calculated checksum do not agree. This condition is probably caused by an I/O error.

XXX -- HOME BLOCK I/O ERROR

An error was detected in reading or writing the volume home block.

XXX -- HOME BLOCK READ ERROR

An error as detected in reading the home block. This error usually indicates that the volume is not ready in the drive. Wait until it is ready and reissue the command.

XXX -- HOME BLOCK WRITE ERROR

A write error was detected in writing the volume home block.

XXX -- ILLEGAL DEVICE ddnn:

The device specified at task-build time by the Assign command keyword option (Devices) does not exist in the system.

XXX -- ILLEGAL DEVICE/VOLUME

The device specified is not a valid task residence device.

XXX -- ILLEGAL FIRST APR

A privileged task built to run using APR 4, 5, or 6 as its base, is too large. Using APR4, the task is larger than 12K; using APR5, the task is larger than 8K; using APR6, the task is larger than 4K. Applies only to mapped systems.

XXX -- ILLEGAL FUNCTION

A command has been entered which MCR cannot recognize.

XXX -- ILLEGAL KEYWORD VALUE

A value entered for a keyword exceeds its limits. Enter a legal value.

XXX -- ILLEGAL PRIORITY

The value of the priority in the command is out of range (i.e., not 1 to 250(10), inclusive).

XXX -- INDEX FILE BIT MAP I/O ERROR

An error was detected in writing out the index file bit map.

XXX -- INPUT I/O ERROR

In attempting to read the next command, the Open command detected an error.

XXX -- INVALID ADDRESS

The address specified as an argument in the Open command references a nonexistent memory location or an address outside the boundaries of the specified partition.

XXX -- INVALID ANSWER

In response to a question from the indirect file processor directive .ASK, the operator entered something other than Y, N, or null, followed by carriage return. The question will be repeated.

XXX -- INVALID KEYWORD

A keyword has been encountered which is not recognizable by the specific command processor.

XXX -- INVALID LOAD DEVICE

The Boot or Install command has detected a device which is invalid as a system or task residence device.

XXX -- INVALID TIME PARAMETER

A time field is incorrect.

XXX -- INVALID UIC

A value of zero for either the group number or the member number was detected.

IOT EXECUTION

This is a TKTN message. The subject task has executed an IOT instruction and no SST routine was specified to process the trap.

XXX -- I/O OUTSTANDING

Tasks remain in the system with I/O requests waiting to complete. A system cannot be saved with I/O outstanding.

Action: Wait for I/O to complete and re-enter the command.

XXX -- LABEL BLOCK I/O ERROR

In saving the system image, the Save command writes the transfer address in the label block of the system image file. An error occurred during this write attempt.

XXX -- LABEL BLOCK READ ERROR

The Boot command cannot read the label block of the system image.

XXX -- LENGTH MISMATCH COMMON BLOCK <common-name>

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. A task's label block data must match system data for that task before it can be installed.

XXX -- LISTING DEVICE NOT AVAILABLE

The device-unit to which MCR is attempting to display information (PAR, DEV, etc.) is attached and is unavailable for use.

LOAD FAILURE. READ ERROR

This is a TKTN message. The requested task could not be loaded.

XXX -- LOGICAL DEVICE NOT IN SYSTEM

The specified logical device is not defined in the logical device tables.

XXX -- LUN OUT OF RANGE

An attempt has been made to re-assign a LUN which is greater than the maximum number of LUNs allocated during task build.

XXX -- MAXIMUM INDIRECT FILES EXCEEDED

An attempt has been made to access an indirect file at a depth greater than four levels. MCR indirect files may nest to a maximum of four levels in depth.

MEMORY PROTECT VIOLATION

This is a TKTN message. The task encountered a memory protect violation and no SST routine was specified to process the trap.

XXX -- MFD WRITE ERROR

An error was detected in writing out the master file directory.

XXX -- NEW DEVICE NOT KNOWN TO SYSTEM

New device in Redirect command is not known to the system (does not exist in the device tables).

XXX -- NO BAD BLOCK DATA FOUND

Although automatic bad block specification was selected, no bad block file could be found on the volume. This is a warning message.

XXX -- NO LUNS

The task which is the argument of the LUNs command does not have any logical units. This is not an error message, but rather an indication that there are no assignments to display.

XXX -- NO POOL SPACE

Request for dynamic memory by MCR cannot currently be satisfied by the Executive.

XXX -- NO ROOM AVAILABLE IN STD FOR NEW TASK

This message means that there is no dynamic storage available to make an entry in the System Task Directory (STD); therefore, no task can be installed.

XXX -- NO TRANSFER ADDRESS

The Boot command could not find a transfer address in a virgin system image (result of a task build).

XXX -- NON EXISTENT MEMORY

An attempt has been made to define a partition in non-existent memory.

NON RSX EMT EXECUTION

This is a TKTN message. The task executed an EMT instruction having an argument other than 377(8) and no SST routine was specified to process the trap.

XXX -- NOT ENOUGH APRS FOR TASK IMAGE

Because of the virtual base address of the task, there are not enough APRs left to map the entire task image.

XXX -- NOT MOUNTABLE DEVICE

The specified device-unit is not a mountable device and therefore cannot be mounted.

XXX -- NOT MOUNTED

The device specified is not currently mounted.

*** dnn: NOT READY

This is a TKTN message. Specified device is not ready.

XXX -- NOT SYSTEM IMAGE

The Boot command has determined that the file is not a system image.

XXX -- NOT VALID SAVE DEVICE

The Save command may be issued only to directory devices.

ODD ADDRESS OR OTHER TRAP FOUR

This is a TKTN message. The task executed a word instruction with an odd address or referenced a non-existent memory location in an unmapped system and no SST routine was specified to process the trap.

XXX -- NT DEVICE NOT MOUNTED

The device NT: must be mounted before any other network devices.

XXX -- OLD DEVICE ATTACHED

An attempt has been made to redirect an attached device-unit.

XXX -- OLD DEVICE MOUNTED

An attempt has been made to redirect a mounted device-unit.

XXX -- OLD DEVICE NOT KNOWN TO SYSTEM

An attempt has been made to redirect an unknown device-unit.

XXX -- OTHER VOLUME MOUNTED

An attempt has been made to mount a volume on a device-unit which already has a volume mounted.

XXX -- OUTPUT ERROR

An MCR terminal-write operation has failed.

PARITY ERROR

This is a TKTN message. A parity error occurred during task execution. The task is fixed in memory so that the memory will not be reused for another task.

XXX -- PARTITION BUSY

The partition in which a task is to be fixed is currently occupied by a running task; thus, the task may not be fixed.

XXX -- PARTITION ALREADY EXISTS

An attempt has been made to define a partition with a name that is already in use.

XXX -- PARTITION NOT COMMON

A partition specified for a common or resident library is not defined as a common partition.

XXX -- PARTITION NOT IN SYSTEM

The partition name specified as an argument in a command, or during task build, cannot be found in the system.

XXX -- PAUSING. TO CONTINUE TYPE "RES tttttt"

Not an error message. The indirect file processor just executed a .PAUSE directive.

XXX -- PRIVILEGED COMMAND

A command was issued from a non-privileged terminal that is available only to privileged terminals.

XXX -- PRIVILEGED TASK LARGER THAN 12K

Privileged tasks have a maximum size of 12K.

XXX -- PSEUDO DEVICE ASSIGNMENT ERROR

A logical device name may not be assigned to a pseudo device.

XXX -- PSEUDO DEVICE REDIRECT ERROR

An attempt has been made to redirect a pseudo device to another pseudo device.

XXX -- RECORD LARGER THAN 80. BYTES

An indirect file for MCR contains a record whose length is greater than 80. bytes.

XXX -- REDEFINING SPECIAL SYMBOL

An attempt has been made to change the value of an indirect file processor special symbol (those bracketed with "<" ">"). Special symbols may not be redefined.

RESERVED INST EXECUTION

This is a TKTN message. The task has executed an illegal instruction and no SST routine was specified to process the trap.

XXX -- RETRIEVAL POINTERS WRONG FORMAT

The index file retrieval pointers are not in the correct format for RSX-11M. Refer to the RSX-11 I/O Operations Reference Manual for details.

*** ddnn: SELECT ERROR

This is a TKTN message. The selected device is not ready, or there are multiple drives with the same unit number.

XXX -- SPACE USED

An attempt has been made to create a partition or a subpartition in a main storage area already occupied.

XXX -- SPECIFIED PARTITION FOR COMMON BLOCK

An attempt is being made to install a task in a common block.

XXX -- SPECIFIED PARTITION TOO SMALL

The task being installed is larger than the partition into which it is being installed.

SST ABORT. BAD STACK

This is a TKTN message. An SST cannot be effected because the SST parameters cannot be pushed onto the task's stack, or a stack overflow has been detected in an unmapped system, as indicated by a nonzero value in the header guard word.

XXX -- STORAGE BIT MAP FILE READ ERROR

An error was encountered while attempting to read the storage allocation.

XXX -- SYMBOL TABLE OVERFLOW ssssss

The indirect file processor symbol table is full; there is no space for symbol ssssss.

XXX -- SYNTAX ERROR

This message generally indicates that the required information has been entered incorrectly. The user can usually correct this condition by retyping the arguments according to the command specifications.

XXX -- TASK ACTIVE

The task used as the argument of the command is active.

XXX -- TASK ALREADY FIXED

The task used as the argument of the Fix command is already fixed in memory.

XXX -- TASK AND PARTITION BASES MISMATCH

The base of the partition does not match that of the task being installed. Applicable only to unmapped systems.

XXX -- TASK BEING ABORTED

A request for the execution of a task is being made, but the task is currently being aborted due to an error or as the result of an MCR Abort command.

XXX -- TASK BEING FIXED

An attempt was made to either fix or alter the priority of a task that is in the process of being fixed.

XXX -- TASK CHECKPOINTABLE

A checkpointable task cannot be fixed in memory.

TASK EXIT WITH OUTSTANDING I/O

This is a TKTN message. Tasks should terminate all I/O operations before exiting. The system does, however, perform all outstanding I/O.

XXX -- TASK IMAGE I/O ERROR

The Install command cannot read the task image file.

XXX -- TASK IMAGE CURRENTLY INSTALLED

The requested task image is already installed. Applicable only to checkpointable tasks.

XXX -- TASK INSTALLED IN PARTITION

An attempt has been made to eliminate a partition containing installed tasks.

XXX -- TASK NAME ALREADY IN USE

An attempt has been made to install a task with the same name as one already in the system.

XXX3-- TASK NOT ACP

The task specified as an ACP does not have the characteristics of an ACP.

XXX -- TASK NOT ACTIVE

The specified task is not currently active.

XXX -- TASK NOT FIXED

An attempt has been made to unfix a task which is not fixed, or to open a memory location of a task which is not fixed.

XXX -- TASK NOT IN SYSTEM

The referenced task has not been installed.

XXX -- TASK NOT SUSPENDED

The task used as the argument of the Resume command is not currently suspended.

TBIT TRAP OR BPT EXECUTION

This is a TKTN message. The task has either set the T bit in the Processor Status Word or executed a Breakpoint Trap instruction and no SST routine was specified to process the trap.

TRAP EXECUTION

This is a TKTN message. The task has executed a TRAP instruction and no SST routine was specified to process the trap.

XXX -- TI REDIRECT ERROR

The psuedo device TI: may not be redirected.

XXX -- TOO MANY COMMON BLOCK REQUESTS

A task is limited to three common block references.

XXX -- TOO MANY LUNS

A task has requested more than 255(10) LUNs to be assigned.

XXX -- TOO MANY SUBPARTITIONS

A main user-controlled partition is limited to a maximum of seven subpartitions.

XXX -- UNDEFINED COMMON BLOCK <common-name>

A task references a common block that is not defined in the system. Usually, this message indicates that the task was built to run in another system.

XXX -- UNDEFINED LABEL llllll

The label llllll specified in a .GOTO or .ONERR directive cannot be found.

XXX -- UNDEFINED SYMBOL ssssss

The symbol ssssss is being tested, but it has not been defined by the indirect command file.

XXX -- UNKNOWN MAIN PARTITION

An attempt to define a subpartition of a non-existent main partition has been made.

XXX -- VOLUME MOUNTED

Mounted volumes may not be initialized.

XXX -- VOLUME NOT MOUNTED

Volume on which a UFD is to be created must be mounted before accessing the Files-11 structure.

XXX -- VOLUME(S) STILL MOUNTED

A system may not be saved with volumes mounted. To correct the problem, dismount the volumes and retry the command. The MCR Devices command may be used to display the mounted volumes.

XXX -- VOLUME STRUCTURE NOT SUPPORTED

RSX-11M does not support the Files-11 structure level of the volume being mounted.

XXX -- WRITE ATTRIBUTE FAILURE

An error was encountered while writing the attributes of either the MFD or the newly created UFD.

XXX -- WRITE CHECK NOT SUPPORTED

Write check is only supported for RK05 disks.

XXX -- WRONG VOLUME

The volume label and the label specified in the command do not match.

APPENDIX A

MCR COMMAND SUMMARY

Initialization Commands

```
ASN [[ppnn:]=[llnn:]][/GBL]

BOO[T] [filespec]

DMO[UNT] dev:[volume-label][/Keyword]

   Keyword:

        /UIC = [group,member]

INI[TVOL] dev:[volume-label][/keyword][/keyword]...

   Keywords:

        /BAD   = [options]
        /CHA   = [characteristics words]
        /EXT   = block-count
        /FPRO  = [system,owner,group,world]
        /INDX  = logical-block-number
        /INF   = initial-index-file-size
        /MXF   = file-count
        /PRO   = [system,owner,group,world]
        /UIC   = [group,member]
        /WIN   = retrieval-pointer-count

INS[TALL] filespec[/keyword][/keyword]...

   Keywords:

        /CKP  = option
        /INC  = size
        /PMD  = option
        /PAR  = pname
        /PRI  = number
        /TASK = taskname
        /UIC  = [group,member]
```

```
MOU[NT] dev:[volume-label][/keyword][/keyword]

   Keywords:

        /ACP   = taskname
        /EXT   = block-count
        /FPRO  = [system,owner,group,world]
        /LRU   = count
        /PRTCL = line protocol name
        /RCHK  = redundancy check routine name
        /TEL   = telephone number
        /UIC   = [group,owner]
        /UNL     (no value is associated with UNL)
        /VI      (no value associated with VI)
        /WIN   = retrieval-pointer-count


SET /keyword

Only one keyword per command is permitted.

   Keywords for setting device characteristics:

        /BUF   = dev:[size]
        /LA30S [= dev:]
        /LOWER [= dev:]
        /PRIV  [= dev:]
        /SLAVE [= dev:]
        /SPEED =dev:[rcv:xmit]
        /UIC   [=[group,member][dev:]]
        /UIC   [=dev:]
        /VT05B [= dev:]
        /WCHK  [= dev:]

   Keywords for partitions:

        /MAIN  = pname[:base:size:type]
        /SUB   = mname:pname[:base:size]
        /POOL  [=top]


TIM[E] [hrs:mins[:secs]] [mnth/day/year]


UFD dev:[volume-label][/Keyword]

   Keywords:

        /ALLOC = number-of-entries
        /PRO = [system,owner,group,world]
```

Informational Commands

DEV[ICES]

LUN[S] taskname

PAR[TITIONS]

TAS[KLIST]

Task Control Commands

ABO[RT] taskname

ALT[ER] taskname /keyword

Keyword:

/PRI = priority

CAN[CEL] taskname

FIX taskname

REA[SSIGN] taskname lun nud:

RED[IRECT] nud:=old:

REM[OVE] taskname

RES[UME] taskname

RUN taskname [/Keyword][/Keyword]

Keywords:

/RSI = nnnnu
/UIC = [group,member]

RUN taskname dtime [/Keyword][/Keyword]

Keywords:

/RSI = nnnnu
/UIC = [group,member]

```
RUN taskname sync [dtime][/Keyword][/Keyword]

   Keywords:

        /RSI = nnnnu
        /UIC = [group,member]


RUN taskname atime [/Keyword][/Keyword]

   Keywords:

        /RSI = nnnnu
        /UIC = [group,member]

RUN [$]filename [/Keyword]...

   Keywords:

        /UIC  = [group,owner]
        /PRI  = priority
        /PAR  = partition name
        /TASK = taskname
        /INC  = size


UNF[IX] taskname
```

System Maintenance Commands

```
BRK

OPE[N] memory address [+ or -n][/Keyword]

   (memory address) (contents of address)/[value]line-terminator


   Line Terminator Options:

        ESC      = ESCape or ALTMODE: if entered, the last
                   location opened is set to the new value.
                   Further, if ESC or ALTMODE is entered with no
                   associated value, no related location is opened
                   and the Open command is terminated. The ESC
                   (or ALTMODE) key is the only means of exit from
                   the MCR Open function.

        CR       = Carriage return: the next sequential location
                   is opened.

        ^CR      = Up-arrow (circumflex on some terminals) return:
                   The previous location is opened.

        *CR      = Asterisk carriage return or at-sign carriage
         or        return: the location pointed to by the final
        @CR        contents of the opened location is opened.
```

SAV[E][/Keyword]

Keyword:
/WB

APPENDIX B

BASIC MCR SYNTAX AND ERROR MESSAGES

```
   ABO[RT]   taskname
   CAN[CEL]  taskname
   RES[UME]  taskname
*  ATL
*  TAL
*  OPE[N]  X                        (unmapped)

         X(+/-n)[/PAR=partition] (mapped)
            <ESC>  exit from OPE
            <CR>  open next
            ^<CR> open previous
            * or @ <CR> open indirect
            -<CR> open location at PC offset

   RED[IRECT] nud=old
   REM[OVE] taskname
   RUN taskname [/RSI=nnu][/UIC=[group,member]]
   RUN taskname dtime [/RSI=nnu][/UIC=[group,member]]
   RUN taskname stime dtime [/RSI=nnu][/UIC=[group,member]]
   RUN taskname atime [dtime][/RSI=nnu][/UIC=[group,member]]
   TIM[E] [[hh:mm:ss][mm/dd/yy]]
        prints date as  dd-mmm-yy

   terminators <CR> or <ESC>
```

Error messages are of the form

```
   XXX -- nn [- message]*
```

where:

```
     XXX        - is the name of the MCR command issuing the error.
     NN         - is the message code number.
     [-message] - is the optional error message text.
```

---

* Selectable during RSX-11S system generation.

BASIC MCR ERROR MESSAGE CODES AND TEXT

CODE	TEXT
1	ILLEGAL FUNCTION
2	SYNTAX ERROR
3	NO POOL SPACE
4	INVALID KEYWORD
5	DEVICE NOT IN SYSTEM
6	PRIVILEGED COMMAND
7	PARTITION NOT IN SYSTEM
8	IO ERROR
9	BYTE ADDRESS
10	INVALID TIME PARAMETER
11	TASK NOT IN SYSTEM
12	INVALID ADDRESS
13	TASK NOT ACTIVE
14	TASK BEING ABORTED
15	TASK NOT SUSPENDED
16	TASK ACTIVE
17	CIRCULAR REDIRECT ERROR
18	OLD DEVICE NOT KNOWN TO SYSTEM
19	NEW DEVICE NOT KNOWN TO SYSTEM
20	PSUEDO DEVICE REDIRECT ERROR
21	OLD DEVICE ATTACHED
22	TI REDIRECT ERROR
23	INVALID UIC
24	ILLEGAL KEYWORD VALUE
26	LISTING DEVICE NOT AVAILABLE
27	DEVICE NOT REDIRECTABLE

INDEX

RSX-11M Operator's
Procedures Manual
DEC-11-OMOGA-B-D

READER'S COMMENTS

NOTE: This form is for document comments only. Problems with software should be reported on a Software Problem Report (SPR) form

Did you find errors in this manual? If so, specify by page.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Is there sufficient documentation on associated system programs required for use of the software described in this manual? If not, what material is missing and where should it be placed?

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Non-programmer interested in computer concepts and capabilities

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

If you require a written reply, please check here. ☐

Please cut along this line.

---------- Fold Here ----------

---------- Do Not Tear - Fold Here and Staple ----------

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

JSINESS REPLY MAIL
) POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

stage will be paid by:

digital

Software Communications
P. O. Box F
Maynard, Massachusetts 01754

digital equipment corporation

Printed in U.S.A.