RSTS/E System Directives Manual

Order No. AA-EZ10A-TC

June 1985

This manual contains general information on run-time systems and describes RSTS/E monitor, RSX emulator, and RT11 emulator directives for the assembly-language programmer.

OPERATING SYSTEM AND VERSION:	RSTS/E	V9.0
SOFTWARE VERSION:	RSTS/E	V9.0

digital equipment corporation, maynard, massachusetts

CONTENTS

FIGURES

TABLES

Preface

Obectives

This manual describes directives to the RSTS/E monitor, the RSX emulator, and the RT11 emulator that you can use in MACRO programs. To use these directives, you should be familiar with the MACRO-11 assembly language. MACRO is the standard assembler for DIGITAL PDP-11 computers and is available under various operating systems for the PDP-11. Note that the syntax is basically the same for all operating systems.

Manual Structure

This manual contains seven chapters and two appendixes:

Chapter 1	Gives an overview of run-time systems and jobs as they relate to the system directives.
Chapter 2	Describes the RSTS/E environment (memory allocation and job space) for the general monitor directives.
Chapter 3	Contains reference material for the general monitor directives that you can use in programs compiled under either the RSX or RT11 run-time systems.
Chapter 4	Describes the RSX environment for the RSX directives.
Chapter 5	Contains reference material for the directives processed by the RSX emulator or the RSX run-time system.
Chapter 6	Describes the RT11 environment for the RT11 directives.
Chapter 7	Contains reference material for the directives processed by the RT11 emulator in the RT11 run-time system.
Appendix A	Lists the RSTS/E errors you can get during directive processing.
Appendix B	Summarizes MODE and RECORD values and other useful information for disks, flexible diskettes, magnetic tape, line printers, terminals, and pseudo keyboards.

Related Documents

For information about the syntax of MACRO assembly language, see the *PDP-11 MACRO-11 Language Reference Manual*.

Where appropriate, this manual references the following manuals from the RSTS/E documentation set:

RSTS/E System Installation and Update Guide

RSTS/E System Manager's Guide

RSTS/E System User's Guide

BASIC-PLUS Language Manual

RSTS/E Programming Manual

RSTS/E Programmer's Utilities Manual

RSTS/E Task Builder Reference Manual

Conventions

This manual uses the following conventions:

→ The arrow means points to, or contains the address of, as when the stack pointer register points to the first item in the stack. For example:

```
SP→   item at top of stack
      item one word down from top of stack
```

( ) Parentheses mean the contents of the item that the parentheses surround. For example, the contents of the program counter would be represented as:

```
(PC)
```

[ ] Brackets around an item indicate that the item is optional. For example:

```
QIO$ param1 [,param2]
```

{ } Braces around two or more items indicate that you must choose one of the enclosed items. For example:

```
QIO$
QIOW$
```

<> Angle brackets around two or more items tell the MACRO assembler that the items make up a list. For example:

```
GLOBAL<name1[,name2,...]>
```

Summary of Technical Changes

The functions of several RSTS/E monitor directives have been expanded for V9.0, and some new directives have been added. The following is a summary:

.AST
A new directive to enable or disable asynchronous AST deliveries from the user program.

.ASTX
A new directive to return control to the calling program when an asynchronous routine completes its operation.

.CMDLN
A new directive to let you pass up to 1K words of data when chaining.

.CORE
Quotas and I&D space.

.MESAG (Message Send/Receive)
A new subfunction (-11) of .MESAG, sends privilege data. A new qualifier, "conditional remove" was added to the Remove Receiver subfunction.

.READA
A new directive, performs asynchronous read and/or read-check operations.

.SPEC (line printers)
New .SPEC functions return the current value of a 32-bit page counter, or the current vertical and horizontal line position.

.SPEC (magnetic tape)
New .SPEC functions return end-of-volume (EOV) and error condition acknowledge (ECA) data.

UU.CFG (Change system characteristics)
A new directive, lets you change the system default characteristics if you have the necessary privileges.

UU.CHK (Check access function)
A new directive, performs privilege checking functions.

UU.LOK
A new subfunction, lets you look up marked-for-delete files.

UU.ONX (Open next file)
A new directive, performs an open next disk file on an I/O channel. The file specification can be for a single file or a series of files.

UU.PRV (Set/clear/read current privileges)
A new directive to let you read the current privilege mask. You can also selectively set or clear bits in the current privilege mask.

UU.RTS
A new subfunction to let you create dynamic regions in memory.

UU.TB1 (Get Monitor Tables - Part 1)
Some of the monitor tables have been expanded.

UU.TB3 (Get Monitor Tables - Part 3)
Some of the monitor tables have been expanded.

UU.TRM
Calls to UU.TRM have been extended for new features.

UU.3PP (Third-party privilege check)
A new directive, enables or disables third-party privilege checking.

.WRITE
A new error condition, end-of-file (EOF), was added for an attempt to write beyond the end of the disk when the disk was opened non-file-structured. Also, additional reasons for getting a PRVIOL error condition have been documented.

.WRITA
A new directive, performs asynchronous write operations.

Five new RSX emulation directives were added:

CRRG$
Creates a dynamic region and optionally attaches to it.

RDBBK$
RDBDF$
Two new directives for use with resident library definition blocks (RDBs).

WDBBK$
WDBDF$
Two new directives for use with window definition blocks (WDBs).

PART I
Introduction

There are two MACRO assemblers available for the run-time systems on RSTS/E: one for RSX and one for RT11. You will use one of these two run-time systems to assemble and, in most cases, run your programs. In addition to user programs, you can also write or modify run-time systems that run under direct control of the RSTS/E monitor.

This manual describes the three types of system directives available to RSTS/E assembly language programmers:

- o General monitor directives
- o RSX emulator directives
- o RT11 emulator directives

Before you start using these directives, it is helpful to understand some basic concepts about RSTS/E run-time systems and jobs.

Run-Time Systems

A run-time system lets you write code that can be shared by many users when it is in memory. In a time-sharing system such as RSTS/E, shareable code is an important consideration.

Run-time systems are normally written as pure code. This means they have only instructions and fixed data, and contain no variable data. Such code saves space, since many jobs can use it. It also saves time since run-time systems are not swapped in and out of memory like user programs. Because the run-time systems contain no variable data, they do not need to be swapped out to disk; they are simply reloaded when they are needed again.

RSTS/E always has at least one permanently resident run-time system. It is called the primary run-time system, because it is the first (and only) run-time system at system start-up. RSTS/E loads other run-time systems when programs request them, and they remain in memory only as long as necessary.

User Environment

The DCL, BASIC-PLUS, RSX, and RT11 run-time systems all have a keyboard monitor. That is, they can accept, analyze, and act on commands you type at a terminal keyboard. The *RSTS/E System User's Guide* gives an overview of the user environment these run-time systems provide.

Program Environment

Run-time systems also provide an environment for programs. MACRO programmers are concerned with the program environment provided by the RT11 run-time system. This run-time system includes:

- o A loader. This code loads a program from disk into memory and starts its execution.
- o An emulator. This code emulates directives handled by DIGITAL's RT-11 operating system for the PDP-11 computer.

Note

> The RSX run-time system only provides a user (command) environment. Both the program loader and the emulation functions for the RSX environment are built into the RSTS/E V9.0 monitor.

A run-time system usually takes up space in the 32K-word area called the user job area. Therefore, a run-time system limits the size of your program to less than 32K words. The RT11 run-time system takes 4K words of virtual memory. The RSX run-time system is in the monitor. Chapter 2 explains space requirements in greater detail.

Should you program under the RSX or RT11 run-time system? RSX is usually a better choice because it is in the monitor and it offers easy access to most resident libraries. Your decision depends on:

- o Whether you are coding MACRO subroutines for use in a high-level language program
- o Which set of program development tools better satisfies your needs
- o Whether you want to use resident libraries
- o Whether you need separate instruction and data space (I&D space) support for programs larger than 32K words

High-Level Languages

When you write MACRO subroutines for use in high-level language programs, the high-level language dictates which run-time system you must use. BASIC-PLUS-2, COBOL-81, PDP-11 COBOL, DIBOL, and FORTRAN-77 all run under the RSX emulator, while FORTRAN-IV runs under the RT11 run-time system. You must compile, link, and run all the modules in your program under the same run-time system, whether your program is written in MACRO or in a high-level language.

Program Development Tools

As Table 1-1 shows, RSTS/E provides one set of program development tools for the RSX environment and another set for the RT11 environment.

Table 1-1: RSX and RT11 Development Tools

Tool	RSX	RT11
Assembler	MAC	MACRO
Linker	TKB	LINK
Librarian	LBR	LIBR
Patch Utility	PAT	PAT

While the tools for each environment perform similar functions, they differ in their speed and capabilities:

- o Assemblers -- RSTS/E supports two MACRO-11 assemblers, the RSX-based MAC assembler and the RT11-based MACRO assembler. The two assemblers are nearly identical in function and performance and produce similar output.

- o Linkers -- RSTS/E supports two linkers: the Task Builder (TKB) for RSX-based programs and LINK for RT11-based programs. While LINK is faster than the Task Builder, the Task Builder is more powerful. It can link much larger and more complex overlay structures (including co-trees) than LINK. Unlike LINK, the Task Builder has options for linking to resident libraries and support for separation of instructions and data.

- o Librarians -- RSTS/E provides LBR for RSX-based programs and LIBR for RT11-based programs. You can create object and macro libraries with either utility. LBR also lets you create universal libraries, which can contain any type of file, including text files.

- o Object module patch utilities -- RSTS/E provides a PAT utility for each environment. Both let you update code in a relocatable binary object module.

For details on these program development tools, see the:

- *RSTS/E Task Builder Reference Manual* -- Describes the Task Builder
- *RSTS/E Programmer's Utilities Manual* -- Describes the RSX-based MACRO assembler, librarian, and object module patch utilities
- *RSTS/E RT11 Utilities Manual* -- Describes the RT11-based MACRO assembler, librarian, linker, and object module patch utilities
- *PDP-11 MACRO-11 Language Reference Manual* -- Describes the MACRO-11 relocatable assembler

Resident Libraries

When you program under RSX, you can easily use DIGITAL-supplied resident libraries (such as RMS-11 and FMS-11) as well as create your own resident libraries. Also, because of the Task Builder's cluster library feature, many resident libraries can share the same virtual address space in your program.

You can also use resident libraries under the RT11 emulator, but the coding is more difficult. Unlike RSX, you must use .PLAS directives to map and create address windows inside your task. Coding these directives can be quite complex. The Task Builder, on the other hand, has options that build tables describing your task and the window to map, and automatically includes the code to perform the necessary .PLAS directives for you. Thus, RSX is a more practical choice than RT11 if you plan to use resident libraries.

Instruction and Data Space

The manipulation of Instruction and Data Space (I&D Space) is an advanced programming technique that effectively doubles the user's virtual address range from 32K words to 64K words. The memory management unit in some PDP-11 processors can relocate data and instruction references with separate base address values. Thus, it is possible to have a user program of 64K words consisting of 32K words of pure instructions or procedure code and 64K words of data -- all within a program's virtual address range.

Directives for Each Programming Environment

RSTS/E has three types of directives:

- o Monitor ("native" RSTS/E directives)
- o RSX emulator
- o RT11 emulator

Monitor directives are processed directly by the RSTS/E monitor (see Chapter 3). You can assemble monitor directives using either the RSX-based or the RT11-based MACRO assembler, and you can use these directives in both user programs and run-time systems. (When you write a program to run under the RT11 run-time system, you must precede all monitor directives with a special "prefix EMT"; see Chapter 6 for details.)

The RSX emulator, which is part of the RSTS/E monitor, processes the RSX emulator directives. These directives, which have basically the same form and function as a subset of the RSX-11M-PLUS operating system monitor directives, perform non-file-structured I/O and trap handling. You must use the RSX-based MAC assembler to assemble the RSX emulator directives, and you can use them only in a user program that runs under the RSX emulator. Chapters 4 and 5 describe the RSX run-time system environment and emulator directives in detail.

RT11 emulator directives are processed by the RT11 emulator, which is in the RT11 run-time system. These directives provide most of the single-job programmed requests available to MACRO programmers using the RT-11 operating system. The RT11 run-time system also provides directives for the RSTS/E environment not available under the RT-11 operating system. You must use the RT11-based MACRO assembler to assemble RT11 emulator directives; you can use them only in a user program that runs under the RT11 run-time system. Chapters 6 and 7 describe the RT11 run-time system environment and emulator directives in detail.

Writing or Modifying a Run-Time System

If you want to modify an existing run-time system or code your own run-time system, you can use either assembler. You may find the RT11-based programming tools easier to use for this purpose than the RSX-based programming tools, mainly because it is easier to link run-time systems with the LINK program than with the Task Builder. Run-time systems always have a specific address for their top (highest) address. When you use LINK, you can specify the top address the first time you link the run-time system. But when you use the

Task Builder, you have to link your run-time system twice -- once to find its size, and again to adjust its top address to the value you want.

Unlike a program, a run-time system can contain monitor directives only, not RSX or RT11 emulator directives. In addition, you must store the run-time system file (the product of assembling and linking) on the system disk in save image library (SIL) format. To create a SIL file, use:

- o MAKSIL -- For run-time systems assembled with MAC and linked with the Task Builder
- o SILUS -- For run-time systems assembled with MACRO and linked with LINK.

Jobs

Like run-time systems, you can view "jobs" from several angles. To the RSTS/E monitor, a job is a unit of work generally associated with activity at a terminal. For example, suppose that a user types a line at a previously inactive terminal. The monitor creates a job, assigning a job number and allocating internal tables for bookkeeping. The monitor then passes control to a new-user entry point in the primary run-time system.

The primary run-time system has code at this entry point that causes the LOGIN program to be loaded from the system disk and executed. LOGIN analyzes what was typed and performs the normal log-in dialogue. When LOGIN exits for a valid login, control passes to the default keyboard monitor, which waits for further input from the terminal.

The monitor considers running the LOGIN program and the default keyboard monitor, and whatever else occurs at the terminal until the user logs out as the same job. (If the log-in sequence was not valid, LOGIN exits with the job still logged out. The monitor destroys the job and releases the job number.)

As a MACRO programmer, your awareness of the job concept probably focuses on the work space RSTS/E provides for each job, and the fact that the run-time system can take part of this work space. Chapter 2 describes the allocation of work space.

This chapter explains how and why one copy of a run-time system, shared by many users, can still take up space in each user's work area. The sections in this chapter and their purposes are:

- How RSTS/E Allocates Memory: Physical and Virtual Addressing

  This section provides some background on memory accessing in the PDP-11.

- Job Space: High Segment and Low Segment

  This section explains how RSTS/E uses memory accessing to define a job space for each user to run programs.

- Important Installation Options

  This section briefly describes resident libraries and the special-case disappearing RSX run-time system.

- Low-Segment Details: First 512. Bytes of the Low Segment

  This section gives specifics on how the monitor uses the low 512. bytes of virtual address space.

- High-Segment Details: Pseudovectors

  This section explains how the monitor and the run-time systems use pseudovectors to communicate with each other.

How RSTS/E Allocates Memory: Physical and Virtual Addressing

All RSTS/E systems use the memory management feature available on PDP-11 computers. This feature extends the addressable memory range of the PDP-11 processor by using hardware registers called Active Page Registers (APRs).

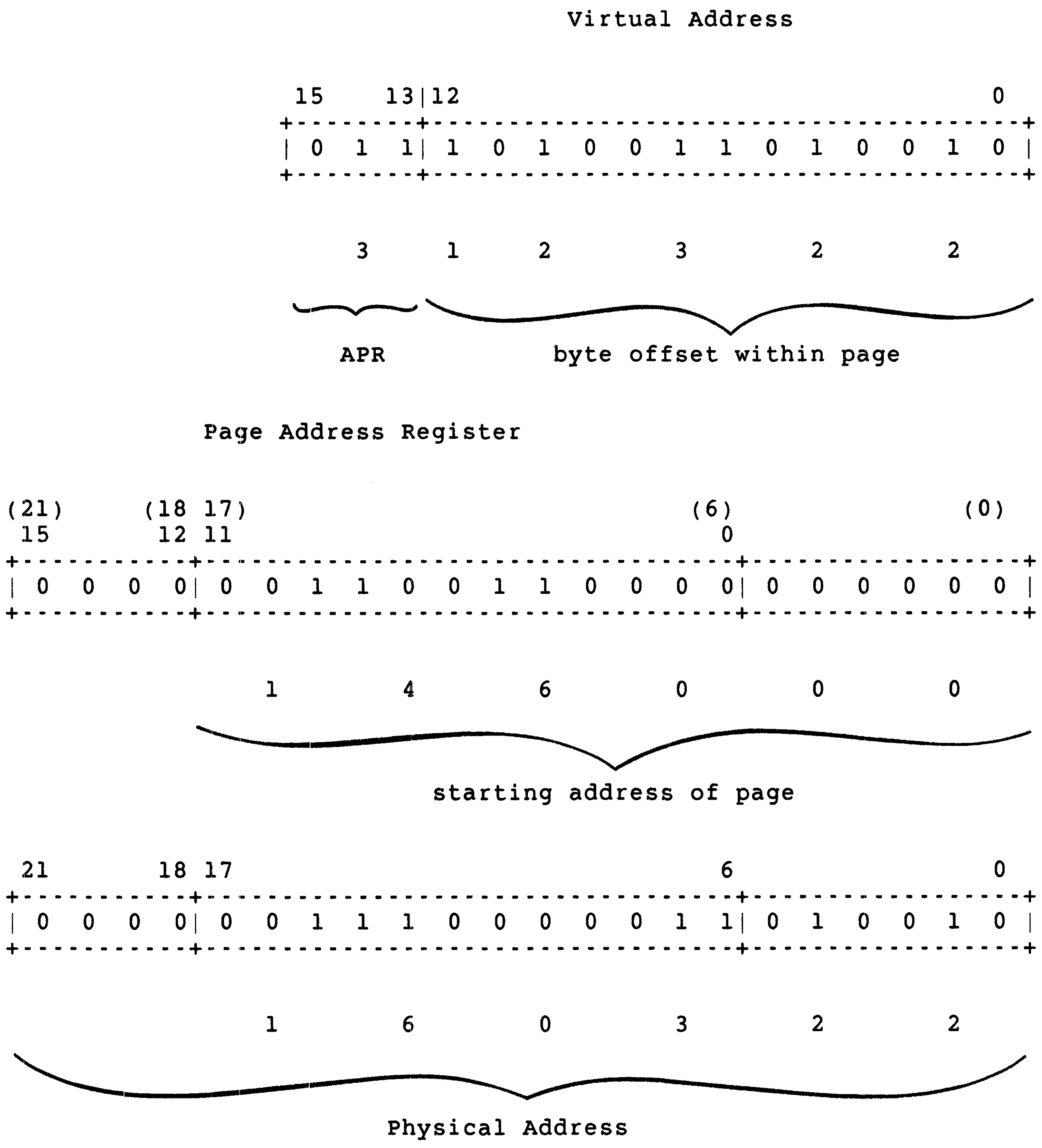
The PDP-11 processor handles 16-bit operand addresses, allowing reference to 32K words. The PDP-11 is byte-addressable, so the address range is from 0 through 2^16 -1 (65535 decimal, 177777 octal), which equals 64K bytes or 32K words. With the memory management unit, RSTS/E treats a 16-bit address as a relocatable (virtual) address that is combined with information in an APR to form an 18-bit (22-bit, for the PDP-11/23-PLUS, 24, 44, 70, 73, and 84) physical address.

On some PDP-11s (11/44, 11/45, 11/50, 11/55, 11/70, 11/73, and 11/84), the memory management unit gives you two areas of 32K words each. You can put code (instructions) in one of these areas and data in the other. RSTS/E requires that you write your program using special techniques to take advantage of both of these areas. This capability is known as Instruction and Data (I&D) Space. You have to use the RSX tools (specifically TKB) to build programs with separate instruction and data sections.

The *PDP-11 Architecture Handbook* explains in detail how the APRs function. Briefly, an APR consists of two 16-bit registers. These registers define a page of contiguous memory. The Page Address Register (PAR) defines the physical memory location where the page begins. The Page Descriptor Register (PDR) defines, among other things, the maximum length of the page and how you can access it (for example, read/write or read-only).

In Figure 2-1, the virtual address of 72322 (octal) identifies APR 3 and byte 12322 (octal) of the page defined by APR 3. The PAR of APR 3 indicates a starting address of 146000 (octal) for the page. The physical address obtained is 146000+012322, or 160322 (octal). The byte offset field in the virtual address is 13 bits long. The maximum size of a page, then, is 2^13 bytes, or 4096 words. In other words, one APR can map a virtual address range of up to 4K words into an equal extent of physical memory.

Figure 2-1 shows how you can combine a virtual address and a PAR to form a physical address in memory.



```
                              Virtual Address

                 15    13|12                                   0
                 +--------+-------------------------------------+
                 | 0  1  1| 1  0  1  0  0  1  1  0  1  0  0  1  0 |
                 +--------+-------------------------------------+

                       3    1    2         3         2         2

                       APR           byte offset within page

             Page Address Register

 (21)      (18 17)                           (6)                (0)
  15        12 11                             0
 +----------+--------------------------------+-----------------+
 | 0  0  0  0| 0  0  1  1  0  0  1  1  0  0  0  0| 0  0  0  0  0  0 |
 +----------+--------------------------------+-----------------+

                 1        4        6        0        0        0

                         starting address of page

  21        18 17                             6                 0
 +----------+--------------------------------+-----------------+
 | 0  0  0  0| 0  0  1  1  1  0  0  0  0  0  1  1| 0  1  0  0  1  0 |
 +----------+--------------------------------+-----------------+

                 1        6        0        3        2        2

                            Physical Address
```

Figure 2-1: How a Physical Address Is Formed

The 16-bit virtual address defines which APR the system uses and the byte offset within that page. The system handles the PAR of the indicated APR as though it contains bits 6-17 (6-21 for the PDP-11/23-PLUS, 24, 44, 70, 73, and 84) of an 18-bit (or 22-bit) physical address, defining the start of the page.

The memory management unit on the PDP-11 consists of two sets of APRs; eight in each set on machines without I&D Space, 16 in each set on machines with separate I&D Space. Since each APR can map a 4K segment of virtual memory to physical memory, each set of APRs can provide access to 32K words of physical memory on non-I&D Space machines, or 64K words on machines with I&D Space capability.

Note

> The PDP-11/44, 45, 50, 55, 70, 73, and 84 have three sets of APRs; the additional set is for supervisor mode mapping, which RSTS/E does not use.

The monitor uses one set, called the "kernel mode APRs," to map itself in physical memory. It uses the other set, called the "user mode APRs," to map the job that is active during the current time slice of time-shared processing.

Figure 2-2 shows the concept of mapping with the APRs.

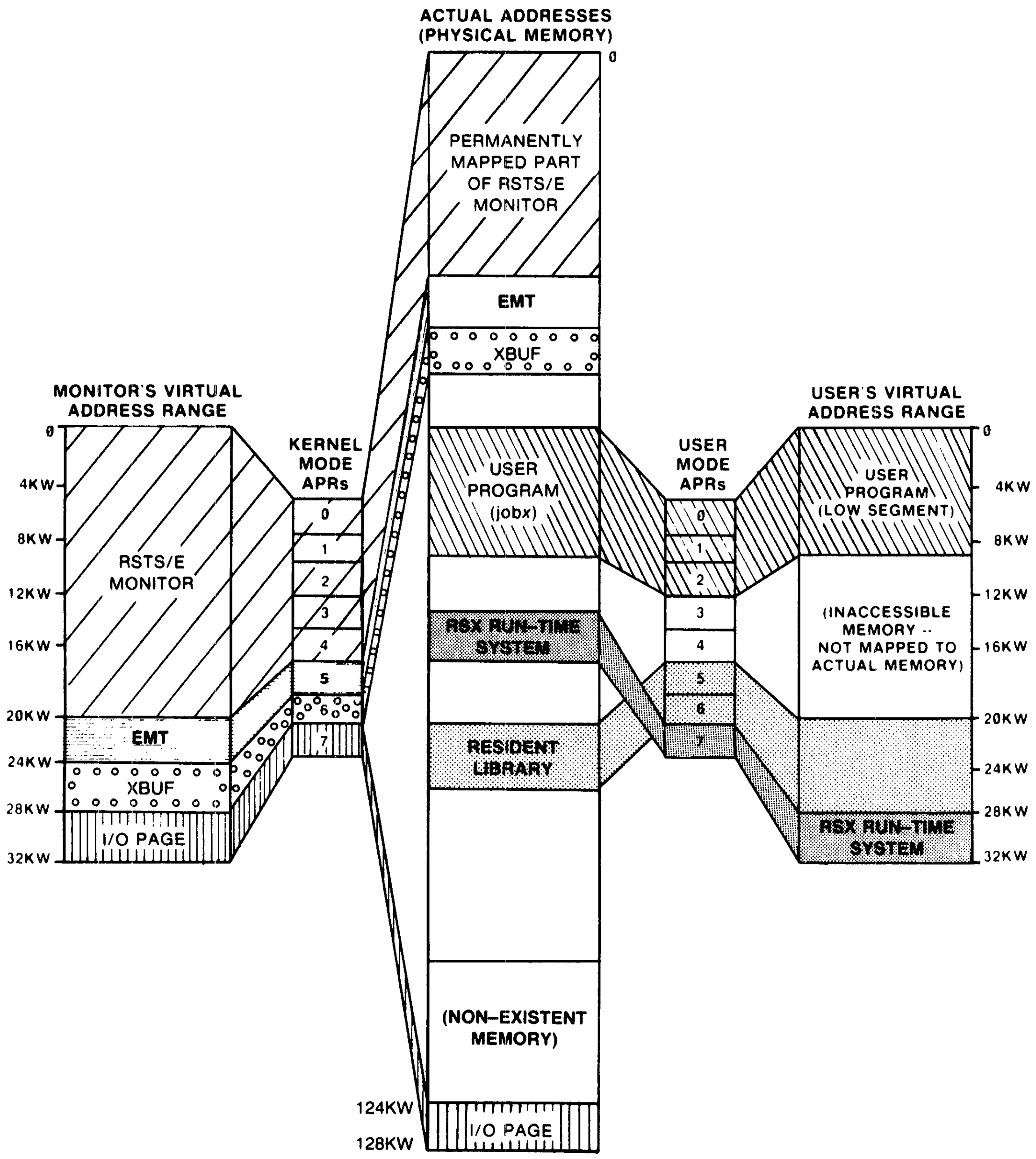


Figure 2-2: Memory Mapping with the APRs

On the PDP 11/44, 45, 50, 55, 70, 73, and 84, the RSTS/E monitor can take advantage of a hardware function, called I&D space. This function lets a program separate its instructions and data into their own virtual address space. On these processors, there are actually two sets of eight APRs for each mode. RSTS/E uses one set to map instructions, and the other set to map data. So, instead of 32K maximum job size, there can be 32K of I-space and 32K of D-space.

The monitor may use this type of mapping, depending on the number of small buffers the system manager selects with the INIT option. (The *RSTS/E System Installation and Update Guide* describes INIT.) For example, if the number of requested small buffers is large enough, the monitor may use D-space APR 1 to map small buffers and I-space APR 1 to map common routines.

Figure 2-3 shows a task executing on an I&D-space system without using separate I&D space. Note how the I-space APRs and the D-space APRs point to the same physical memory.

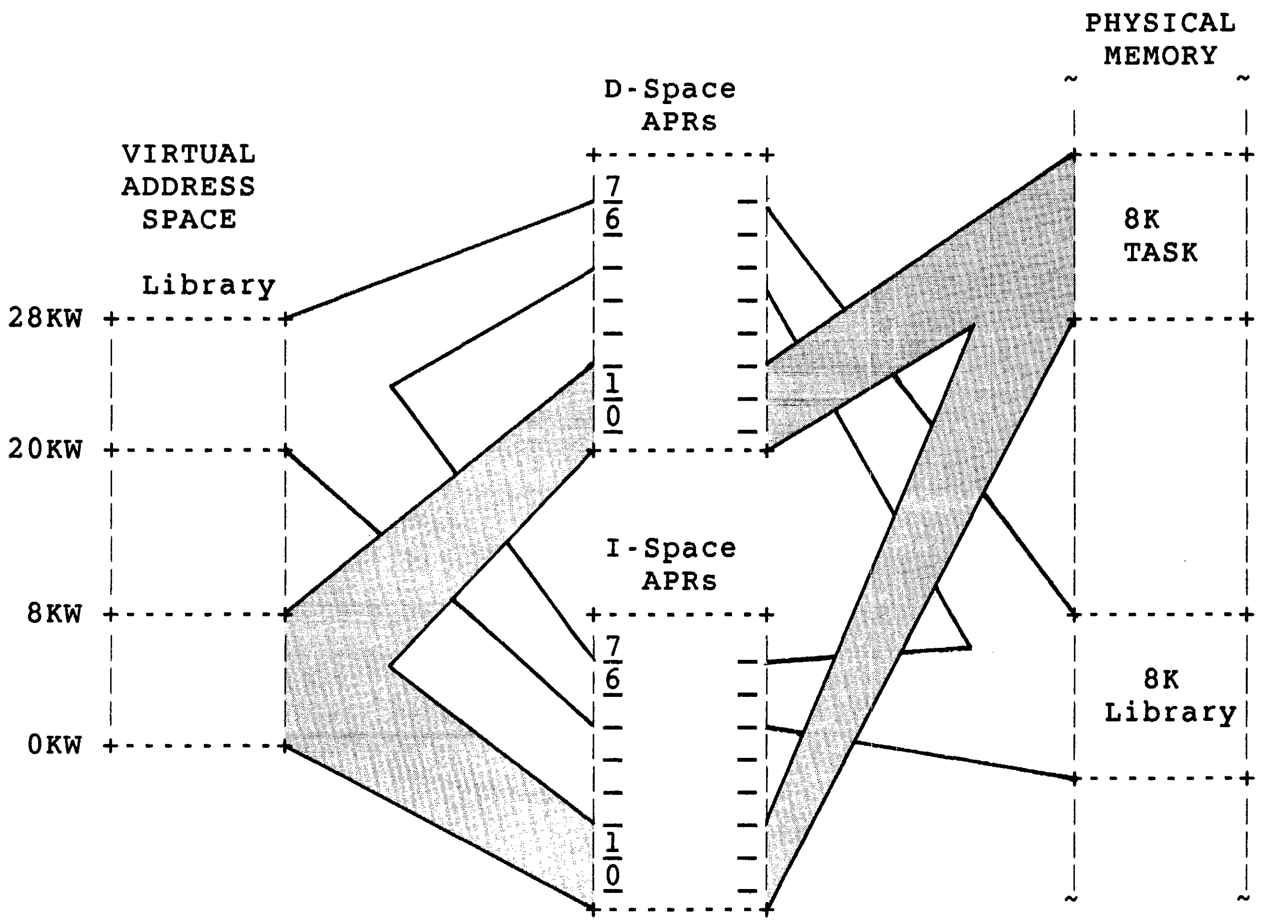


Figure 2-3: Conventional Task Linked to a Library in an I&D-Space System

Figure 2-4 shows a task using separate I&D Space. Note how the tasks I Space APRs and D Space APRs point to different physical memory. User programs can also take advantage of I&D space to increase their available virtual address space.

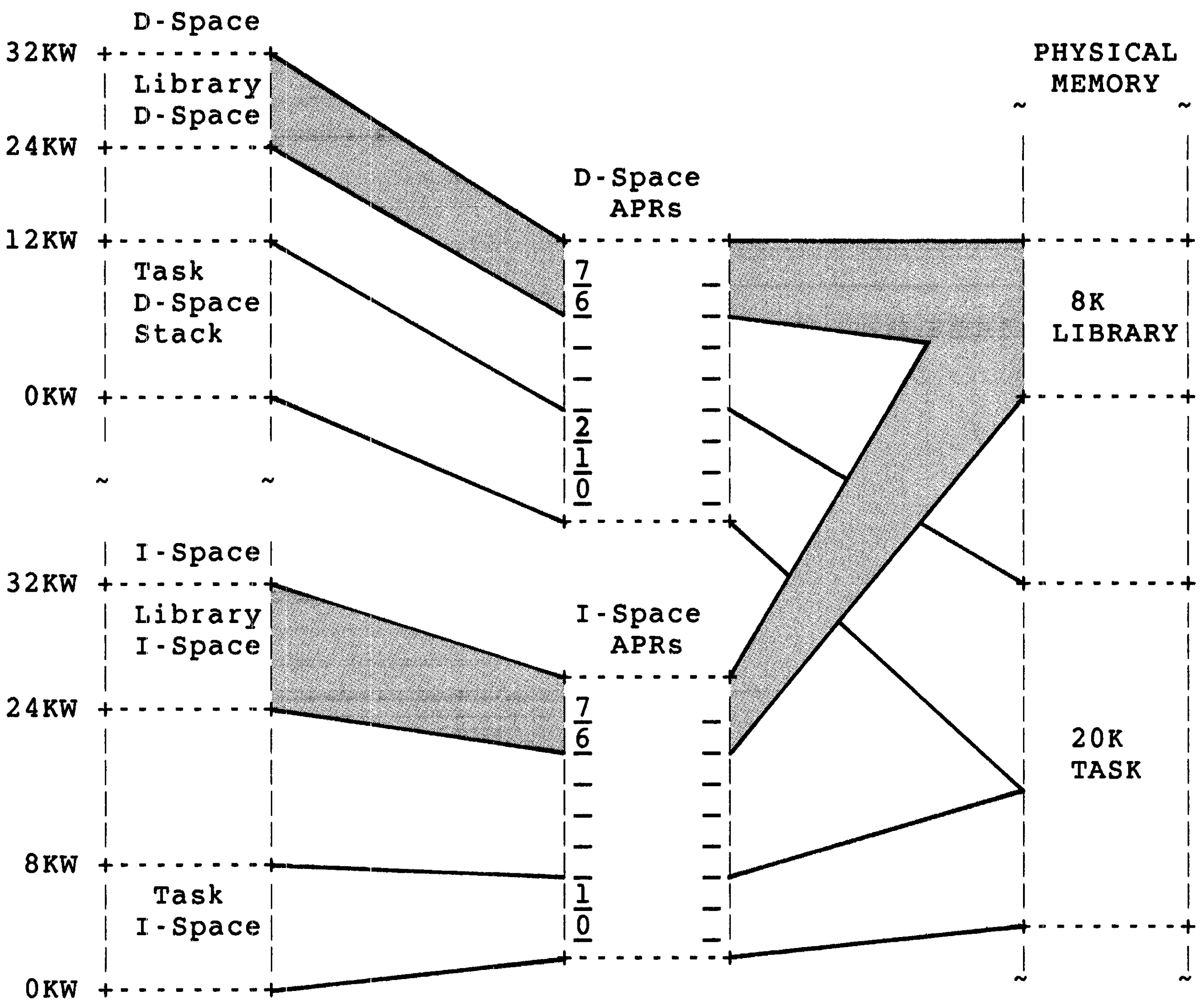


Figure 2-4: I&D-Space Task Mapping in an I&D-Space System

Job Space: High Segment and Low Segment

The RSTS/E monitor is designed to handle work requested by a user through an interface: the run-time system. For example, the BASIC-PLUS, DCL, RSX, and RT11 run-time systems (available as part of a RSTS/E system) each provide their own keyboard monitor to accept and process user commands.

Some of these run-time systems also contain code to handle their own sets of directives, accepting and expanding user program calls to the monitor. For example, the RT11 run-time system provides I/O calls to the monitor using the monitor requests native to RT-11, which the run-time system translates to the equivalent requests known to RSTS/E, which are handled directly by the monitor.

Thus, the run-time system communicates with both the user program and the monitor. Execution control passes back and forth between these three independent elements; data is passed between them using established ranges of virtual addresses. The monitor must be able to access both the run-time system and the user job image at any given time. The monitor does this by setting up the run-time system as part of the 32K words accessible through the eight user APRs.

The monitor assigns an area for the run-time system in the high portion of virtual address space, called the high segment. The user job image (that is, the utility program, compiler, assembler, or executable user program that is currently being executed for the job) is in the low portion of virtual address space, called the low segment.

Note

If you are using the monitor's RSX emulation, there is no high segment at all.

As part of its housekeeping for each job, the monitor keeps track of:

- o Where the currently appropriate run-time system is
- o Where the user job image is
- o What the values were in the program counter register (PC), program status word (PSW), and other job-context information at the end of the last time slice

Before the next time slice for the job, the monitor loads the APRs with the correct values for the job and loads the PC, PSW, and so forth, so execution continues where it left off.

In any case, the high segment or run-time system takes up some multiple of 4K words of virtual address space, due to the APR mapping (see the previous section). For example, the BASIC-PLUS run-time system, can take from 13 to 16K words of physical memory, depending on options selected when the system is installed. Even though the physical memory required may be only 13K words, it still requires four APRs to map this range, leaving four APRs (or a maximum of 16K words) for a user program.

The monitor uses certain areas within the high segment and the low segment to get information from the job (defining what work the monitor is to do for it) and to pass information back to the job.

Figure 2-5 shows the job area in virtual addresses. RSTS/E uses the first 512. bytes to pass information between the monitor, the run-time system, and the user job image for certain types of monitor directives. The monitor uses the pseudovector region in high virtual memory to determine, for example, where control is to be passed when a job is initially entered. The run-time system loads this area with entry points and values to define itself to the monitor.

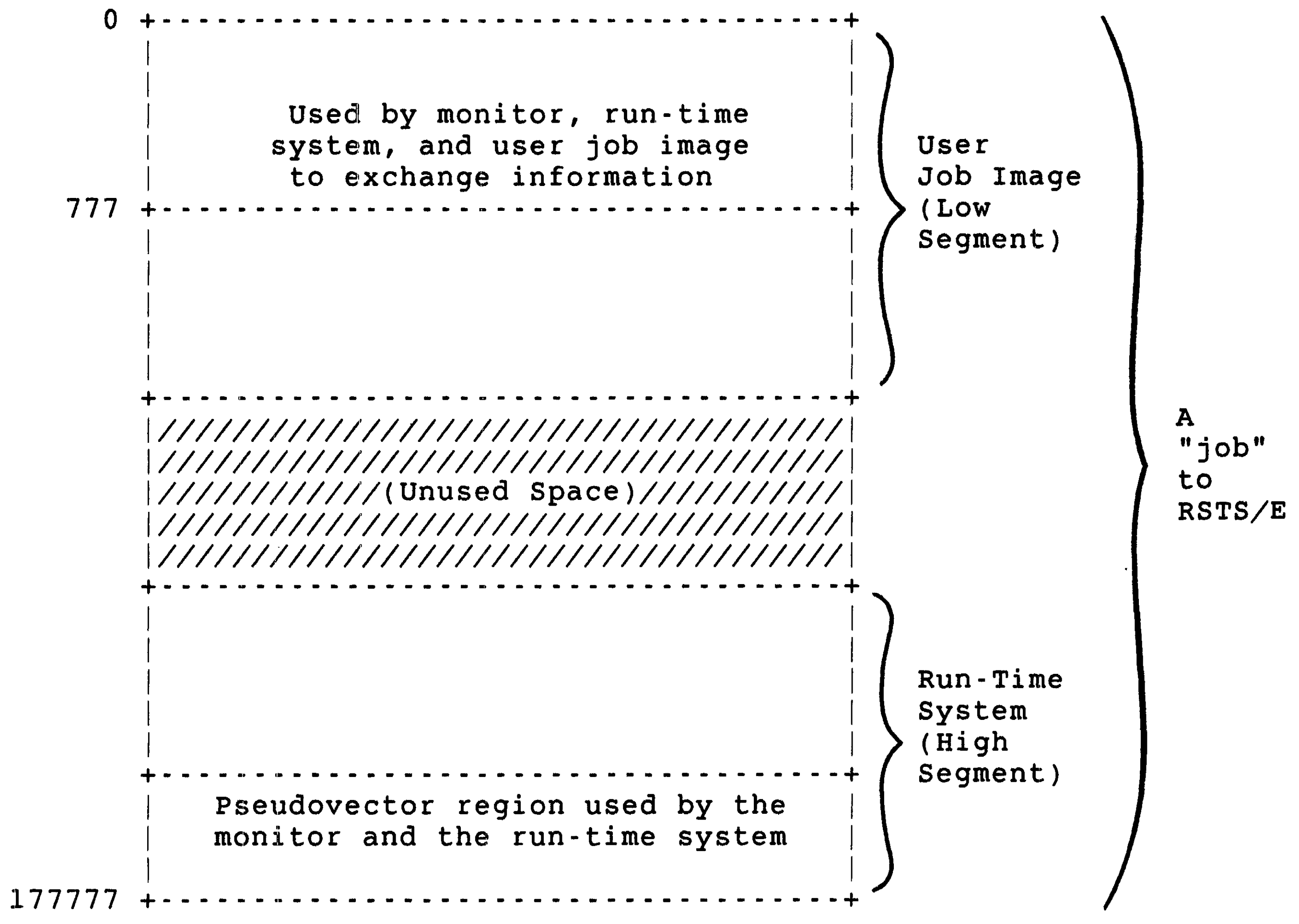

0
Used by monitor, run-time system, and user job image to exchange information
777
User Job Image (Low Segment)
(Unused Space)
A "job" to RSTS/E
Run-Time System (High Segment)
Pseudovector region used by the monitor and the run-time system
177777

Figure 2-5: Job Area in Virtual Memory

The following subsections give more detail on these areas:

- o If you are interested in using the general monitor directives Chapter 3 describes, you should read "Low-Segment Details: First 512. Bytes of the Low Segment."
- o If you want to code your own run-time system or modify one of the existing ones and need to know about the pseudovector region, you should read "High-Segment Details: Pseudovectors."
- o If you are using only the directives Chapters 5 or 7 describe, the RSX and RT11 run-time system directives set up the first 512. bytes of memory for you.

Low-Segment Details: First 512. Bytes of the Low Segment

The monitor attaches special significance to the first 512. bytes of virtual address space in the low segment. The RSX Task Builder and RT11 Linker automatically allocate this space. These programs always assign relocatable addresses beginning at location 1000 unless you request otherwise. If you want to use the general monitor directives Chapter 3 describes, your program must fill parts of this area with information for the monitor; the monitor passes information back in this area.

Rather than use octal addresses, you can use the COMMON.MAC prefix file, which Chapter 3 describes ("Prefix File COMMON.MAC") to assign mnemonic names to commonly used addresses and offsets. COMMON.MAC does not allocate space, but rather assigns mnemonic names to areas within the first 512. bytes of virtual address space. Use the mnemonics assigned with COMMON.MAC to make the code more readable and easier to maintain.

Figure 2-6 shows the general regions in this area. Note that a run-time system may use some of the areas differently when it assumes control. For example, the RSX emulator uses the memory labeled default SP stack area as a table of logical units. The Task Builder automatically generates a user stack after the first 512. bytes of virtual address space. "First 512. Bytes of Low Segment for RSX" in Chapter 4 briefly describes how RSX uses the first 512. bytes.

If you use the general RSTS/E directives, you should reference only the areas that are shown with mnemonics provided by COMMON.MAC. The mnemonics to the right in Figure 2-6 are assigned through COMMON.MAC. A general description of the mnemonics follows Figure 2-6. The general monitor calls in Chapter 3 describe specific formats for the areas the calls use.

```
+-----------------------------------------------+    0
|   controlled solely by job  --  user job       |
|          image or run-time system              |
+-----------------------------------------------+   60
|           used by monitor for job              |
|   context information to make job swappable    |
+-----------------------------------------------+  110
|             used by monitor for                |
|  hardware floating-point context information   |
|            to make job swappable               |
+-----------------------------------------------+  170
|            default SP stack area               |
|                                                |
+-----------------------------------------------+  400   KEY USRSP
|                    keyword                     |
|                                                |
+-----------------------------------------------+  402   FIRQB
|              file request block                |
|                                                |
+-----------------------------------------------+  442   XRB
|            transfer request block              |
+-----------------------------------------------+  460   CORCMN
|                                                |
|              core common area                  |
|                                                |
+-----------------------------------------------+  660
|           controlled solely by job             |
+-----------------------------------------------+  734   USRPPN
|  user-assignable project-programmer number     |
+-----------------------------------------------+  736   USRPRT
|    user-assignable default protection code     |
+-----------------------------------------------+  740   USRLOG
|        user logical device name table          |
+-----------------------------------------------+  776
```

Figure 2-6: First 512. Bytes of Low Segment

KEY (Keyword)

The keyword defines the job's status in the time-sharing environment; for example, the job's privilege. Bits in the keyword can be set and cleared by the monitor or by the job (either the run-time system or the user job image). The job can manipulate some bits in the keyword with the .SET and .CLEAR directives.

The keyword is "refreshed" by the monitor at certain points; for example, when a run-time system is entered at P.RUN, where the intent is to load and execute a program file in the user job image (.RUN directive). For a keyword refresh, the monitor clears bit 15 and bits 7-0 and sets the remaining bits to indicate the job's current status.

Only seven bits (8-14) are significant to the monitor. You can use the other bits in whatever manner you want.

```
            JFSPRI-------------+
          JFFPP------------+   |
        JFPRIV---------+   |   |
      JFSYS--------+   |   |   |
    JFNOPR-----+   |   |   |   |
  JFBIG----+   |   |   |   |   |
JFLOCK-+   |   |   |   |   |   |
       |   |   |   |   |   |   |
   15  14  13  12  11  10  9   8   7                             0
   +------------------------------+-------------------------------+
   |  |        status bits        |                               |
   +------------------------------+-------------------------------+
```

The following descriptions apply when the keyword is set to one:

JFLOCK	The job does not want to be swapped. You can change this bit with .SET and .CLEAR. When this bit is set, the only normal condition that causes the job to be swapped is when the job asks for a memory size expansion (see .CORE directive) and there is not enough room to do the expansion where the job now is in memory.
JFBIG	The job can exceed its private memory maximum (see .CORE directive). This bit is set if the job currently has EXQTA privilege, usually because the job is running a privileged program. JFBIG is an informational bit that the system updates whenever the EXQTA privilege is turned on or off.
JFNOPR	The job is not yet logged in. JFNOPR is an informational bit that the monitor can alter when the job is logged in.
JFSYS	The job is currently running with temporary privileges. The monitor sets JFSYS when a job with insufficient privileges needs to run a privileged program. Once the program is run, the job can regain temporary privilege by setting this bit and can drop privilege temporarily by clearing it.

Note

When a job exits from a privileged utility that can be executed by users with insufficient privileges, the monitor clears this bit so another user cannot use the temporary privilege set up for the job.

JFPRIV This bit is only for compatibilty with RSTS/E releases prior to V9.0. JFPRIV is set if the current job has all of the following privileges: HWCFG, SWCFG, SYSIO, RDMEM, and WWRITE.

Note

Any new software should not reference JFPRIV.

JFFPP The contents of the hardware floating-point unit (if any) should be part of the context of this job. That is, information in the floating-point registers should be saved and restored along with the rest of the user job image during swapping. Any program that uses the hardware floating-point unit should set this bit. It can be changed with the .SET and .CLEAR directives.

JFSPRI The job is running with a special run priority: 1/2 level higher than normal. This bit can be changed with the .SET and .CLEAR directives. The monitor clears JFSPRI when the program exits.

USRSP (User Space)

COMMON.MAC assigns the value 400 to USRSP. The monitor automatically loads this value into the stack pointer register (SP) when a job is created. SP is also reset to this value under certain conditions, effectively establishing a default user stack area for the job beginning at word 376. The user stack area ends at location 170. Any attempt to push the stack past location 170 results in a stack overflow error that is handled by the run-time system (see the description of P.BAD later in this chapter).

You can change SP if you want. However, any attempt to reset SP to any location between 0 and 167 causes a stack overflow error. In addition, the monitor resets SP to 400 when a run-time system is entered with a .RUN, .CCL, or .RTS directive, and when certain catastrophic errors occur, such as a fatal disk error while the user job image was being swapped.

Note

You need to be aware that the monitor resets SP at these times only if you are coding or modifying a run-time system. The system does not return control to a user program under these conditions, because the program cannot recover.

FIRQB (File Request Block)

The FIRQB is the main communication area between the monitor and the job for monitor directives that involve file or device operations such as open, close, and so forth. Either the run-time system or the user job image can use this area:

- o If you use the general monitor directives that Chapter 3 describes, your MACRO program must store values in the FIRQB before issuing some of the directives.

- o If you choose to use the directives in the RSX emulator or the RT11 run-time system, the RSX or RT11 emulation code intercepts the request, sets up the FIRQB and other relevant areas, then calls the monitor to handle the request.

Figure 2-7 shows the general format of the FIRQB, with all mnemonics that COMMON.MAC assigns. In addition, the size of the FIRQB (32. bytes) has the mnemonic FQBSIZ.

```
Mne-   Octal                                          Octal   Mne-
monic  Offset                                         Offset  monic
          +--------------------+---------------------+
        1 |////////////////////|   returned status   |  0 FIRQB
          +--------------------+---------------------+
FQFUN   3 | CALFIP/.UUO subfnc.|    job number * 2   |  2 FQJOB
          +--------------------+---------------------+
FQSIZM  5 |  MSB of file size  | channel number * 2  |  4 FQFIL/
          +--------------------+---------------------+    FQERNO
        7 |   project number   |  programmer number  |  6 FQPPN
          +--------------------+---------------------+
       11 |         file name in RAD50 format        | 10 FQNAM1
          |                (2 words)                 |
       13 |                                          | 12
          +--------------------+---------------------+
       15 |     file type in RAD50 format (1 word)   | 14 FQEXT
          +--------------------+---------------------+
       17 |             LSB of file size             | 16 FQSIZ
          +--------------------+---------------------+
       21 |               buffer length              | 20 FQBUFL/
          +--------------------+---------------------+    FQNAM2
       23 |                   mode                   | 22 FQMODE
          +--------------------+---------------------+
       25 |               status flags               | 24 FQFLAG
          +--------------------+---------------------+
FQPROT 27 |  protection code   | =0, prot. code real | 26 FQPFLG
          +--------------------+---------------------+
       31 |     device name (2 ASCII characters)     | 30 FQDEV
          +--------------------+---------------------+
       33 | =0,unit number real|  device unit number | 32 FQDEVN
          +--------------------+---------------------+
       35 |               cluster size               | 34 FQCLUS
          +--------------------+---------------------+
       37 |  number of entries in directory lookup   | 36 FQNENT
          +--------------------+---------------------+
```

Figure 2-7: General FIRQB Format

XRB (Transfer Request Block)

The XRB is the main communication area between the monitor and the user for monitor directives handling file or device I/O. It is also the area in which the monitor stores information requested by straightforward information-request calls. As with the FIRQB, the general monitor directives (see Chapter 3) require that you store and retrieve information directly to and from the XRB. The RSX and RT11 emulators handle additional directives, which they translate to calls using the XRB (see Chapters 5 and 7).

Figure 2-8 shows the general format of the XRB, with all mnemonics that COMMON.MAC assigns. In addition, the size of the XRB (14. bytes) has the mnemonic XRBSIZ.

```
Mne-   Octal                                          Octal   Mne-
monic  Offset                                         Offset  monic
          +---------------------+---------------------+
        1 |           buffer size in bytes            |  0 XRLEN
          +---------------------+---------------------+
        3 |       number of bytes transferred         |  2 XRBC
          +---------------------+---------------------+
        5 |              buffer address               |  4 XRLOC
          +---------------------+---------------------+
XRBLKM  7 | MSB of block number | channel number * 2  |  6 XRCI
          +---------------------+---------------------+
       11 |            LSB of block number            | 10 XRBLK
          +---------------------+---------------------+
       13 |         wait time for terminals           | 12 XRTIME
          +---------------------+---------------------+
       15 |              device modifier              | 14 XRMOD
          +---------------------+---------------------+
```

Figure 2-8: General XRB Format

A buffer, as defined by XRLOC for its start and (XRLOC+XRLEN)-1 for its last byte, can be either an input buffer or an output buffer. RSTS/E uses input buffers to read data into user memory and output buffers to write data from user memory. These buffers must lie wholly within either the job image (low segment), or the run-time system (high segment), or in a window mapped to some library.

For input buffers, the value passed in XRB+XRBC must be zero. For output buffers, the value passed in XRB+XRBC is the number of bytes to be sent and cannot be zero if the value in XRB+XRLEN is nonzero.

In addition, input buffers are subject to the following rules:

- o If the buffer is in the low segment, the address defined by the contents of XRB+XRLOC must be greater than 170 to avoid destroying the job-context data used in swapping the job.
- o If the buffer is in the high segment, it must not fall within the pseudovector region. That is, it must not fall above the location P.OFF. In addition, the run-time system must currently be mapped read/write because the monitor is writing data to the buffer for the receive (see PF.RW bit description in P.FLAG word).
- o If the buffer is in a library window, the library must be installed as read/write and must be attached and mapped read/write.

CORCMN (Core Common Area)

The CORCMN is used as a common data exchange area when it is necessary to exchange lengthy data (such as strings) between the monitor and the job or between programs running under the same job number.

For example, the monitor uses CORCMN to pass to the job a string that is the full name of a command that has been recognized as a valid Concise Command Language (CCL) command. The RSTS/E CCL lets users type one-line commands to call utilities that might otherwise require several input lines from the terminal. For example:

CCL Form

```
$ PIP FILE1.=FILE2.

$
```

Regular Form

```
$ RUN $PIP
*FILE1.=FILE2.
*CTRL/Z

$
```

To centralize decoding, the monitor analyzes CCL commands by comparing them to those defined by the system manager (usually during system start-up). With the .CCL directive, a job can ask the monitor to analyze a string to see if it is an acceptable command. If it is, the monitor passes control to the run-time system associated with that CCL command and passes the command and any arguments to the job in the CORCMN area.

The general format of the CORCMN area is:

```
+-------------------+-------------------+
|  byte 1 of string |  number of bytes  | 460 CORCMN
|                   |  in string        |
+-------------------+-------------------+
|  byte 3 of string |  byte 2 of string | 462
+-------------------+-------------------+
|                   .                   |
|                   .                   |
|                   .                   |
|       (up to 127. bytes of data)      |
+-------------------+-------------------+
```

USRPPN, USRPRT, USRLOG

This area is set up using the .ULOG directive to store the assigned project-programmer number (USRPPN), default protection code (USRPRT), and assigned logical device names (USRLOG), which the monitor then uses when an .FSS directive is executed. The .FSS directive causes the monitor to convert a file name string to the standard RSTS/E file specification format; that is, to the FIRQB format.

The .ULOG and .FSS directives also let you define and use some nonstandard area to contain these values (see Chapter 3). However, the .ULOG directive sets up 18. words in the same basic format; the .FSS directive expects these values in the same relative locations.

Note

> All manipulation of this data must be done using the .ULOG directive. Direct program access to this data will not be compatible with possible future enhancements and will not be supported.

High-Segment Details: Pseudovectors

The monitor and the run-time system use the pseudovector region to communicate with each other. Figure 2-9 shows the general layout of this area. As with the low 512. bytes of virtual address space, the file COMMON.MAC assigns mnemonic names to locations in this area. These names are shown to the right in Figure 2-9. The following text describes each of these areas in detail. If you want to modify or code your own run-time system, the format and meaning of these areas is important. Otherwise, you might want to examine them to see how the run-time system and the monitor communicate.

```
+-------------------------------------------------+
|      flags describing the run-time system       | 177732 P.FLAG/
+-------------------------------------------------+        P.OFF
|            normal executable file type          | 177734 P.DEXT
+-------------------------------------------------+
|                   (reserved)                    | 177736
+-------------------------------------------------+
|   minimum size, in K words, of user job image   | 177740 P.MSIZ
+-------------------------------------------------+
|     exception address for FIS hardware option   | 177742 P.FIS
+-------------------------------------------------+
|                   (reserved)                    | 177744
+-------------------------------------------------+
|                   (reserved)                    | 177746
+-------------------------------------------------+
|              entry point for new user           | 177750 P.NEW
+-------------------------------------------------+
|  entry point for new user with program to run   | 177752 P.RUN
+-------------------------------------------------+
|    exception address for various "bad" errors   | 177754 P.BAD
+-------------------------------------------------+
|exception address for BPT instruction and T-bit  | 177756 P.BPT
+-------------------------------------------------+
|      exception address for IOT instruction      | 177760 P.IOT
+-------------------------------------------------+
|exception address for nonmonitor EMT instruction| 177762 P.EMT
+-------------------------------------------------+
|   exception address for all TRAP instructions   | 177764 P.TRAP
+-------------------------------------------------+
|         exception address for FPP or FPU        | 177766 P.FPP
+-------------------------------------------------+
|  exception address when user types one CTRL/C   | 177770 P.CC
+-------------------------------------------------+
| exception address when user types two CTRL/Cs   | 177772 P.2CC
+-------------------------------------------------+
|   maximum size (in K words) of user job image   | 177774 P.SIZE
+-------------------------------------------------+
|                  (reserved)                     | 177776
+-------------------------------------------------+
```

Figure 2-9: Format of Pseudovector Region of High Segment

In general, the pseudovector region contains:

- o Values and flags that define the capabilities of the run-time system for the monitor. For example, one flag indicates whether the run-time system has a keyboard monitor.

- o Addresses pointing to locations in the run-time system where the monitor is to pass control when certain conditions occur.

These addresses fall into three categories:

- Addresses for Synchronous Exceptions. Control passes to these locations when the job executes an instruction that causes a trap to the monitor. The monitor passes control to the run-time system along with the contents of the program counter (PC) and program status word (PSW). The term "synchronous" is used in the sense that the trap occurs at the same time as (and is a direct result of) some instruction executed by the job. These traps may or may not indicate an error. For example (except for the PDP-11/23 PLUS or 11/24, if the job executes an instruction with an odd address, control passes to one of these trap addresses. If the job simply executes a BPT instruction, control passes to another of these addresses.

- Addresses for Asynchronous Exceptions. Control passes to these locations as a result of some event, which can be either of the following:

  o External to the execution of the job (for example, the user types a CTRL/C at the terminal)

  o Internal but asynchronous process (such as an error in the hardware floating-point processor, whose execution overlaps that of the PDP-11 CPU)

  When such conditions occur, control passes to the monitor, which passes control to the run-time system, along with the contents of the PC and PSW. If a floating-point trap occurred, the monitor also passes along the floating exception code (FEC) and floating exception address (FEA). For the asynchronous traps, the PC and PSW do not refer to the instruction that caused the trap, but to the instruction that was executing in the central processor when the trap occurred.

- Entry Point Addresses. The monitor passes control to the run-time system at entry-point addresses when some major transition point is reached for the job. For example, when you type a RUN or CCL command at the terminal, the monitor passes control to an entry point in the appropriate run-time system, to load and execute the requested program.

Note

> The term pseudovector arises from the relationship of some of these (one-word) trap addresses in the pseudovector region to the (two-word) vector addresses in kernel-mode memory set up to handle error traps and interrupts in the PDP-11. When the RSTS/E monitor receives control as a result of a trap to certain of these vector addresses, it passes control on to the run-time system at addresses specified in the pseudovector region.

Normally, you code the contents of the pseudovector region as part of the run-time system file. However, the INSTALL/RUNTIME_SYSTEM command, used to define a file as an auxiliary run-time system, has qualifiers that cause the monitor to override certain portions of the pseudovector region in the file and use values assigned in the INSTALL command. For example, one bit in one word of the pseudovector region states whether the run-time system is read/write or read-only when it is loaded in memory. Normally, this would be read-only, but for debugging a run-time system with the Octal Debugging Tool (ODT), which allows you to change memory, the run-time system must be read/write. The /NOREAD_ONLY qualifier in the INSTALL command lets you tell the monitor that until further notice, this run-time system is read/write, regardless of what is specified in the pseudovectors. The *RSTS/E System Manager's Guide* describes the INSTALL/RUNTIME_SYSTEM command.

Run-Time System Capability and Default Definitions

The following mnemonics refer to values and flags that define run-time system capabilities for the monitor.

P.OFF

Use the P.OFF mnemonic to define the first word of the pseudovector region. It is currently set equivalent to 177732, the same as P.FLAG.

P.FLAG

The monitor expects the P.FLAG word to be set with flags that define the capabilities of the run-time system:

```
            PF.KBM-------------+
          PF.1US-----------+   |
        PF.RW----------+   |   |
      PF.NER-------+   |   |   |
    PF.REM-----+   |   |   |   |
  PF.CSZ---+   |   |   |   |   |
PF.EMT     |   |   |   |   |   |
   |       |   |   |   |   |   |
   15  14  13  12  11  10  9   8   7                             0
   +------------------------------+-------------------------------+
   |         flags                |(prefix EMT code if PF.EMT=1)  |
   +------------------------------+-------------------------------+
```

PF.EMT

This bit is set to indicate that the run-time system wants to handle a call that would normally be handled by the monitor. To show how the bit works, it is necessary to first describe what normally happens when a monitor directive is translated and executed.

All of the monitor directives that this manual describes are translated to emulator trap (EMT) instructions. The direct monitor calls are one-for-one translations; that is, one call is translated to one EMT (see Chapter 3). The code to process the call is in the monitor itself.

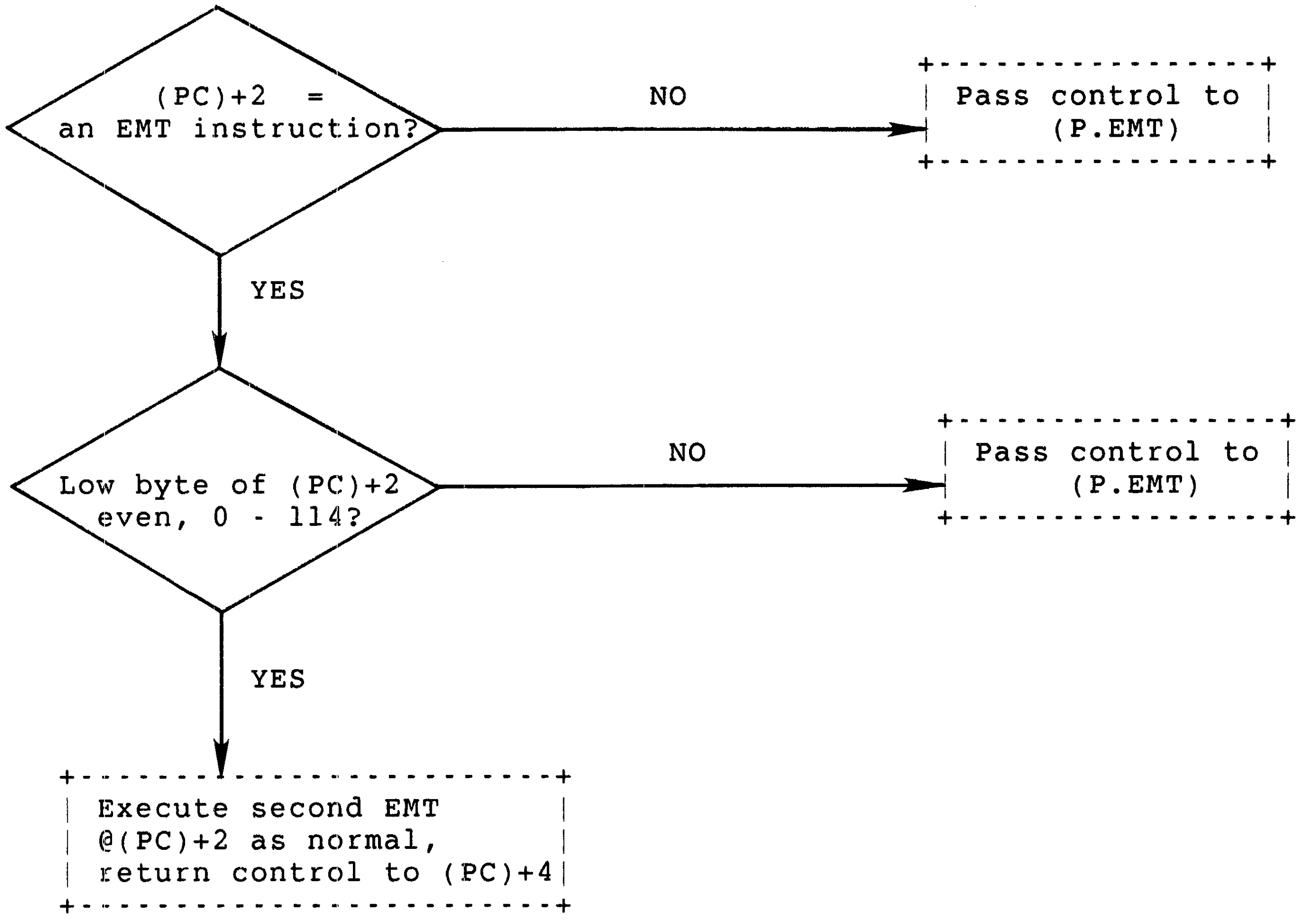
The RSX and RT11 emulator calls may be translated to more than one instruction, but the code always contains an EMT. In addition, the direct monitor calls are translated to an EMT with a low byte that is an even number within the range 0 to 114 (octal). When such an instruction is executed, control transfers directly to the monitor, the call is processed, and control returns to the instruction following the EMT.

An EMT instruction with an odd value in the range 1 to 113 in the low byte, or any value in the range 115 to 377, also transfer control to the monitor. The monitor examines the low byte, discovers that the EMT is not one of its own, and transfers control to the run-time system at the entry point defined by location P.EMT in the pseudovector region.

Now the PF.EMT bit is set to one to indicate that the run-time system wants to process EMTs that are normally processed by the monitor, that is, with an even low byte in the range 0 to 114. When PF.EMT is set to one, all EMTs cause control to pass to the run-time system at the P.EMT entry point, except those immediately preceded by a special prefix EMT -- an EMT whose low byte is equal to the low byte of P.FLAG.

Specifically, when PF.EMT equals one, the monitor handles all EMT instructions as follows:

- o Any EMT whose low byte is not equal to the low byte of P.FLAG causes control to pass through the monitor (unprocessed except for examination), and back to the run-time system at the address contained in the P.EMT word.
- o An EMT whose low byte is equal to the low byte of P.FLAG causes control to pass to the monitor, which looks at the word following the EMT with the special code; that is, at the word in location (PC)+2. Action is taken according to the value of this word:


(PC)+2 = an EMT instruction?
NO
Pass control to (P.EMT)
YES
Low byte of (PC)+2 even, 0 - 114?
NO
Pass control to (P.EMT)
YES
Execute second EMT @(PC)+2 as normal, return control to (PC)+4

In other words, the run-time system does special processing for all EMTs, except those preceded by a special prefix EMT. The RT11 run-time system uses this feature so it can emulate the RT-11 operating system's directives properly.

PF.CSZ

For a user job image executed as a result of a .RUN directive, the monitor preallocates memory based on information provided by the run-time system under which the image is executing. When this bit is set, the monitor preallocates memory based on the size of the file referenced in the .RUN directive:

space (in K words) = (filesize + 3)/4

Filesize is the number of 512-byte blocks required for the file on disk. (The division by four is performed because there are four 512-byte blocks in 1K word. The addition of three rounds any fraction of the integer divide to the next whole integer.)

When PF.CSZ is clear, the monitor preallocates memory for the image according to the value specified in the P.MSIZ word of the pseudovector region.

PF.REM

When the PF.REM bit is set, the monitor immediately removes the run-time system from memory when no job is using it. When this bit is clear, the monitor leaves the run-time system in memory until the space is actually needed by something else.

PF.NER

When this bit is set, the monitor does not log errors occurring within the run-time system to the system error log.

PF.RW

When this bit is set, the monitor maps the run-time system as read/write. (See the read/write feature of the Page Descriptor Register of an APR, in the section, "How RSTS/E Allocates Memory: Physical and Virtual Addressing.") This is a useful feature when debugging a run-time system. In normal operation, this bit should be clear, indicating that the run-time system is to be mapped read-only.

PF.1US

When the PF.1US bit is set, the monitor allows only one job to use the run-time system; that is, it is not handled as shareable code.

PF.KBM

When this bit is set, the monitor expects that the run-time system can function as a keyboard monitor. Note that the run-time system can function as a job keyboard monitor only when this bit is set. See the .RTS directive in Chapter 3 for a discussion of job keyboard monitors.

P.FLAG COMBINATIONS

The PF.1US, PF.RW, PF.NER, and PF.REM bits are useful flags when you are debugging a run-time system:

- PF.1US limits access to the run-time system to one user.
- PF.RW is necessary if you want to use the ODT routine to change memory.
- PF.NER keeps the run-time system from logging useless errors while debugging.
- PF.REM ensures that the run-time system will be reloaded each time it is used. (Otherwise, an old copy might still remain in memory when you really wanted to debug a new copy.)

P.DEXT

You can set this word to three Radix-50 characters that the monitor uses as a default runnable file type. If a .RUN directive executes with no file type given, the monitor scans its list of installed run-time systems in the order they were installed (see Chapter 3).

Note

> The order of installation shows up in the display that the SHOW RUNTIME_SYSTEM command produces.

For the first run-time system in the list (the primary run-time system), the monitor looks for a file with the file name given in the .RUN and a type that is the default type for the run-time system. (For example, the BASIC-PLUS run-time system fills this word with .BAC; RT11, with .SAV; and RSX, with .TSK. If the run-time system has no runnable file type, this word should be set to zero.) If such a file is found, it is set up for the .RUN. If no such file is found, the monitor searches for a file with the given file name and the next run-time system's default runnable file type, and so forth. Note that the order in which the file types are chosen does not depend in any way on the run-time system executing the .RUN.

P.MSIZ

The P.MSIZ word gives the minimum allowable size for a user job image, in K words, for this run-time system. The monitor uses this value as a check when the job issues a .CORE directive to change the size of the user job image in memory (see Chapter 3). The value of P.MSIZ must be an integer between 1 and the value in P.SIZE, inclusive.

P.SIZE

The P.SIZE word contains the maximum size, in K words, that a user job image can be for this run-time system. The monitor uses this value as a check when a job issues a .CORE directive to change the size of the user job image in memory. P.SIZE must be an integer between 1. and 32., inclusive. The effective upper limit is 32. minus the size of the run-time system, rounded up to a multiple of four. (Remember that the APR mapping requires that space for the run-time system be allocated in units of 4K words.) Thus, a run-time system that required 5K words could set an upper limit here of 24. (32.-8.). However, it could set P.SIZE to some smaller value.

Synchronous Exception Handler Addresses

These mnemonics refer to locations in the run-time system where control is to pass for synchronous exceptions.

P.FIS

The monitor interprets the P.FIS word as the trap address for the hardware floating-point instruction set available on the PDP-11/35 and 40. Whenever an instruction from this set is executed that causes a trap to the kernel mode vector at 244, the monitor passes control to the run-time system at the location specified by the contents of the P.FIS word.

This exception pushes two words onto the user's SP stack: the contents of the PC and PS registers at the time of the exception. For example:

```
SP⟶ (PC) at the time of the exception
     (PS) at the time of the exception
     word to which SP pointed before the exception
```

Whatever action the run-time system wants to take for this exception should be done at the location specified by the contents of P.FIS. A return from interrupt (RTI) instruction returns control to the point where it was when the exception occurred.

P.BAD - Synchronous Exceptions

The monitor passes control to the run-time system at the location specified by the contents of P.BAD when any of the following synchronous exceptions occur:

- o Memory management unit exception (trapped to kernel mode vector at 250).
- o The job tries to execute a reserved instruction (trapped to kernel mode vector at 10).

- o The job issues an instruction with an odd address (trapped to kernel mode vector at 4).

This exception pushes two words onto the user's SP stack: the contents of the PC and PS registers at the time of the exception. For example:

```
SP──> (PC) at the time of the exception
      (PS) at the time of the exception
      word to which SP pointed before the exception
```

The monitor returns an error code in the first byte of the FIRQB so the run-time system can determine which error occurred. The error codes are:

B.4	Odd address
B.10	Reserved instruction
B.250	Memory management unit exception

The run-time system is responsible for processing these errors in whatever manner it sees fit. In general, most run-time systems provided with RSTS/E systems report the error, using the UU.ERR subfunction of the .UUO directive and perhaps print the PC value from the top of the stack. You can use an RTI instruction to return control to the point where it left off when the exception occurred. Note that some asynchronous exceptions also use this address.

P.BPT

The P.BPT word contains the exception address for a BPT instruction and for T-bit exceptions. When the job issues a BPT instruction or a T-bit exception occurs (to the kernel mode vector at 14), the monitor passes control to the run-time system for the job at the address specified by the contents of this word.

This exception pushes two words onto the user's SP stack: the contents of the PC and PS registers. For example:

```
SP──> (PC) at the time of the exception
      (PS) at the time of the exception
      word to which SP pointed before the exception
```

The run-time system processes these exceptions in any fashion it sees fit at the location specified by the contents of P.BPT. The RTI or RTT instructions can be used to return control to the user's program at the point where it was when the exception occurred.

P.IOT

The P.IOT word contains the exception address for an IOT instruction. Whenever the job issues an IOT instruction (trapped to kernel mode vector at 20), the monitor passes control on to the run-time system at the address specified by the contents of this word.

This exception pushes two words onto the user's SP stack: the contents of the PC and PS registers at the time of the exception. For example:

```
SP──> (PC) at the time of the exception
      (PS) at the time of the exception
      word to which SP pointed before the exception
```

The run-time system can process the exception in any fashion it sees fit. You can use an RTI instruction to return control to the point where it was when the exception occurred.

P.EMT

This word contains the location to which control is transferred for nonmonitor EMT instructions; that is, for EMT instructions whose low byte is odd within the range 1 to 113 or any value in the range 115 to 377. If the PF.EMT bit is set in the P.FLAG word in the pseudovector region, control is transferred here for all EMT instructions except those preceded by the special prefix EMT, as described previously.

The exception pushes two words onto the user's SP stack: the contents of the PC and PS registers at the time of the exception. For example:

```
SP──> (PC) at the time of the exception
      (PS) at the time of the exception
      word to which SP pointed before the exception
```

The run-time system is responsible for processing the EMT as it sees fit. You can use the RTI instruction to return control to the point where it was when the exception occurred.

Note

All EMT instructions are reserved for use by DIGITAL.

P.TRAP

This is the location to which control is transferred for all TRAP instructions (operation codes 104400 through 104777, inclusive). Whenever the job executes such an instruction (trapped to kernel mode vector 34), the monitor passes control to the run-time system at the location specified by the contents of this word.

This exception pushes two words onto the user's SP stack: the contents of the PC and PS registers at the time of the exception. For example:

```
SP──> (PC) at the time of the exception
      (PS) at the time of the exception
      word to which SP pointed before the exception
```

The run-time system is responsible for processing the exception as it sees fit. You can use an RTI instruction to return control to the point where it was when the exception occurred.

Asynchronous Exception Handler Addresses

These mnemonics refer to locations within the run-time system where control is to pass for asynchronous exceptions.

P.FPP

This location is the exception address for the hardware floating-point processor (FPP) for the PDP-11/34A, 44, 45, 50, 55, 60, 70, 73, and 84 asynchronous unit or the KEF11-AA or FPF-11 for the PDP-11/23-PLUS and 24. Whenever the unit takes an exception trap (to kernel mode vector at 244), the monitor passes control to the run-time system at the location specified by the contents of this word. The Floating-point Exception Code (FEC) and Floating-point Error Address (FEA) of this unit are not otherwise accessible.

Therefore, the monitor pushes these two values onto the user's SP stack, in addition to the contents of the PC and PS registers at the time of the interrupt. For example:

```
SP—→ FEC
     FEA
     (PC) at the time of the exception
     (PS) at the time of the exception
     word to which SP pointed before the exception
```

The run-time system can process the exception as appropriate, clean the stack (remove the FEC and FEA), and issue an RTI instruction to return control to the user's program at the point where it was when the exception occurred.

P.CC

This is the location to which control passes when a CTRL/C is entered at any terminal on any channel that this job accepts. The monitor stops further programmed output for the job (CTRL/O effect) and cancels any pending character output.

The user's SP stack is modified at entry. For example:

```
SP—→ (PC) at the time of the exception
     (PS) at the time of the exception
     word to which SP pointed before the exception
```

The run-time system can process the CTRL/C as you want. All run-time systems supplied with a RSTS/E system abort the job, unless the user job image has indicated that it wants to handle CTRL/C traps itself (see the SCCA$S and the .SETCC directives).

P.2CC

This is the exception address taken when the user enters a second CTRL/C before the run-time system has been able to respond to the first CTRL/C. (That is, the monitor has received two CTRL/Cs before it has been able to pass control to the run-time system at the location specified by the contents of P.CC in the time-sharing environment.)
As with one CTRL/C, when the P.2CC point is entered, further programmed output is canceled (CTRL/O effect), and any pending character output is canceled. Two words are pushed onto the user's SP stack. For example:

```
SP──> (PC) at the time of the exception
      (PS) at the time of the exception
      word to which SP pointed before the exception
```

The run-time system can process the condition as you want (BASIC-PLUS exits immediately, returning control to the P.NEW entry point in the default keyboard monitor). An RTI instruction would return control to the point where the program left off, but this annoys the user who entered the two CTRL/Cs expecting to get out.

P.BAD - Asynchronous Exceptions

The monitor passes control to the location specified by P.BAD whenever any of the following asynchronous errors occur:

- The user's SP stack overflows.
- A fatal disk error occurs when the job is swapped. The original contents of the user job image are lost.
- A memory parity fault occurs in the user job image. The original contents of the user job image are lost.
- A fatal disk error occurs when a run-time system or resident library is loaded. Control passes to P.BAD in the default keyboard monitor when the load error occurs for a run-time system.

None of these errors are recoverable. An error is returned in the first byte of the FIRQB to indicate which error occurred, KEY is refreshed, and the contents of the general registers (R0 through R5) are random. SP is reset to the value USRSP.

In general, most run-time systems provided with RSTS/E systems report the error, using the UU.ERR subfunction of the .UUO directive, and also the ??Program lost-sorry message (UU.ERR call with FUCORE value). Then, the run-time systems exit to the job keyboard monitor, using .RTS. The ??Program lost-sorry message prints to warn you that user logical values have been destroyed.

The error codes that RSTS/E returns in the first byte of the FIRQB are:

B.STAK	The user's SP stack overflowed
B.SWAP	Fatal disk error on swap
B.PRTY	Memory parity fault
NRRTS	Fatal disk error on run-time system or resident library load

Control is also transferred to P.BAD for some synchronous exceptions.

Entry Points

These mnemonics refer to locations within the run-time system where control is to pass at certain transition points for the job.

P.NEW

The monitor passes control to this entry point under the assumption that new user or next request processing is to be done. Compare this to the P.RUN entry point, where a specific program is to be run under this run-time system. P.NEW is commonly used as the entry point to switch back to a job's keyboard monitor. For example, the .EXIT directive passes control to this entry point in the system default keyboard monitor. You can use the .RTS directive to pass control to P.NEW in a job's keyboard monitor or a specifically named run-time system. Note that a job can establish its own job keyboard monitor, which is different from the default keyboard monitor (see Chapter 3).

By examining KEY and the XRB, the run-time system can determine how and by whom it was entered at P.NEW, if this is significant. (Run-time systems that do not have keyboard monitors would probably want to exit (using .EXIT) to the default keyboard monitor at P.NEW.)

The three conditions under which control passes to the P.NEW entry are:

- o Brand new job on the system -- In this case, JFNOPR (bit 12 in KEY) is set (the job is not yet logged in), and the words at location XRB+2 and XRB+4 are zero (the monitor requested the entry, not a run-time system). This indicates that the monitor has passed control to this location, having received input over channel zero in a logged-out state (occurs only for the primary run-time system).

  The run-time system should run some predetermined program to read (.READ directive) the input line that the monitor has buffered. For example, DCL executes SY:[1,2]LOGIN.* (the LOGIN utility) in this case.

- o Switch to this run-time system when job logged out -- In this case, JFNOPR (bit 12 in KEY) is set, and the name of the calling run-time system is given as two RAD50 words in locations XRB+2 and XRB+4. The calling run-time system is the run-time system under whose control the directive was issued that caused the switch.

  For this case, the run-time system should issue a logged-out prompt message. For example, the BASIC-PLUS run-time system prints "Bye" and returns control to the monitor. (Normally, control does not pass to the run-time system in this case. If LOGIN does not recognize the line that it read (as in previous case), it kills itself, destroying the job and returning control to the monitor.)

- o Switch to this run-time system when job logged in -- In this case, JFNOPR (bit 12 in KEY) will be clear. The name of the calling run-time system is given as two words of RAD50 in locations XRB+2 and XRB+4 or is zero if this job was just created by UU.JOB (see Chapter 3).

  For this situation, the run-time system should issue its logged-in prompt and attempt to read the next command from the terminal open on channel zero. BASIC-PLUS prints "Ready", DCL prints "$", RSX prints ">", and RT11 prints ".". Then, all wait for further input.

Keyboard monitors should read channel zero (the job's terminal) using the keyboard monitor wait feature of .READ. The monitor will kill jobs that execute this read in a logged-out state; otherwise, it is an infinite-wait read.

The monitor usually does some housekeeping for the job at the time the P.NEW entry point is entered. Specifically, the word at location FIRQB+FQJOB is always set to two times the job number assigned by the monitor when the job was created, and KEY is refreshed with current information about the job. Third-part privilege checking is turned off if it was on (see UU.3PP). Furthermore, SP is reset to 400 (see USRSP description), and all the general registers (R0 through R5) contain zero. Note, however, that I/O channels are left open. Therefore, the run-time system should reset all I/O channels.

On exit from privileged programs, some additional clean-up is done:

- o Temporary privileges are dropped
- o User memory is cleared (upwards from location 1000)
- o User job image size is reset to the value in P.MSIZ

Note

> This housekeeping is not done if a specific request is made to pass control to a run-time system without changing the job-context information (see the .RTS directive).

The following information exists in the XRB at the time the P.NEW entry point is entered:

XRB on P.NEW Entry

```
Mne-  Octal                                              Octal   Mne-
monic Offset                                              Offset  monic
          +--------------------------------------------+
       1  | 1 for switch without housekeeping; else 0  |  0
          |                                            |
       3  +--------------------------------------------+  2
          |      name of the calling run-time system   |
       5  |          (2 words in RAD50 format)         |  4
          +--------------------------------------------+
       7  |      -1 if calling RTS = new RTS; else 0   |  6
          |                                            |
      11  +--------------------------------------------+ 10
          |           whatever values were here        |
      13  |           when the switch was made         | 12
          |                                            |
      15  +--------------------------------------------+ 14
```

XRB+0	This word contains a value of one, if control was transferred by an .RTS directive using the switch without changing the job-context option (see Chapter 3).
XRB+2	The two words beginning here contain the name of the calling run-time system, in RAD50 format. If control was transferred here directly by the monitor, these two words contain zero.
XRB+6	This word contains minus one if the calling run-time system is the same as the one that now has control. This word is zero otherwise; that is, if the calling run-time system is not the same as the called run-time system.
XRB+10	The contents of the next three words will be the same as they were when the switch occurred. That is, data can be passed from run-time system to run-time system here. If control has been transferred to P.NEW directly by the monitor, these three words are zero.

P.RUN

The monitor passes control to the P.RUN entry point when an executable program is to be run for a job under control of this run-time system. This can occur as the result of either a .RUN or a .CHAIN directive (in which a job has directly asked for a file to be run) or a .CCL directive (in which a job has asked the monitor to check a string to see if it is a valid CCL command, and if so, execute the appropriate file).

The monitor opens the file to be run (a disk file) on channel 15. However, the file has not been read; it is up to the run-time system to load and execute the file. The run-time system should also reset all I/O channels except 15, in case they are open.

The monitor performs the same housekeeping operations as with P.NEW (reset the stack, and so on). In addition, if the program to be run is a privileged program, and the caller does not have all of the program's privileges, the monitor sets the JFSYS bit in KEY, saves the current privileges, and adds the program's privileges to the current privileges.

The monitor passes data to the run-time system in the XRB, FIRQB, and KEY areas of the user job image (low segment).

XRB on P.RUN Entry

```
Mne- Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
          +-------------------------------------------+
        1 |   flag bits describing entry conditions   |  0
          +-------------------------------------------+
        3 |                                           |  2
          |   name of run-time system which issued    |
        5 |          the call to this one             |  4
          +-------------------------------------------+
        7 |               random value                |  6
          +-------------------------------------------+
       11 |                                           | 10
          |                                           |
       13 |      same value as when the caller        | 12
          |         issued the .RUN or .CCL           |
       15 +-------------------------------------------+ 14
```

XRB+0 This word contains flag bits that describe the entry conditions. (The STATUS variable in BASIC-PLUS returns these values.)

Bit	Meaning
15	When set, indicates the entry was made as the result of a .CCL directive. When clear, indicates the entry was made as the result of a .RUN or a .CHAIN directive.
14	When set, indicates the caller issued a .CCL directive with a /DETACH switch, with the intent that this run-time system executes the file in detached mode. It is up to the run-time system to take action on this flag. You can detach a job by using the UU.DET subfunction of the .UUO directive; see Chapter 3.
13	When set, indicates that the caller issued a directive with a /SIZE switch; that is, the file is to be run at a specific size. The size is given in bits 0-7. It is up to the run-time system to set the size as indicated (see the .CORE directive).
12-8	Reserved for future use.
7-0	If the value of these bits is zero, then no special size for this program run is called for. If the value is greater than zero, it indicates the size, in K words, that the program should be run at. If the value is less than zero, the absolute value indicates an increment, in K words, to the size that the program would normally run at.

XRB+2 These two words contain the name of the run-time system under which the .RUN, .CHAIN, or .CCL directive to this run-time system was issued, in RAD50 format.

XRB+6 The contents of this word are random.

XRB+10 The three words beginning here contain the same information that they held when the job issued the .RUN, .CHAIN, or .CCL directive.

FIRQB on P.RUN Entry

```
Mne-    Octal                                              Octal   Mne-
monic   Offset                                             Offset  monic
          +--------------------+--------------------+
        1 |////////////////////////////////////////|  0
          +--------------------+--------------------+
        3 |////////////////////|   job number * 2   |  2 FQJOB
          +--------------------+--------------------+
        5 |////////////////////////////////////////|  4
          +--------------------+--------------------+
        7 |  project number    | programmer number  |  6 FQPPN
          +--------------------+--------------------+
       11 |      file name in RAD50 format          | 10 FQNAM1
          |              (2 words)                  |
       13 |                                         | 12
          +--------------------+--------------------+
       15 |   file type in RAD50 format (1 word)    | 14 FQEXT
          +--------------------+--------------------+
       17 |      file size in 512-byte blocks       | 16 FQSIZ
          +--------------------+--------------------+
       21 |      default buffer size for disk       | 20 FQBUFL
          +--------------------+--------------------+
       23 |////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |           device description            | 24 FQFLAG
          +--------------------+--------------------+
FQPROT 27 |  protection code   |    cluster size    | 26 FQPFLG
          +--------------------+--------------------+
       31 |    device name (2 ASCII characters)     | 30 FQDEV
          +--------------------+--------------------+
       33 |     flag byte      | device unit number | 32 FQDEVN
          +--------------------+--------------------+
       35 |       file identification index         | 34 FQCLUS
          +--------------------+--------------------+
       37 |            entry parameter              | 36 FQNENT
          +--------------------+--------------------+
```

FIRQB+FQJOB	The job number (assigned by the monitor when this job was created) times two.
FIRQB+FQPPN	The project-programmer number for the file that is to be run.
FIRQB+FQNAM1	The name of the file that is to be run, as two words in RAD50 format.
FIRQB+FQEXT	The type of the file that is to be run, as one word in RAD50 format.
FIRQB+FQSIZ	The file size, in 512-byte blocks.

FIRQB+FQBUFL	The recommended size, in bytes, for the buffer size in a .READ directive for this file.
FIRQB+FQFLAG	Flag bits defining the device. They are set to indicate that this is a disk file. (See the FQFLAG description in the open function of the CALFIP directive.)
FIRQB+FQPROT-1	The file cluster size, modulo 256. (That is, a file cluster size of 256 is indicated by a zero byte here.) This byte is the same as the FQCLUS value supplied in the open functions of the CALFIP directive, except that it is returned in a byte instead of a word.
FIRQB+FQPROT	The protection code of the file.
FIRQB+FQDEV	The device name of the disk device, as two ASCII characters.
FIRQB+FQDEVN	The unit number of the disk device.
FIRQB+FQDEVN+1	The low-order two bits of this byte are set to indicate whether or not the device is part of the public structure:

Bit 0 = 0	The device is in the public structure.
= 1	The device is a private disk.
Bit 1 = 0	A specific device was not specified in the open function.
= 1	A specific device was specified in the open function.

FIRQB+FQCLUS — The file identification index of this file. This word is significant in that you can use it in place of the file name in subsequent opens of the file on disk. You can open the file with the OPNFQ subfunction of CALFIP, using:

- o An explicit PPN in FIRQB+FQPPN
- o A zero word in FIRQB+FQNAM1
- o An explicit device name in FQDEV and FQDEVN
- o The file identification index in FIRQB+FQNAM1+2.

FIRQB+FQNENT	Parameter word from the caller. The .RUN or .CHAIN directive, which causes entry at P.RUN in a run-time system, allows the caller to specify a parameter word to be passed to the run-time system. Bit 15 of this word may or may not be the same as the caller passed, however. If the .RUN or .CHAIN directive was issued from a privileged program with temporary privileges enabled, bit 15 is passed by the caller. Otherwise, bit 15 is cleared. For .CCL entries, bit 15 is set by the monitor. If the CCL definition for the CCL being executed has the /PRIVILEGED qualifier included in it, the bit 15 is set; otherwise, it is cleared.

PART II Monitor Directives

Introduction

This chapter describes the general directives to the RSTS/E monitor. These directives are available to the MACRO programmer under both the RSX and RT11 run-time systems. They are Emulator Trap (EMT) instructions that are processed directly by the monitor. A run-time system does not examine or process these general directives.

Summary of General Monitor Directives

Table 3-1 summarizes the general monitor directives. Detailed descriptions are given in the sections that follow this introductory material. The descriptions are arranged alphabetically by mnemonic name.

Some directives, which cause a change in run-time system or change in job size, should not be executed by a program (user job image) running under the RT11 run-time system, or unpredictable results may occur. These directives are marked with an asterisk (*) in Table 3-1. (Note, however, that these are not restrictions for assembling using MACRO -- the assembler for the RT11 run-time system. If you are coding a run-time system, you use these directives and assemble under either MACRO or MAC.)

Table 3-1: Summary of General Monitor Calls

Name	EMT Code (Octal)	Description
CALFIP	0	Call the File Processor portion of the RSTS/E monitor. Includes "housekeeping" functions for file/device I/O such as OPEN, CLOSE.
.READ	2	Read from a previously opened file or device.
.WRITE	4	Write to a previously opened file or device.
.CORE*	6	Change memory size allocated for user job image.
.SLEEP	10	Sleep job for n seconds.
.PEEK	12	Peek at the monitor's memory.

Table 3-1: Summary of General Monitor Calls (Cont.)

Name	EMT Code (Octal)	Description
.SPEC	14	Special function.
.TTAPE	16	Enter tape mode.
.TTECH	20	Enable echo on a channel.
.TTNCH	22	Disable echo on a channel.
.TTDDT	24	Enter ODT submode on a channel.
.TTRST	26	Cancel CTRL/O effect.
.TIME	30	Get timing information.
.POSTN	32	Get device's horizontal position.
.DATE	34	Get current date.
.SET	36	Set keyword bits.
.STAT	40	Get statistics for job.
.RUN*	42	Run new program (user job image).
.NAME	44	Install a new program name.
.EXIT*	46	Exit to default keyboard monitor.
.RTS*	50	Switch to new run-time system.
.ERLOG	52	Log an error from run-time system.
.LOGS	54	Check for logical devices.
.CLEAR	56	Clear keyword bits.
.MESAG	60	Message send/receive.
.CCL*	62	Check string to see if Concise Command Language (CCL).
.FSS*	64	Scan a string for valid RSTS/E file specifications.

Table 3-1: Summary of General Monitor Calls (Cont.)

Name	EMT Code (Octal)	Description
.UUO	66	Execute monitor FIP call (access to BASIC-PLUS SYS calls to FIP).
.CHAIN**	70	Execute user job image under same run-time system.
.PLAS	72	Access a shared library.
.ULOG	76	Assign/reassign/deassign device or user logical.
.READA	102	Asynchronous read.
.WRITA	104	Asynchronous write.
.ASTX	106	AST exit.
.CMDLN	112	Read/write command line buffer.
.AST	114	AST enable/disable.

* This directive should not be used by a program running under control of either the RT11 run-time system or the RSX emulator.

** These directives should not be used by a program running under control of the RT11 run-time system.

Prefix File COMMON.MAC

The monitor directives that this chapter describes require that you pass parameters to the monitor in the FIRQB and XRB; values are also returned to your program in these areas. The file COMMON.MAC, provided with all RSTS/E kits, relates mnemonics to often-used addresses, offset values, and function codes, eliminating the need for octal coding and addressing. These mnemonics are used in the directive descriptions that follow; DIGITAL recommends their use for readable, maintainable code.

How to Assemble with COMMON.MAC

COMMON.MAC is a prefix file; it is assembled with your other MACRO source files under either the RSX or RT11 run-time systems. For example, under the RT11 run-time system, the following sequence would assemble the files COMMON.MAC, SRC1.MAC, and SRC2.MAC into the object module file OBJ.OBJ with an assembly listing file OBJ.LST:

```
RUN $MACRO
*OBJ,OBJ=COMMON,SRC1,SRC2
```

Similarly, under the RSX run-time system, this sequence would assemble the files COMMON.MAC, SRC1.MAC, and SCR2.MAC into the object module file OBJ.OBJ with an assembly listing file OBJ.LST:

```
RUN $MAC
MAC>OBJ,OBJ=COMMON,SRC1,SRC2
```

Macros Provided in COMMON.MAC

In addition to providing mnemonics, the COMMON.MAC file contains macros that can be used in programs assembled under either the RSX or RT11 run-time systems, as long as COMMON.MAC is assembled with the source, as described previously. These macros are:

TITLE name,desc,nn,date,editors

The TITLE macro sets a title (.TITLE) from the name and description (desc) parameters and builds an identification (.IDENT) from the specified number nn. The IDENT has the form xx.xnn, where xx.x is the current RSTS/E version number (09.0 for V9.0), and nn is the edit level you specify. Descriptive information is placed in the table of contents as follows:

```
EDIT:         DATE:         BY:
 nn           date          editors
```

ORG section[,offset]

ORG defines the origin address of a program section. The first occurrence of an ORG with a given section name causes all instructions requiring memory space following the ORG to be assigned consecutive relocatable addresses starting with zero or, if an offset is given, with the octal address given. Following occurrences of an ORG with the same section name causes resumption of addressing wherever it left off before, because of an intervening ORG.

The ORG macro also defines a symbol with the same name as the section at the first relative location within the section. Every invocation of ORG also defines (or redefines) the section to be returned to by the macro UNORG.

DEFORG	section

The DEFORG macro is the same as the ORG macro except that the symbol at relative 0 (the section name) is declared as a global symbol. By convention, the module that defines the section (rather than just uses it) issues the DEFORG macro.

TMPORG	section[,offset]

TMPORG is the same as ORG except that it does not define (or redefine) the section to be reentered by the UNORG macro. In this way, the module can temporarily enter a new section and then return to the main section using UNORG without having to know the main section name.

UNORG	The UNORG macro will reenter the section most recently declared in an ORG or DEFORG macro.

INCLUDE	name1[,name2,...]

The INCLUDE macro indicates that the module issuing the INCLUDE requires the named modules (name1, ...). The name(s) should be declared with DEFORG(s) in the required modules.

INCLUDE declares the listed section names as global symbols and issues the macro directive .SBTTL with the heading "INCLUDE FROM LIBRARY 'name'" to be inserted in the assembly listing table of contents. INCLUDE documents the named sections as required by this section.

.DSECT	[start][,cref]

The .DSECT macro starts a dummy program section (with the MACRO directive .ASECT) at relocatable address 0 or at the address given by the optional argument start. If the cross-reference (cref) parameter is given (nonblank), the program section is included in the cross-reference listing, if you request one for the assembly.

The .DSECT macro is used in the file COMMON.MAC to define the system parameters and offsets.

For example, coding of this form is used in COMMON.MAC to assign the proper values to the mnemonics in the pseudo-vector region:

```
.DSECT    177776,NOCREF
          .BLKW    -1
P.SIZE:   .BLKW    -1
P.2CC:    .BLKW    -1
P.CC:     .BLKW    -1
    .
    .
    .
```

Note

A .DSECT is used at the end of the file COMMON.MAC. This means that you must explicitly start your MACRO program with an ORG macro or .PSECT directive to begin your program at relocatable address 0. Otherwise, your code will be regarded as a continuation of the .DSECT, and the program will not assemble properly.

.BSECT   [HIGH][,cref]

The .BSECT macro is like the .DSECT macro except that the default starting address is 1 instead of 0. If the argument HIGH is used, the starting address is 400 (octal). This starting address lets you use .BSECT to generate bit values. The .BSECT macro is used in COMMON.MAC to define mnemonics for bit locations. For example, the following coding assigns the mnemonics to the bit locations in the keyword (KEY; see Chapter 2). Note that the period (.) after .BLKB is required.

```
.BSECT     HIGH,NOCREF
JFSPR1:    .BLKB   .
JFPP:      .BLKB   .
    .
    .
    .
```

.EQUATE   symbol,value

.EQUATE defines the given symbol to have the supplied value (which may be an expression) by using the equivalent of:

```
.DSECT     value
.symbol:
UNORG
```

.BLKW0	[amount][,value]
	The .BLKW0 macro is similar to the MACRO directive .BLKW, which reserves a specified number of words of storage space. The amount can be any expression, the default is one. While .BLKW just reserves space, .BLKW0 fills the space with the value you specify; the default is zero.
.BLKB0	[amount][,value]
	The .BLKB0 macro is similar to the MACRO directive .BLKB, which reserves a specified number of bytes of storage space. The amount can be any expression, the default is one. .BLKB0 fills the space with the value you specify; the default is zero.
GLOBAL	<name1[,name2,...]>
	GLOBAL declares the name symbols as external global symbols.
RETURN	[register]
	The RETURN macro generates an RTS PC by default but can generate any other RTS instruction if you specify an explicit register.
JMPX	label
	JMPX is just like the JMP instruction but will also declare the label to be an external global (that is, jump external).
CALL	subroutine[,register[,argument list]]
	You can use CALL instead of JSR PC to call subroutines. If an explicit register is specified, then the call is JSR using that register. If an argument list is specified, it generates a list of .WORD arguments in line with the subroutine call.
CALLR	subroutine
	CALLR is equivalent to a CALL to a subroutine immediately followed by a RETURN. CALLR generates a JMP instruction.
CALLX	subroutine
	CALLX is just like CALL, but it also declares the subroutine name as an external global symbol.

CALLRX	subroutine CALLRX is just like CALLR except that the subroutine name is declared as an external global symbol.

Error Mnemonics: Symbol Table File ERR.STB

When the monitor processes the directives that this chapter describes, any errors that it detects are passed back to the job in the first byte of the FIRQB as a binary value. The ERR.STB file, provided with all RSTS/E kits, relates mnemonic values to these binary codes, so you do not have to analyze and process errors in octal. The descriptions in this chapter all refer to the mnemonics provided by ERR.STB. See Appendix A for a list of all possible errors.

The symbols are automatically resolved at link time if you include ERR.STB with the files you link with either TKB (the Task Builder for the RSX run-time system) or LINK (the linker for the RT11 run-time system). For example, under the RSX run-time system, the following code links ERR.STB and MAIN.OBJ to produce the executable file IMG1.TSK, a memory allocation file MP1.MAP, and a symbol definition file SF1.STB:

```
  RUN $TKB
  TKB>IMG1,MP1,SF1=ERR.STB,MAIN
```

Similarly, under the RT11 run-time system, the following sequence links ERR.STB and MAIN.OBJ to produce the executable file IMG1.SAV, a memory allocation file MP1.MAP, and a symbol definition file SF1.STB:

```
  RUN $LINK
  *IMG1,MP1,SF1=ERR.STB,MAIN
```

Programming Hints

Preset the FIRQB and XRB to Zero

The monitor directives in this chapter pass information to the monitor in the FIRQB and XRB areas of the low 512. bytes of memory. It is usually a good idea to clear the entire FIRQB and XRB before issuing a call, to ensure that no extraneous information (for example, from data returned on a call) has been left in the areas that could affect how the call executes.

In some cases, however, you may want to leave the FIRQB and XRB alone. The .FSS call, for example, scans a string and, if it is a valid file specification, returns to the FIRQB the information needed to open the file with the CALFIP call. You do not want to clear the FIRQB before opening the file with CALFIP.

Note

> To ensure compatibility with future releases of RSTS/E, you should always set to zero any fields in the FIRQB and XRB diagrams that are shaded or are documented as reserved or not used.

The following example contains three routines that clear the FIRQB and XRB:

CLRFQX	Clears both the FIRQB and XRB.
CLRFQB	Clears the FIRQB.
CLRXRB	Clears the XRB.

The values FQBSIZ and XRBSIZ used in these routines are defined in COMMON.MAC.

```
          .ENABL  LSB

CLRFQX::  PUSH    <R0,R1>                ;Save R0,R1
          MOV     #FIRQB,R0              ;Point to FIRQB.
          MOV     #<<FQBSIZ+XRBSIZ>/2>,R1 ;Compute how many words
          BR      10$                    ;to clear.

CLRFQB::  PUSH    <R0,R1>                ;Save R0,R1
          MOV     #FIRQB,R0              ;Point to FIRQB.
          MOV     #<FQBSIZ/2>,R1         ;Compute how many words
          BR      10$                    ;to clear.

CLRXRB::  PUSH    <R0,R1>                ;Save R0,R1
          MOV     #XRB,R0                ;Point to XRB
          MOV     #<XRBSIZ/2>,R1         ;Compute how many words
                                         ;to clear.

10$:      CLR     (R0)+                  ;Zero it out ...
          SOB     R1,10$                 ;'til all done
          POP     <R1,R0>                ;Restore R0,R1.
          RETURN

          .DSABL  LSB
```

Data Returned to FIRQB and XRB

If a call completes without error, the monitor sets byte 0 of the FIRQB to zero. If an error occurs on a call, the monitor sets byte 0 of the FIRQB to an error code. Likewise, the monitor always sets the byte at FIRQB+2 to the current job number times two when a call completes.

In some circumstances, it may be useful to know what happens to the passed-data when a call completes. For instance, is the file name still there? Bytes not specified as containing returned-data are undefined. Do not rely on these values when coding your programs because DIGITAL reserves the right to change the values returned in these bytes at any time. In addition, if an error occurs, the data returned may or may not have replaced the data passed. It depends on how far processing for the call got before the error occurred.

Channel Numbers for I/O

Directives that handle I/O use a channel number to refer to a device. In device or file opens, a channel number is related to a specific device defined in the call. Directives that transfer data (.READ, .WRITE, .READA, .WRITA) can then refer to a channel number rather than define a device or file.

Valid channel numbers range from 0 through 17. Channel 0 is the job's terminal; for example, a .WRITE to channel 0 writes to the terminal which is running the job. Channel 0 is always open. Similarly, the monitor opens a file to be run on channel 17 when control transfers to the P.RUN entry point in a run-time system. Thus, user jobs may define and use channels 1 to 16.

Directives That Do I/O

The CALFIP subfunctions OPNFQ, CREFQ, CRBFQ, and CRTFQ open a file or device and relate the specified channel number to that file or device:

- OPNFQ opens a file or device for input
- CREFQ creates a file, that is, opens a file or device for output
- CRBFQ creates a binary (executable) output file on disk
- CRTFQ creates a temporary file on disk

The directives .READ and .WRITE transfer data between memory and a device or file specified by channel number.

The CLSFQ (close) and RSTFQ (reset) subfunctions of CALFIP close a device or file and free the associated channel number so it can be used with another device or file.

Directives That Support I/O

The file string scan (.FSS) directive is useful for programs that process files specified by a terminal user. The .FSS directive examines a string of characters and, if it is a valid RSTS/E file specification, converts it to the FIRQB format used to open a file. Thus, your program can accept a typed string from the job's terminal and use .FSS to convert the string to the FIRQB format to do I/O on the file.

You can use the LOKFQ subfunction of CALFIP to search for disk files that meet wildcard file specifications. For example, you could search an account on disk for all files with names beginning with the characters DD.

.AST -- Enable/Disable AST Delivery

Form

```
   .AST
```

Function

The .AST directive has two functions: disable AST delivery and enable AST delivery. The .AST disable function stalls all AST deliveries from the monitor until the user explicitly enables them with the .AST enable directive.

Privileges Required

None

Data Passed

The only data passed for this directive is the byte 0 of the FIRQB. If zero, enable AST deliveries. If minus one, disable AST deliveries.

Data Returned

Except for a possible error code in byte 0 of the FIRQB, this directive does not return any meaningful data.

Errors

BADFUO	Illegal function code.

.ASTX -- Exit from AST Routine

Form

```
.ASTX
```

Function

The .ASTX directive is similar to a RETURN in a normal subroutine. It instructs the monitor that the asynchronous routine has completed and that control should return to the job at the point it was interrupted. All AST routines must finish with a .ASTX directive.

When an AST routine issues the .ASTX directive, the PSW previously stored on the user's stack is validated. The PSW is forced into the standard mode (previous user/current user mode, register set 1, and priority 0). The PC and PSW are then used to return control to the program at the point where it was interrupted.

The XRB and bytes 0, 6, and 7 of the FIRQB are restored. If the user has altered the PC, PSW, or destroyed the stack contents, no specific error is returned. Instead, bad PC and/or stack causes fatal errors which are handled in the usual way (entry to P.BAD in the RTS).

To make sure this directive behaves correctly, the AST routine must make sure all general registers (R0 to R5) and the stack pointer have the same contents as on entry to the AST routine. In addition, if any part of the FIRQB other than bytes 0, 6, and 7 has been used, the AST routine must restore the contents of the FIRQB to what it had on entry. Failure to observe these rules may produce unexpected results.

CALFIP -- Call the File Processor

Form

CALFIP

Function

The CALFIP directive to the RSTS/E monitor handles housekeeping necessary for input/output on RSTS/E. For example, CALFIP lets you open a channel for file or device I/O.

You select the particular function by setting a function field in the FIRQB (at offset FQFUN). Other parameters are also passed to the monitor in the FIRQB, depending on the function requested.

Table 3-2 lists the CALFIP subfunctions by function code. The sections following Table 3-2 describe the subfunctions in alphabetical order.

Table 3-2: Summary of CALFIP Subfunctions

FQFUN Value (Octal)	Mnemonic	Action Performed (BASIC-PLUS Equivalent)
0	CLSFQ	Close an open channel (CLOSE)
2	OPNFQ	Open a channel (OPEN FOR INPUT)
4	CREFQ	Create/extend a file (file-structured OPEN FOR OUTPUT)
6	DLNFQ	Delete a file by name (KILL)
10	RENFQ	Rename a file (NAME...AS)
12	DIRFQ	Get directory information
14	UUOFQ	Process UUO
16	ERRFQ	Get error message text
20	RSTFQ	Reset (close) a channel or all channels (except channel 0)
22	LOKFQ	Look up a file
24	ASSFQ	Allocate a device
26	DEAFQ	Deallocate a device
30	DALFQ	Deallocate all devices
32	CRTFQ	Create/extend a temporary file on disk
34	CRBFQ	Create/extend a compiled image file on disk (file-structured OPEN FOR OUTPUT, protection code bit 6 always set)

ASSFQ (Allocate a Device)

Form

```
MOVB #ASSFQ,FIRQB+FQFUN
               .
               .
               .
(Set up FIRQB to define device)
               .
               .
               .
CALFIP
```

Function

The ASSFQ subfunction reserves a physical device for a job or transfers assignment of a currently owned device to another job.

Privileges Required

DEVICE to allocate a device for the current job if the requested device is restricted. HWCTL to seize a device or reallocate a device to a job in another account.

Data Passed

```
                                 FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    ASSFQ (= 24)     |/////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |/////////////////////| =0,assign;≠0,job no.| 10 FQNAM1
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |  DOS or ANS (1 word RAD50) or 0 (magtape) | 14 FQEXT
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |100001 for snagging assign/reassign; else 0| 22 FQMODE
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |     device name (2 ASCII characters)      | 30 FQDEV
          +---------------------+---------------------+
       33 |  ≠0, unit no. real  | device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFUN	The function code ASSFQ (octal value = 24).
FIRQB+FQNAM1	This byte is set to zero to indicate an assign; if nonzero, it is used as the job number to which the device is to be reassigned. The high byte of this word (FIRQB+11) must be set to zero. If you do not have HWCTL privilege, you can reassign a device only to a job that is logged in to the same account as your current account.

FIRQB+FQEXT	When the device is magnetic tape, this word can contain either DOS or ANS in RAD50 format, to indicate DOS or ANSI label format for the tape drive. It can also be set to zero to indicate the system default for the drive.
FIRQB+FQMODE	This word is set to 100001 to assign a device that is currently assigned to another user. This is a snagging assign and is available only to users with HWCTL privilege. If you do not want a snagging assign, this word must be zero.
FIRQB+FQDEV	Device name, as two ASCII characters.
FIRQB+FQDEVN	The device unit number is passed here in binary. A nonzero value in FIRQB+FQDEVN+1 indicates an explicit device unit number. A zero value in FIRQB+FQDEVN+1 indicates no unit number.

Data Returned

Other than a possible error code in byte 0 of the FIRQB, the ASSFQ subfunction returns the previous owner of the device in FIRQB+FQFIL.

Errors

For Assign (byte at FIRQB+FQNAM1 = 0):

NODEVC — The device name specified at FIRQB+FQDEV is not a valid device name.

NOTAVL — The device and unit specified exists on the system, but the attempt to reserve it is prohibited because:

- o The device is currently reserved by another job.
- o The device or its controller has been disabled by the system manager.
- o The device is a keyboard line for a pseudo keyboard only.

PRVIOL — The device requires DEVICE privilege to allocate.

For Reassign (byte at FIRQB+FQNAM1 ≠ 0):

BDNERR	The job number specified does not exist.
INUSE	The device specified is currently open or has an open file.
NODEVC	The device name is a logical device name for which a physical device is not currently assigned.
NOTAVL	(See previous description for Assign.)
PRVIOL	You do not have HWCTL privilege and you tried to reallocate a device to a job that is logged in to an account other than your current account.

Example

The following code reassigns magnetic tape unit 0 (MT0:) to job 12:

```
MAGT:   .ASCII  /MT/
        CALL    CLRFQB                  ;CALL ROUTINE TO CLEAR FIRQB
        MOVB    #ASSFQ,FIRQB+FQFUN      ;SET FUNCTION CODE
        MOVB    #12.,FIRQB+FQNAM1       ;ASSIGN TO JOB 12
        MOV     #^RANS,FIRQB+14         ;ANSI-LABEL TAPE
        MOV     MAGT,FIRQB+FQDEV        ;MAGTAPE DEVICE
        CLRB    FIRQB+FQDEVN            ;UNIT NO. 0
        MOVB    #377,FIRQB+FQDEVN+1     ;UNIT NO. REAL
        CALFIP
        TSTB    FIRQB                   ;ANY ERRORS?
        BNE     ERRTN                   ;BRANCH TO PROCESS ERROR
```

See Programming Hints for information on the CLRFQB routine.

CLSFQ (Close a Channel)

Form/Example

```
CHANO=8.                                ; Set value for channel
MOVB    #CLSFQ,FIRQB+FQFUN              ; Set function code in FIRQB
MOVB    #CHANO*2,FIRQB+FQFIL            ; Set channel 8 for CLOSE
CALFIP                                  ; Execute monitor directive
```

Function

The CLSFQ function closes a channel. The specific action taken depends on the device or file that was previously opened on the channel and whether it was opened for input or output, as well as the mode with which it was opened. For example, closing a channel on which a magnetic tape was opened for input in file-structured mode causes the monitor to position the tape at the end-of-file (EOF). See the *RSTS/E Programming Manual* for a description of the actions taken on closing various devices/files.

Requesting CLSFQ for a channel that is not currently open returns with no action taken and no error is indicated.

Privileges Required

None

Data Passed

```
                                FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
          +--------------------+--------------------+
        1 |////////////////////////////////////////|  0
          +--------------------+--------------------+
FQFUN   3 |     CLSFQ (= 0)    |////////////////////|  2
          +--------------------+--------------------+
        5 |////////////////////|  channel no. * 2   |  4 FQFIL
          +--------------------+--------------------+
        7 |////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |////////////////////////////////////////| 14
          +--------------------+--------------------+
       17 |////////////////////////////////////////| 16
          +--------------------+--------------------+
       21 |////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQFUN	The function code CLSFQ (octal value = 0).
FIRQB+FQFIL	Channel number times two; defines the channel to be closed.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the CLSFQ function of CALFIP does not return any meaningful data.

Errors

All errors with the CLSFQ function of CALFIP are device- dependent. See Appendix A for a full list of errors.

Example

The following MACRO code closes the file or device on channel 12:

```
  CALL     CLRFQB                          ;CLEAR FIRQB
  MOVB     #CLSFQ,FIRQB+FQFUN              ;SET FUNCTION CODE IN FIRQB
  MOVB     #12.*2,FIRQB+FQFIL              ;SET CHANNEL 12 FOR CLOSE
  CALFIP                                   ;EXECUTE MONITOR DIRECTIVE
  TSTB     FIRQB                           ;TEST BYTE 0 FOR ERROR
  BNE      ERRTN                           ;BRANCH TO PROCESS ERROR
```

See Programming Hints for information on the CLRFQB routine.

CRBFQ (Create a Binary [Executable] File and Open It on a Channel)

Form

```
  MOVB #CRBFQ,FIRQB+FQFUN    ;SET FUNCTION CODE
          .
          .
          .
  (Set parameters in FIRQB appropriate to device)
          .
          .
          .
  CALFIP
```

Function

The CRBFQ function creates and opens a binary (executable) file. It is identical to the CREFQ function, except that the protection code is automatically set to indicate an executable file, and the file must be opened on a disk device.

Privileges Required

TUNE to set caching mode. SYSIO to set the privileged-program bit (bit 7 in FQPROT). A matching PPN, GWRITE, or WWRITE, and/or SYSIO to create or rename a file. You also need write access (by protection code, GWRITE, WWRITE, and/or SYSIO) to supersede an existing file.

Data Passed

```
                              FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    CRBFQ (= 34)     |/////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 |     (must = 0)      |   channel no. * 2   |  4 FQFIL
          +---------------------+---------------------+
        7 |   project number    |  programmer number  |  6 FQPPN
          +---------------------+---------------------+
       11 |         file name in RAD50 format         | 10 FQNAM1
          +                                           +
       13 |                 (2 words)                 | 12
          +---------------------+---------------------+
       15 |     file type in RAD50 format (1 word)    | 14 FQEXT
          +---------------------+---------------------+
       17 |      (file size in 512-byte blocks)       | 16 FQSIZ
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |                   mode                    | 22 FQMODE
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
FQPROT 27 |   file protection   | ≠0, prot.code.real  | 26 FQPFLG
          +---------------------+---------------------+
       31 |     device name (2 ASCII characters)      | 30 FQDEV
          +---------------------+---------------------+
       33 | ≠0, unit no. real   | device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |             file clustersize              | 34 FQCLUS
          +---------------------+---------------------+
       37 |   device cluster number for first block   | 36 FQNENT
          +---------------------+---------------------+
```

FIRQB+FQFUN	The function code CRBFQ (octal value = 34).
FIRQB+FQFIL	Channel number times two; defines the channel upon which the file is to be opened.
FIRQB+FQSIZM	On other types of create opens, this byte contains the most significant bits (MSB) of the file size. Executable (binary) files cannot be greater than 65,535 blocks so this byte must always be passed as zero.
FIRQB+FQPPN	The PPN with which the file is to be created. The project number is in the high byte (FQPPN+1) and the programmer number in the low byte (FQPPN). A value of

	zero in both bytes defaults to the PPN under which the calling program is running.
FIRQB+FQNAM1	The file name created, as two words of RAD50 data.
FIRQB+FQEXT	The file type, as one word of RAD50 data.
FIRQB+FQSIZ	The desired file size, in 512-byte blocks. The file is preextended to the specified size; that is, the space for the file is allocated when the file is opened, rather than as it is written.
FIRQB+FQMODE	The mode with which the file is to be opened; values and actions taken are as described for the MODE modifier in file-structured OPEN FOR OUTPUT statements for disk, as the *RSTS/E Programming Manual* describes. If a mode value is used, bit 15 of this word must be set to 1.
FIRQB+FQPROT	File protection code; values for this field define read/write and execute access to the created file (see the *RSTS/E System User's Guide*). If you want a default protection code, then set a full word of zeros at FIRQB+FQPFLG. In this case, either the system default for protection code will be used, or, if the CRBFQ will be deleting a previously existing file with the same file name, type, PPN, and device, the file protection code of the previously existing file will be used. To assign a specific file protection code, a nonzero value is passed in byte FIRQB+FQPFLG (by convention, 255) and the specific file protection code in byte FIRQB+FQPROT. Bit 6 is automatically set, indicating that the file is executable. The *RSTS/E System User's Guide* describes the protection codes for executable files.
FIRQB+FQDEV	The device name is passed here as two ASCII characters; it must be a disk device. If this word is zero, the public disk structure is assumed.
FIRQB+FQDEVN	The device unit number is passed here in binary. A nonzero value in FQDEVN+1 indicates an explicit device unit number. A zero value in FQDEVN+1 indicates no unit number.
FIRQB+FQCLUS	This parameter has the same function as the CLUSTERSIZE option in BASIC-PLUS. The *BASIC-PLUS Language Manual* describes the CLUSTERSIZE option for disks.
FIRQB+FQNENT	Device cluster number for placement of block 1 of the file. When you are creating a new file, you can place

block 1 of the file on a particular block by specifying the disk device cluster number in this word. If this word is zero, no placement is done. If it is nonzero, the monitor will try to place the file at the specified device cluster or as near after it as possible. If the first block of the file can be placed at or after the specified device cluster number, the monitor sets a bit in the file's entry in the User File Directory (UFD). If the first block of the file cannot be placed at or after the specified device cluster number, the file is placed at the lowest free block on the disk, the bit in the file's entry in the UFD is not set, and no error is returned.

A value of -1 specifies the center of the disk; a value of -2 means immediately after the directory.

Data Returned

```
                                FIRQB
Mne-   Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |/////////////////////| current job no. * 2 |  2 FQJOB
          +---------------------+---------------------+
FQSIZM  5 |     (always 0)      | channel number * 2  |  4 FQFIL
          +---------------------+---------------------+
        7 |   project number    | programmer number   |  6 FQPPN
          +---------------------+---------------------+
       11 |        file name in RAD50 format          | 10 FQNAM1
          +                                           +
       13 |                (2 words)                  | 12
          +---------------------+---------------------+
       15 |    file type in RAD50 format (1 word)     | 14 FQEXT
          +---------------------+---------------------+
       17 |     (file size in 512-byte blocks)        | 16 FQSIZ
          +---------------------+---------------------+
       21 |    reasonable buffer size for device      | 20 FQBUFL
          +---------------------+---------------------+
       23 |               (as passed)                 | 22 FQMODE
          +---------------------+---------------------+
       25 |            device description             | 24 FQFLAG
          +---------------------+---------------------+
FQPROT 27 |   protection code   | clustersize, mod256 | 26 FQPFLG
          +---------------------+---------------------+
       31 |    device name (2 ASCII characters)       | 30 FQDEV
          +---------------------+---------------------+
       33 |      flag byte      | device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |        file identification index          | 34 FQCLUS
          +---------------------+---------------------+
       37 |               (as passed)                 | 36 FQNENT
          +---------------------+---------------------+
```

FIRQB+FQJOB	Current job number times two.
FIRQB+FQFIL	Channel number times two; defines the channel on which the file is open.
FIRQB+FQPPN	The PPN under which the file is open. An actual PPN is returned here even if this word was passed as zero.
FIRQB+FQNAM1	The file name created, as two words of RAD50 data.
FIRQB+FQEXT	The file type created, as one word of RAD50 data.

FIRQB+FQSIZ	The size to which the file was preextended, in 512-byte blocks.
FIRQB+FQBUFL	Reasonable buffer size for disk reads and writes, in bytes. (Always 512 for disk.)
FIRQB+FQFLAG	Description of the device just opened (the same information as the BASIC-PLUS STATUS variable). The low byte contains the device's handler index, always zero (DSKHND) for disk. The high byte contains a set of status flags, irrelevant here since the device is always disk. (See the OPNFQ subfunction for more information on these settings.)
FIRQB+FQPFLG	The file cluster size, modulo 256. That is, a file cluster size of 256 is indicated by zero. This is the same as the value passed at FIRQB+FQCLUS, except that it is returned in a byte instead of a word.
FIRQB+FQPROT	The protection code of the file. Bit 6 is 1, and bits 5 through 0 are as passed. Bit 7 is as passed if the caller has SYSIO privilege; otherwise it is 0.
FIRQB+FQDEV	The device name of the disk device, as two ASCII characters. The actual device name is returned here, even if this word was passed as zero.
FIRQB+FQDEVN	The device unit number. The actual unit number is returned here, even if FIRQB+FQDEVN+1 was passed as zero.
FIRQB+FQCLUS	The file identification index of this file. This word is significant mainly in that it can be used in place of the file name in subsequent opens of the file on disk. You can open the file with the OPNFQ subfunction of CALFIP using an explicit PPN in FIRQB+FQPPN, a zero word in FIRQB+FQNAM1, an explicit device name in FQDEV and FQDEVN, and the file identification index in FIRQB+FQNAM1+2. Note that there is no performance gain for using the file identification index instead of the file name. The file identification index is provided for compatibility with RSX. Furthermore, the file identification index is changed when the REORDR utility is run (see the *RSTS/E System Manager's Guide*).

Errors

NOTCLS	The specified channel is already open. It must be closed before it can be opened again.
PRVIOL	The specified device is not a disk device. The CRBFQ function can be executed only for a disk device.
QUOTA	Extending the file causes the disk quota to be exceeded. This error does not occur if the user has EXQTA privilege.
xxxxx	Other errors are device-dependent. See Appendix A for a full list of possible error codes.

Example

The following MACRO code sets up the FIRQB for the CRBFQ function of CALFIP. The PPN is set to 2,210; the file name and type are set to FILNAM.TYP. The protection code is set such that the file is read/write-protected against everyone but the caller (user with PPN 2,210), and execute-protected against all but the caller and those in the caller's project (users with project number = 2). The file is opened on disk unit 2 (DK2:). File size and cluster size are not specified. The cluster size defaults to the pack cluster size and the file size is not preallocated.

```
DK:    .ASCII   /DK/
         CALL     CLRFQB                 ;CLEAR FIRQB
         MOVB     #CRBFQ,FIRQB+FQFUN     ;SET FUNCTION CODE
         MOVB     #4*2,FIRQB+FQFIL       ;SET CHANNEL = 4
         MOVB     #2,FIRQB+FQPPN+1       ;SET PROJECT NUMBER =2
         MOVB     #210.,FIRQB+FQPPN      ;SET PROG. N0.=210.
         MOV      #^RFIL,FIRQB+FQNAM1    ;SET FILE NAME AND
         MOV      #^RNAM,FIRQB+FQNAM1+2  ;TYPE TO
         MOV      #^RTYP,FIRQB+FQEXT     ;"FILNAM.TYP"
         MOVB     #<8.+16.+32.>,FIRQB+FQPROT ;SET PROTECTION CODE
         MOVB     #255.,FIRQB+FQPFLG     ;SET PROTECTION CODE REAL
         MOV      DK,FIRQB+FQDEV         ;SET DEVICE TO DISK,
         MOVB     #2,FIRQB+FQDEVN        ;UNIT 2
         MOVB     #255.,FIRQB+FQDEVN+1   ;(EXPLICIT DEVICE NO.)
         CALFIP
```

See Programming Hints for information on the CLRFQB routine.

CREFQ (Create a File and Open It on a Channel)

Form

```
  MOVB    #CREFQ,FIRQB+FQFUN   ;SET FUNCTION CODE
          .
          .
          .
  (set parameters appropriate to device)
          .
          .
          .
  CALFIP
```

Function

The CREFQ function performs the same action as a file-structured OPEN FOR OUTPUT statement in BASIC-PLUS. Parameters defining the device, file name and type, protection code, mode, file size, and cluster size can be used by setting values in the FIRQB. The choice depends upon the device.

For example, CREFQ with a file name and type on a magtape device with mode = 128 causes an OPEN for APPEND operation. A search for an existing file with the specified name and on the specified device is made; the file would have to be the last file on the tape. When found, the tape is positioned after the last record in the file, ready for data to be written and appended. The *RSTS/E Programming Manual* describes the file-structured OPEN for OUTPUT operation for the various devices.

Privileges Required

TUNE to set caching mode. SYSIO to set the privilege bit (bit 7 in FQPROT). A matching PPN, GWRITE, WWRITE, and/or SYSIO to create or rename a file. You also need write access (by protection code, GWRITE, WWRITE, and/or SYSIO) to supersede an existing file.

Data Passed

```
                              FIRQB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
           +---------------------+---------------------+
        1  |///////////////////////////////////////////|  0
           +---------------------+---------------------+
FQFUN   3  |    CRBFQ (= 4)      |/////////////////////|  2
           +---------------------+---------------------+
FQSIZM  5  |  MSB of file size   |  channel no. * 2    |  4 FQFIL
           +---------------------+---------------------+
        7  |  project number     | programmer number   |  6 FQPPN
           +---------------------+---------------------+
       11  |        file name in RAD50 format          | 10 FQNAM1
           +                                           +
       13  |                (2 words)                  | 12
           +---------------------+---------------------+
       15  |   file type in RAD50 format (1 word)      | 14 FQEXT
           +---------------------+---------------------+
       17  |             LSB of file size              | 16 FQSIZ
           +---------------------+---------------------+
       21  |///////////////////////////////////////////| 20
           +---------------------+---------------------+
       23  |                   mode                    | 22 FQMODE
           +---------------------+---------------------+
       25  |///////////////////////////////////////////| 24
           +---------------------+---------------------+
FQPROT 27  |  file protection    | ≠0, prot.code.real  | 26 FQPFLG
           +---------------------+---------------------+
       31  |   device name (2 ASCII characters)        | 30 FQDEV
           +---------------------+---------------------+
       33  | ≠0, unit no. real   | device unit number  | 32 FQDEVN
           +---------------------+---------------------+
       35  |             file clustersize              | 34 FQCLUS
           +---------------------+---------------------+
       37  |  device cluster number for first block    | 36 FQNENT
           +---------------------+---------------------+
```

FIRQB+FQFUN	The function code CREFQ (octal value = 4).
FIRQB+FQFIL	Channel number times two; defines the channel upon which the file is to be opened.
FIRQB+FQSIZM	For large disk files (greater than 65,535 blocks), this byte contains the most significant bits (MSB) of the file size. See FIRQB+FQSIZ, for a discussion of the entire 24-bit field used for large files on disk.

FIRQB+FQPPN	The PPN with which the file is to be created. The project number is in the high byte (FIRQB+FQPPN+1), and the programmer number in the low byte (FIRQB+FQPPN). A value of zero defaults to the PPN under which the calling program is running.
FIRQB+FQNAM1	The file name to create, as two words of RAD50 data.
FIRQB+FQEXT	The file type, as one word of RAD50 data.
FIRQB+FQSIZ	The desired file size, in 512-byte blocks. This parameter is relevant only for disk and ANSI magtape files. For disk files, this word forms the least significant bits (LSB) of the file size. It is combined with the byte at FIRQB+FQSIZM to form a 24-bit integer. The disk file is automatically preextended to the indicated size. (That is, the space for the file is allocated when the file is opened, not as it is written.) If preextending is not desired then this value and FIRQB+FQSIZM should be zero. For ANSI magtape, this word has the same function as the FILESIZE option in BASIC-PLUS (see the *RSTS/E Programming Manual*).
FIRQB+FQMODE	The mode with which the file is to be opened; values and actions taken for specific devices are as described for the MODE modifier for file-structured OPEN for OUTPUT statements in the *RSTS/E Programming Manual*. If a mode value is used, bit 15 of this word must be set to 1.
FIRQB+FQPROT	File protection code; values for this field define subsequent read and write access to the opened file (see the *RSTS/E System User's Guide*). If you want a default protection code, then set a full word of zeros at FIRQB+FQPFLG. In this case, either the system default for protection code will be used, or, if the CREFQ will be deleting a previously existing file with the same file name, type, ppn, and device, the file protection code of the previously existing file will be used. To assign a specific file protection code, put a nonzero value in byte FIRQB+FQPFLG (by convention, 255) and the specific file protection code in byte FIRQB+FQPROT. This allows an explicit file protection code of 0.
FIRQB+FQDEV	The device name is passed here as two ASCII characters. A zero word indicates the public disk structure.

FIRQB+FQDEVN	The device unit number is passed here in binary. A nonzero value in FIRQB+FQDEVN+1 indicates an explicit device unit number. A zero value in FIRQB+FQDEVN+1 indicates no unit number.
FIRQB+FQCLUS	This parameter has the same function as the CLUSTERSIZE option in BASIC-PLUS. It is relevant only for disk files and ANSI magtape files. A description of the CLUSTERSIZE option for disk is given in the *BASIC-PLUS Language Manual*; for ANSI magtape, in the *RSTS/E Programming Manual.*
FIRQB+FQNENT	For disk files, the device cluster number for placement of block 1 of the file. When creating a new file, you can place block 1 of the file on a particular block by specifying the disk device cluster number in this word. If this word is zero, no placement is done. If it is nonzero, the monitor will try to place the file at the specified device cluster or as near after it as possible. If the first block of the file can be placed at or after the specified device cluster number, the monitor sets a bit in the file's entry in the UFD. If the first block of the file cannot be placed at or after the specified device cluster number, the file is placed at the lowest free block on the disk, the bit in the file's entry in the UFD is not set, and no error is returned. A value of -1 means the center of the disk; a value of -2 means immediately after the directory.

Data Returned

```
                                 FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
            +---------------------+---------------------+
         1  |///////////////////////////////////////////|  0
            +---------------------+---------------------+
         3  |/////////////////////| current job no. * 2 |  2 FQJOB
            +---------------------+---------------------+
FQSIZM   5  |   MSB of file size  | channel number * 2  |  4 FQFIL
            +---------------------+---------------------+
         7  |   project number    |  programmer number  |  6 FQPPN
            +---------------------+---------------------+
        11  |          file name in RAD50 format        | 10 FQNAM1
            +                                           +
        13  |                 (2 words)                 | 12
            +---------------------+---------------------+
        15  |     file type in RAD50 format (1 word)    | 14 FQEXT
            +---------------------+---------------------+
        17  |              LSB of file size             | 16 FQSIZ
            +---------------------+---------------------+
        21  |     reasonable buffer size for device     | 20 FQBUFL
            +---------------------+---------------------+
        23  |                (as passed)                | 22 FQMODE
            +---------------------+---------------------+
        25  |             device description            | 24 FQFLAG
            +---------------------+---------------------+
FQPROT  27  |   protection code   | clustersize, mod256 | 26 FQPFLG
            +---------------------+---------------------+
        31  |     device name (2 ASCII characters)      | 30 FQDEV
            +---------------------+---------------------+
        33  |      flag byte      | device unit number  | 32 FQDEVN
            +---------------------+---------------------+
        35  |         file identification index         | 34 FQCLUS
            +---------------------+---------------------+
        37  |                (as passed)                | 36 FQNENT
            +---------------------+---------------------+
```

Note

For nondisk devices, the relevant information returned with the CREFQ subfunction is in the two words at FIRQB+FQBUFL and FIRQB+FQFLAG. All other words are simply returned as passed.

FIRQB+FQJOB	Current job number times two.
FIRQB+FQFIL	Channel number times two; defines the channel on which the file is open.

FIRQB+FQSIZM	For large files, this byte contains the MSB of the size to which the file was preextended, in 512-byte blocks. This byte is combined with the word at FIRQB+FQSIZ to form a 24-bit field giving the file size.
FIRQB+FQPPN	The PPN under which the file is open. An actual PPN is returned here even if this word was passed as zero.
FIRQB+FQNAM1	The file name created, as two words of RAD50 data.
FIRQB+FQEXT	The file type created, as one word of RAD50 data.
FIRQB+FQSIZ	The size to which the file was preextended, in 512-byte blocks.
FIRQB+FQBUFL	Reasonable buffer size for this device, in bytes. If you are doing device-independent I/O (that is, if you do not want to keep track of which device is being opened and perform specific opens, reads, and writes, depending on the device), this value is the monitor's best guess for a buffer size to use in subsequent reads and writes on the opened channel. (See the .READ and .WRITE directives.)
FIRQB+FQFLAG	Description of the device just opened (same information as the BASIC-PLUS STATUS variable). The low byte contains the device's handler index. There is one unique handler index for all device types. The high byte contains a set of status flags to allow for device-independent I/O routines.

```
                         DDNFS
                    DDRLO    |
               DDWLO    |    |
          FLGPOS   |    |    |
     FLGMOD   |    |    |    |
FLGFRC   |    |    |    |    |
FLGKB    |   |    |    |    |    |
FLGRND |    |    |    |    |    |    |
      |    |    |    |    |    |    |    |
     15   14   13   12   11   10    9    8| 7                    0
   +-------------------------------------+---------------------+
   |     device-type flags               | device handler index|
   +-------------------------------------+---------------------+
```

High Byte -- Device-Type Flags

The bits in the high byte of the flags word are set to indicate the type of file or device just opened:

FLGRND = 1 The device or file is random-access.
       = 0 The device or file is sequential.

FLGKB	= 1	The file or device is a terminal-type file or device (or is generically a terminal).
	= 0	The file or device is not a terminal-type file or device.
FLGFRC	= 1	The file or device is byte-oriented. That is, the .READ and .WRITE directives handle data in byte units.
	= 0	The file or device is block-oriented. The .READ and .WRITE directives handle data in block units.
FLGMOD	= 1	The file or device accepts modifiers in .READ and .WRITE directives.
	= 0	The file or device does not accept modifiers in .READ and .WRITE directives.
FLGPOS	= 1	The file or device keeps track of its horizontal position and expands characters such as TAB into whatever is appropriate for the file or device. You can determine the current horizontal position with the .POSTN directive.
	= 0	The file or device does not keep track of its horizontal position.
DDWLO	= 1	The file or device has been write-locked (with the mode value in the open) or is generically a read-only device.
	= 0	The file or device is not write-locked.
DDRLO	= 1	The device is generically a write-only device.
	= 0	The file or device is not read-locked.
DDNFS	= 1	The file or device is non-file-structured (or is generically not a file-structured device).
	= 0	The file or device is file-structured.

Low Byte -- Device Handler Index

Bits 0-7 of the flags word contain a handler index that indicates the generic kind of device. The currently defined values follow.

Octal Value	Symbol	Meaning
0	DSKHND	All disks
2	TTYHND	All terminals
4	DTAHND	DECtape
6	LPTHND	All line printers
10	PTRHND	Paper tape reader
12	PTPHND	Paper tape punch
14	CDRHND	Card reader
16	MTAHND	Magnetic tape
20	PKBHND	Pseudo keyboards
22	RXDHND	Flexible diskettes
24	RJEHND	2780 remote job entry
26	NULHND	The null device
30	DMCHND	The DMC11/DMR11 DDCMP interface
36		Reserved
40	KMCHND	KMC11
42	IBMHND	IBM interconnect
46	DMPHND	DMP11/DMV11 device

FIRQB+FQPFLG	The file cluster size, modulo 256. That is, a file cluster size of 256 is indicated by zero. This is the same as the value passed at FIRQB+FQCLUS, except that it is returned in a byte instead of a word.
FIRQB+FQPROT	The protection code of the file. Bit 7 is as passed if the caller has SYSIO privilege, and 0 otherwise; bits 6 to 0 are as passed.
FIRQB+FQDEV	The device name of the disk device, as two ASCII characters. The actual device name is returned here, even if this word was passed as zero.
FIRQB+FQDEVN	The device unit number. The actual unit number is returned here, even if FIRQB+FQDEVN+1 was passed as zero.
FIRQB+FQCLUS	The file identification index of this file. This word is significant mainly in that it can be used in place of the file name in subsequent opens of the file on disk. You can open the file with the OPNFQ subfunction of CALFIP using an explicit PPN in FIRQB+FQPPN, a zero word in FIRQB+FQNAM1, and the file identification index in FIRQB+FQNAM1+2.

Note that there is no performance gain for using the file identification index rather than the file name. The file identification index is provided for compatibility with RSX. Furthermore, the file identification index is changed when the REORDR utility is run (see the *RSTS/E System Manager's Guide*).

Errors

NOTCLS	The specified channel is already open. It must be closed before it can be opened again.
QUOTA	Extending the file causes the disk quota to be exceeded. This error does not occur if the user has EXQTA privilege.
xxxxx	Other errors are device-dependent. See Appendix A for a full list of errors.

Example

The following MACRO code sets up the FIRQB for a CREFQ that opens a file called FILE01.LST on the public disk structure. (The assumption here is that previous code has filled the FIRQB with zeros, so the words at FIRQB+FQDEV and FIRQB+FQDEVN are zero, indicating the public disk. Similarly, the mode (FIRQB+FQMODE) and cluster size (FIRQB+FQCLUS) are zero, indicating normal read/write and the default cluster size.) A protection code of 56 is assigned (write-protected against all but the owner, read-protected against all but those in the owner's project).

```
  MOVB    #CREFQ,FIRQB+FQFUN           ; Set function code in FIRQB
  MOVB    #5*2,FIRQB+FQFIL             ; Set channel = 5
  MOV     #^RFIL,FIRQB+FQNAM1          ; Set file name
  MOV     #^RE01,FIRQB+FQNAM1+2        ;  and type to
  MOV     #^RLST,FIRQB+FQEXT           ;  FILE01.LST
  MOVB    #56.,FIRQB+FQPROT            ; Set protection code to
  MOVB    #255.,FIRQB+FQPFLG           ;  explicit protection code
  CALFIP
```

CRTFQ (Create and Open a Temporary File)

Form

```
  MOVB     #CRTFQ,FIRQB+FQFUN
           .
           .
           .
  (set appropriate parameters)
           .
           .
           .
  CALFIP
```

Function

The CRTFQ function can be used to create and open a temporary file on disk. The file is temporary only in that the monitor generates a file name and type for the file, which is recognized by the LOGOUT utility. LOGOUT destroys such files when the user logs out; the monitor does not inherently destroy temporary files created with CRTFQ.

Most parameters relevant on an open are defaulted. You do not define or refer to the file by name; subsequent read and write operations refer to the channel on which the temporary file is open.

In addition to the function code, you specify a channel and, if desired, a file size and/or file cluster size. If an explicit cluster size is specified, it is used. A specified file size may or may not be used. The monitor attempts to reuse an existing temporary file for the same job by simply reopening that file. In this case, the file's size is the size of the previous file. If no previous temporary file for the job exists, or if one does exist but is already in use by somebody else, then a new file is created, with the specified file size. A new file is also created if an explicit device name or cluster size is given.

Privileges Required

TUNE to set caching mode. You also need write access (by protection code, GWRITE, WWRITE, and/or SYSIO) to supersede an existing file.

Data Passed

```
                                 FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                               Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |     CRTFQ (= 32)    |/////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 |  MSB of file size   |   channel no. * 2   |  4 FQFIL
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |              LSB of file size             | 16 FQSIZ
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |                    mode                   | 22 FQMODE
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |     device name (2 ASCII characters)      | 30 FQDEV
          +---------------------+---------------------+
       33 |  ≠0, unit no. real  | device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |             file cluster size             | 34 FQCLUS
          +---------------------+---------------------+
       37 |  device cluster number for this block     | 36 FQNENT
          +---------------------+---------------------+
```

FIRQB+FQFUN	The function code CRTFQ (octal value = 32).
FIRQB+FQFIL	Channel number times two; defines the channel on which the file is to be opened.
FIRQB+FQSIZM	For large disk files (greater than 65,535 blocks), this byte contains the MSB of the file size. See FIRQB+FQSIZ, for a description of how the entire 24-bit field is used.
FIRQB+FQSIZ	This word contains the LSB of the file's size in 512-byte blocks. (It is combined with the byte at FIRQB+FQSIZM.) The file size may or may not be used.

	If a temporary file already exists that is not in use, the monitor uses the space allocated to the previous temporary file. However, if cluster size is specified, a new file is created, and the file size indicated by the 24-bit file size is used.
FIRQB+FQMODE	The mode with which the file is to be opened. Values are as described for the MODE modifier in OPEN for OUTPUT statements for disk (see the *RSTS/E Programming Manual*). The only relevant modes are for creating a tentative file, creating a contiguous file, creating a conditionally contiguous file, and for data caching. All other mode bits are ignored, except bit 15. If a mode value is used, bit 15 of this word must be set to 1.
FIRQB+FQDEV	The device name is passed here as two ASCII characters; it must be a disk device. A value of zero in this word indicates _SY:, the public disk.
FIRQB+FQDEVN	The device unit number is passed here in binary. A nonzero value in the high byte (FIRQB+FQDEVN+1) indicates an explicit device unit number. A zero value in FIRQB+FQDEVN+1 indicates no unit number.
FIRQB+FQCLUS	File cluster size. Performs the same function as the CLUSTERSIZE option in BASIC-PLUS.
FIRQB+FQNENT	The device cluster number for placement of block 1 of the file on disk. When creating a new file, you can place block 1 of the file on a particular block by specifying the disk device cluster number in this word. If this word is zero, no placement is done. If it is nonzero, the monitor tries to place the file at the specified device cluster or as near after it as possible. If the first block of the file can be placed at or after the specified device cluster number, the monitor sets a bit in the file's entry in the UFD. If the first block of the file cannot be placed at or after the specified device cluster number, the file is placed at the lowest free block on the disk, the bit in the file's entry in the UFD is not set, and no error is returned. A value of -1 means the center of the disk; a value of -2 means immediately after the directory.

Data Returned

```
                                 FIRQB
Mne-   Octal                                              Octal   Mne-
monic  Offset                                             Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////|  0
          +--------------------+--------------------+
        3 |////////////////////| current job no. * 2 |  2 FQJOB
          +--------------------+--------------------+
FQSIZM  5 |  MSB of file size  |  channel no. * 2   |  4 FQFIL
          +--------------------+--------------------+
        7 |  project number    | programmer number  |  6 FQPPN
          +--------------------+--------------------+
       11 |       file name in RAD50 format         | 10 FQNAM1
          +                                         +
       13 |              (2 words)                  | 12
          +--------------------+--------------------+
       15 |   file type in RAD50 format (1 word)    | 14 FQEXT
          +--------------------+--------------------+
       17 |           LSB of file size              | 16 FQSIZ
          +--------------------+--------------------+
       21 |   reasonable buffer size for device     | 20 FQBUFL
          +--------------------+--------------------+
       23 |             (as passed)                 | 22 FQMODE
          +--------------------+--------------------+
       25 |          device description             | 24 FQFLAG
          +--------------------+--------------------+
FQPROT 27 |  protection code   | clustersize, mod256 | 26 FQPFLG
          +--------------------+--------------------+
       31 |   device name (2 ASCII characters)      | 30 FQDEV
          +--------------------+--------------------+
       33 |    flag byte       | device unit number | 32 FQDEVN
          +--------------------+--------------------+
       35 |      file identification index          | 34 FQCLUS
          +--------------------+--------------------+
       37 |             (as passed)                 | 36 FQNENT
          +--------------------+--------------------+
```

FIRQB+FQJOB	Current job number times two.
FIRQB+FQFIL	Channel number times two; defines the channel on which the file is open.
FIRQB+FQSIZM	For large disk files (greater than 65,535 blocks), this byte contains the MSB of the file size in 512-byte blocks. It is combined with the word at FIRQB+FQSIZ to form a 24-bit field giving the file size.
FIRQB+FQPPN	The PPN under which the file is open. An actual PPN is returned here even if this word was passed as zero.

FIRQB+FQNAM1	The file name created; two words of RAD50 data.
FIRQB+FQEXT	The file type created; one word of RAD50 data.
FIRQB+FQSIZ	The size to which the file was preextended, in 512-byte blocks.
FIRQB+FQBUFL	Reasonable buffer size for disk reads and writes, in bytes. (Always 512. for disk.)
FIRQB+FQFLAG	Description of the device just opened (the same information as the BASIC-PLUS STATUS variable). The low byte contains the device's handler index, always zero (DSKHND) for disk. The high byte contains a set of status flags, irrelevant here since the device is always disk. See the OPNFQ subfunction for more information about these settings.
FIRQB+FQPFLG	The file cluster size, modulo 256. That is, a file cluster size of 256 is indicated by zero. This is the same as the value passed at FIRQB+FQCLUS, except that it is returned in a byte instead of a word.
FIRQB+FQPROT	The protection code of the file.
FIRQB+FQDEV	The device name of the disk device, as two ASCII characters. The actual device name is returned here, even if this word was passed as zero.
FIRQB+FQDEVN	The device unit number. The actual unit number is returned here, even if FIRQB+FQDEVN+1 was passed as zero.
FIRQB+FQCLUS	The file identification index of this file. This word is significant mainly in that it can be used in place of the file name in subsequent opens of the file on disk. You can open the file with the OPNFQ subfunction of CALFIP using an explicit PPN in FIRQB+FQPPN, a zero word in FIRQB+FQNAM1, and the file identification index in FIRQB+FQNAM1+2. Note that there is no performance gain for using the file identification index rather than the file name. The file identification index is provided for compatibility with RSX. Furthermore, the file identification index is changed when the REORDR utility is run on the directory (see the *RSTS/E System Manager's Guide*).

Errors

QUOTA	Extending the file causes the disk quota to be exceeded. This error does not occur if the user has EXQTA privilege.
xxxxx	All other errors for this directive are device-dependent. See Appendix A for a full list of errors.

Example

The following MACRO code opens a temporary file on channel 13. Assume that the FIRQB has been initialized to all zeros. Hence, the temporary file is created on the public disk structure:

```
  MOVB    #CRTFQ,FIRQB+FQFUN            ;SET FUNCTION CODE
  MOVB    #13.*2,FIRQB+FQFIL            ;SET CHANNEL TO 13
  CALFIP
```

DALFQ (Deallocate All Devices)

Form

```
  MOVB    #DALFQ,FIRQB+FQFUN
  CALFIP
```

Function

The DALFQ subfunction deallocates all devices currently allocated to the job.

Privileges Required

None

Data Passed

```
                                 FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
           +---------------------+---------------------+
         1 |///////////////////////////////////////////|  0
           +---------------------+---------------------+
FQFUN    3 |     DALFQ (= 30)    |/////////////////////|  2
           +---------------------+---------------------+
         5 |///////////////////////////////////////////|  4
           +---------------------+---------------------+
         7 |///////////////////////////////////////////|  6
           +---------------------+---------------------+
        11 |///////////////////////////////////////////| 10
           +---------------------+---------------------+
        13 |///////////////////////////////////////////| 12
           +---------------------+---------------------+
        15 |///////////////////////////////////////////| 14
           +---------------------+---------------------+
        17 |///////////////////////////////////////////| 16
           +---------------------+---------------------+
        21 |///////////////////////////////////////////| 20
           +---------------------+---------------------+
        23 |///////////////////////////////////////////| 22
           +---------------------+---------------------+
        25 |///////////////////////////////////////////| 24
           +---------------------+---------------------+
        27 |///////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |///////////////////////////////////////////| 30
           +---------------------+---------------------+
        33 |///////////////////////////////////////////| 32
           +---------------------+---------------------+
        35 |///////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |///////////////////////////////////////////| 36
           +---------------------+---------------------+
```

FIRQB+FQFUN   The function code DALFQ (octal value = 30).

Data Returned

The DALFQ subfunction of CALFIP does not return any meaningful data.

Errors

No errors are possible with DALFQ.

DEAFQ (Deallocate a Device)

Form

```
  MOVB     #DEAFQ,FIRQB+FQFUN
               .
               .
               .
  (Define device to be deallocated in FIRQB)
               .
               .
               .
  CALFIP
```

Function

The DEAFQ subfunction deallocates (that is, releases it for use by other jobs) a device from the current job.

Privileges Required

None

Data Passed

```
                              FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////|  0
          +--------------------+--------------------+
FQFUN   3 |     DEAFQ (= 26)   |////////////////////|  2
          +--------------------+--------------------+
        5 |/////////////////////////////////////////|  4
          +--------------------+--------------------+
        7 |/////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |                (must = 0)               | 10 FQNAM1
          +--------------------+--------------------+
       13 |/////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |/////////////////////////////////////////| 14
          +--------------------+--------------------+
       17 |/////////////////////////////////////////| 16
          +--------------------+--------------------+
       21 |/////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |/////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |/////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |/////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |     device name (2 ASCII characters)    | 30 FQDEV
          +--------------------+--------------------+
       33 |  ≠0, unit no. real | device unit number | 32 FQDEVN
          +--------------------+--------------------+
       35 |/////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |/////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQFUN	The DEAFQ function code (octal value = 26).
FIRQB+FQNAM1	The word at this location must be set to zero.
FIRQB+FQDEV	The name of the device to be deallocated, as two ASCII characters.
FIRQB+FQDEVN	The device unit number is passed here in binary. A nonzero value in FIRQB+FQDEVN+1 indicates an explicit device unit number, while a zero value indicates no device unit number.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the DEAFQ subfunction of CALFIP does not return any meaningful data.

Errors

NODEVC	The device or its type specified at FIRQB+FQDEV and FIRQB+FQDEVN is not part of your system configuration.

Example

The following code deallocates LP: from the current job:

```
  CALL    CLRFQB                       ; Clear FIRQB
  MOVB    #DEAFQ,FIRQB+FQFUN           ; Set function code
  MOV     #"LP,FIRQB+FQDEV             ; Device=line printer
  CALFIP
```

See Programming Hints for information on the CLRFQB routine.

DIRFQ (Get Directory Information)

Form

```
  MOVB     #DIRFQ,FIRQB+FQFUN
           .
           .
           .
  (Define device for the directory wanted)
           .
           .
           .
  CALFIP
```

Function

The DIRFQ subfunction of CALFIP returns directory information about a disk, DECtape, or magtape file. Two forms of the call are available. One leaves a magtape file positioned at the EOF and returns the size of the file as part of the directory information. The second form, for magtape only, leaves a magtape file positioned at the beginning of the file, and does not return the size of the file.

Note

For disk directory lookup on a PPN other than that of the caller, DIRFQ returns only those files to which the caller has read or execute access.

Privileges Required

DEVICE to perform a directory lookup or disk lookup on caller's account if the referenced device is restricted. You need read or execute access (by protection code, GREAD, or WREAD) to perform a disk directory lookup on another account.

Data Passed -- Directory Lookup on Index

```
                                FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
             +------------------+------------------+
           1 |//////////////////////////////////////|  0
             +------------------+------------------+
FQFUN      3 |    DIRFQ (= 12)  |///////////////////|  2
             +------------------+------------------+
           5 |        index of file to read         |  4 FQFIL
             +------------------+------------------+
           7 |  project number  | programmer number |  6 FQPPN
             +------------------+------------------+
          11 |//////////////////////////////////////| 10
             +------------------+------------------+
          13 |//////////////////////////////////////| 12
             +------------------+------------------+
          15 |//////////////////////////////////////| 14
             +------------------+------------------+
          17 |//////////////////////////////////////| 16
             +------------------+------------------+
          21 |//////////////////////////////////////| 20
             +------------------+------------------+
          23 |//////////////////////////////////////| 22
             +------------------+------------------+
          25 |//////////////////////////////////////| 24
             +------------------+------------------+
          27 |//////////////////////////////////////| 26
             +------------------+------------------+
          31 |   device name (2 ASCII characters)   | 30 FQDEV
             +------------------+------------------+
          33 | ≠0, unit no. real | device unit number | 32 FQDEVN
             +------------------+------------------+
          35 |//////////////////////////////////////| 34
             +------------------+------------------+
          37 |//////////////////////////////////////| 36
             +------------------+------------------+
```

FIRQB+FQFUN	The function code DIRFQ (octal value = 12).
FIRQB+FQFIL	The index of the file to read. If this word is zero, the monitor returns data for the first file in the directory. For some positive value n, the monitor returns data for the n+1 file in the directory. For magtape, a value of zero causes the monitor to rewind the tape before it gets the information for the first file (by reading the label record of the file). The monitor then spaces the tape forward to the next EOF record and calculates the number of records in the file. The tape is left in that position.

	A nonzero value performs the same action, except that the tape is not rewound.
	For DECtape, the first call issued must have a value of zero in this word to read the directory blocks from the tape before reading the first file. Subsequent calls with this word nonzero read the directory from the BUFF.SYS file. (Directory information for DECtape is kept in this system file on disk to speed up DECtape file processing: see the *RSTS/E Programming Manual*.)
FIRQB+FQPPN	The PPN of the directory to look up, for disk or magnetic tape. (The monitor does not use these bytes if the device is DECtape but simply returns information for each file read on the device.)
	If this word is zero and the device is disk, this directive returns information for the PPN under which this job is being executed.
	If this word is zero and the device is magnetic tape, this directive returns information for each file read, regardless of the PPN under which it was written.
FIRQB+FQDEV	The device name, as two ASCII characters. Must be disk, magnetic tape, or DECtape. If this word is zero, the public disk structure (_SY:) is used.
FIRQB+FQDEVN	The device unit number is passed here in binary. A nonzero value in FQDEVN+1 indicates an explicit device unit number. A zero value in FQDEVN+1 indicates no unit number.

Data Passed - Special Magnetic Tape Lookup

```
                                 FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
                +---------------------+---------------------+
          1     |///////////////////////////////////////////|  0
                +---------------------+---------------------+
FQFUN     3     |     DIRFQ (= 12)    |/////////////////////|  2
                +---------------------+---------------------+
          5     |             index of file to read         |  4 FQFIL
                +---------------------+---------------------+
          7     |  must = 177777 for magnetic tape lookup   |  6 FQPPN
                +---------------------+---------------------+
         11     |///////////////////////////////////////////| 10
                +---------------------+---------------------+
         13     |///////////////////////////////////////////| 12
                +---------------------+---------------------+
         15     |///////////////////////////////////////////| 14
                +---------------------+---------------------+
         17     |///////////////////////////////////////////| 16
                +---------------------+---------------------+
         21     |///////////////////////////////////////////| 20
                +---------------------+---------------------+
         23     |///////////////////////////////////////////| 22
                +---------------------+---------------------+
         25     |///////////////////////////////////////////| 24
                +---------------------+---------------------+
         27     |///////////////////////////////////////////| 26
                +---------------------+---------------------+
         31     |     device name (2 ASCII characters)      | 30 FQDEV
                +---------------------+---------------------+
         33     |  ≠0, unit no. real  | device unit number  | 32 FQDEVN
                +---------------------+---------------------+
         35     |///////////////////////////////////////////| 34
                +---------------------+---------------------+
         37     |///////////////////////////////////////////| 36
                +---------------------+---------------------+
```

FIRQB+FQFUN	The function code DIRFQ (octal value = 12).
FIRQB+FQFIL	Index number of the file to be read. If this word is zero, information is returned for the first file in the directory. If this word is some positive value n, information is returned for the n+1 file in the directory. A value of zero causes the monitor to rewind the tape before getting information from the first file (by reading the label record). It then back spaces the tape one record, leaving it positioned at the beginning-of-file (BOF). (This action leaves the tape positioned such that an open on this file will succeed on a single read from tape.)

A nonzero value does not cause the tape to be rewound; the next record is read (it must be a label), and the tape is backspaced one record. When you are searching a tape for specific files to read, the normal action is to execute this directive with a value of zero first. If the file is one to be read, open the file requesting no rewind, process the file, and close it to position the tape at the EOF. If the file is not one to be read, space the tape forward to the next EOF. (You can do this in MACRO with the .SPEC directive.) Then issue the DIRFQ call with a nonzero value in the word beginning at FIRQB+FQFIL, and continue the process.

FIRQB+FQPPN	This word must be set to 177777 for the special magnetic tape lookup operation.
FIRQB+FQDEV	The device name as two ASCII characters. The device must be a magnetic tape drive.
FIRQB+FQDEVN	The device unit number, in binary. A nonzero value in FIRQB+FQDEVN+1 indicates an explicit device unit number. A zero value in FIRQB+FQDEVN+1 indicates no unit number.

Data Returned (Both)

```
                              FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
          +----------------------+----------------------+
        1 |//////////////////////////////////////////////|  0
          +----------------------+----------------------+
        3 |//////////////////////| current job no. * 2 |  2 FQJOB
          +----------------------+----------------------+
        5 |                 (as passed)                 |  4 FQFIL
          +----------------------+----------------------+
        7 |   project number     | programmer number   |  6 FQPPN
          +----------------------+----------------------+
       11 |          file name in RAD50 format          | 10 FQNAM1
          +                                             +
       13 |                  (2 words)                  | 12
          +----------------------+----------------------+
       15 |     file type in RAD50 format (1 word)      | 14 FQEXT
          +----------------------+----------------------+
       17 |        LSB of file length in blocks         | 16 FQSIZ
          +----------------------+----------------------+
       21 | MSB of file length |   protection code     | 20 FQNAM2
          +----------------------+----------------------+
       23 |            date of last access              | 22 FQMODE
          +----------------------+----------------------+
       25 |              date of creation               | 24 FQFLAG
          +----------------------+----------------------+
       27 |              time of creation               | 26 FQPFLG
          +----------------------+----------------------+
       31 |                                             | 30 FQDEV
          +                 (as passed)                 +
       33 |                                             | 32 FQDEVN
          +----------------------+----------------------+
       35 |        file clustersize (disk only)         | 34 FQCLUS
          +----------------------+----------------------+
       37 |     USTAT byte       | number entries ret. | 36 FQNENT
          +----------------------+----------------------+
```

FIRQB+FQJOB	The current job number times two.
FIRQB+FQPPN	The PPN of the file. The project number is in the high byte (FIRQB+FQPPN+1) and the programmer number is in the low byte (FIRQB+FQPPN).
FIRQB+FQNAM1	The file name; two words in RAD50 format.
FIRQB+FQEXT	The file type; one word in RAD50 format.
FIRQB+FQSIZ	The LSB of the file size, in 512-byte blocks. This word is combined with the byte at FIRQB+21 to form a 24-bit value giving the file's size.

FIRQB+FQNAM2	The protection code of the file.
FIRQB+FQNAM2+1	The MSB of the file's size in 512-byte blocks (see FIRQB+FQSIZ).
FIRQB+FQMODE	The date the file was last accessed, in system internal format: [(year - 1970) * 1000.] + day-within-year See the .DATE directive for a discussion of the system internal format for dates.
FIRQB+FQFLAG	The date the file was created, in system internal format.
FIRQB+FQPFLG	The time that the file was created, in system internal format: minutes before midnight, where midnight = 1440. (See the .DATE directive for a discussion of the system internal format for time.)
FIRQB+FQCLUS	The file cluster size for disk devices. It is not used for tape.
FIRQB+FQNENT	Number of entries returned: 8 for disk; 6 for tape.
FIRQB+FQNENT+1	Internal flag information (disk only):

Bit	Meaning When Set
1	File is placed
2	Some job has write access now
3	File is open in update mode
4	File is contiguous; no extend allowed
5	No delete or rename allowed
7	File is marked for deletion

CALFIP
DLNFQ

Data Passed

```
                              FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
         +-------------------+--------------------+
       1 |////////////////////////////////////////|  0
         +-------------------+--------------------+
FQFUN  3 |     DLNFQ (= 6)   |////////////////////|  2
         +-------------------+--------------------+
       5 |////////////////////////////////////////|  4
         +-------------------+--------------------+
       7 |   project number  | programmer number  |  6 FQPPN
         +-------------------+--------------------+
      11 |         file name in RAD50 format      | 10 FQNAM1
         +                                        +
      13 |                (2 words)               | 12
         +-------------------+--------------------+
      15 |     file type in RAD50 format (1 word) | 14 FQEXT
         +-------------------+--------------------+
      17 |////////////////////////////////////////| 16
         +-------------------+--------------------+
      21 |////////////////////////////////////////| 20
         +-------------------+--------------------+
      23 |////////////////////////////////////////| 22
         +-------------------+--------------------+
      25 |////////////////////////////////////////| 24
         +-------------------+--------------------+
      27 |////////////////////////////////////////| 26
         +-------------------+--------------------+
      31 |     device name (2 ASCII characters)   | 30 FQDEV
         +-------------------+--------------------+
      33 | ≠0, unit no. real | device unit number | 32 FQDEVN
         +-------------------+--------------------+
      35 |////////////////////////////////////////| 34
         +-------------------+--------------------+
      37 |////////////////////////////////////////| 36
         +-------------------+--------------------+
```

FIRQB+FQFUN	The function code DLNFQ (octal value = 6).
FIRQB+FQPPN	The PPN of the file to deleted. The project number is in the high byte (FIRQB+FQPPN+1), and the programmer number is in the low byte (FIRQB+FQPPN). A value of zero defaults to the PPN under which the calling program is running.
FIRQB+FQNAM1	The name of the file to be deleted, as two words of RAD50 data.
FIRQB+FQEXT	The file type, as one word of RAD50 data.

FIRQB+FQDEV	The name of the device containing the file to be deleted, as two ASCII characters; it must be a disk or DECtape device. A value of zero in this word indicates the public disk structure (_SY:).
FIRQB+FQDEVN	The device unit number, in binary. A nonzero value in FQDEVN+1 indicates an explicit device unit number. A zero value in FQDEVN+1 indicates no unit number.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the DLNFQ function of CALFIP does not return any meaningful data.

Errors

NOSUCH	The file specified in the data passed cannot be found.
PRVIOL	Protection violation. An attempt was made to delete a file that is either write-protected against the caller or marked for no delete.

Example

The following code deletes the file MYFIL.LST from the user's account on the public structure. Assume that the FIRQB has been filled with zeros previously:

```
MOVB    #DLNFQ,FIRQB+FQFUN
MOV     #^RMYF,FIRQB+FQNAM1          ;SET FILE NAME
MOV     #^RILE,FIRQB+FQNAM1+2        ;AND TYPE
MOV     #^RLST,FIRQB+FQEXT           ;TO "MYFILE.LST"
CALFIP
```

ERRFQ (Return Error Message Text)

Form/Example

```
  MOVB     #ERRFQ,FIRQB+FQFUN
  MOVB     #ERR,FIRQB+FQERNO
```

Function

The ERRFQ subfunction of CALFIP returns error message text from the system error message file or from the default error message file if an error message file is not currently installed. The text is associated with the value of the error code passed as byte 4 of the FIRQB. This call returns the full RSTS/E error message text associated with errors returned in byte 0 of the FIRQB on all the monitor directives.

Privileges Required

None

Data Passed

```
                                FIRQB
Mne-   Octal                                          Octal   Mne-
monic  Offset                                         Offset  monic
            +--------------------+--------------------+
        1   |/////////////////////////////////////////|  0
            +--------------------+--------------------+
FQFUN   3   |    ERRGF (= 14)    |////////////////////|  2
            +--------------------+--------------------+
        5   |////////////////////|     error code     |  4  FQERNO
            +--------------------+--------------------+
        7   |/////////////////////////////////////////|  6
            +--------------------+--------------------+
       11   |/////////////////////////////////////////| 10
            +--------------------+--------------------+
       13   |/////////////////////////////////////////| 12
            +--------------------+--------------------+
       15   |/////////////////////////////////////////| 14
            +--------------------+--------------------+
       17   |/////////////////////////////////////////| 16
            +--------------------+--------------------+
       21   |/////////////////////////////////////////| 20
            +--------------------+--------------------+
       23   |/////////////////////////////////////////| 22
            +--------------------+--------------------+
       25   |/////////////////////////////////////////| 24
            +--------------------+--------------------+
       27   |/////////////////////////////////////////| 26
            +--------------------+--------------------+
       31   |/////////////////////////////////////////| 30
            +--------------------+--------------------+
       33   |/////////////////////////////////////////| 32
            +--------------------+--------------------+
       35   |/////////////////////////////////////////| 34
            +--------------------+--------------------+
       37   |/////////////////////////////////////////| 36
            +--------------------+--------------------+
```

FIRQB+FQFUN	The function code ERRFQ (octal value = 16).
FIRQB+FQERNO	The error code value (in binary) for which the corresponding error message text is to be returned.

Errors

The ERRFQ subfunction of CALFIP does not return any errors.

Data Returned

```
                              FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 | KB*2(-KB*2+1,detach)|    job number * 2    |  2 FQJOB
          +---------------------+---------------------+
        5 |                                           |  4 FQFIL
          +                                           +
        7 |                                           |  6
          +                                           +
       11 |                                           | 10
          +    error message -- padded with nulls     +
       13 |                                           | 12
          +                                           +
       15 |      to 28 characters (ASCII format)      | 14
          +                                           +
       17 |                                           | 16
          +                                           +
       21 |                                           | 20
          +                                           +
       23 |                                           | 22
          +                                           +
       25 |                                           | 24
          +                                           +
       27 |                                           | 26
          +                                           +
       31 |                                           | 30
          +                                           +
       33 |                                           | 32
          +                                           +
       35 |                                           | 34
          +                                           +
       37 |                                           | 36
          +---------------------+---------------------+
```

FIRQB+FQJOB	The current job number times two.
FIRQB+3	If the job is attached, two times the currently attached keyboard number. If the job is detached, the one's complement of two times the currently detached keyboard number. For example, if the keyboard number is 5, the value in FIRQB+3 is: 12 for an attached job and 365 (377-12) for a detached job.
FIRQB+FQFIL	The error message text begins in this byte. The text is padded with zeros to 28 characters, if necessary.

LOKFQ (Disk File/Wildcard Lookup)

Form

```
  MOVB    #LOKFQ,FIRQB+FQFUN
          .
          .
          .
  (Set up FIRQB to define file/wildcard)
          .
          .
          .
  CALFIP
```

Function

The LOKFQ subfunction of CALFIP does one of two actions:

- o Looks for a file on disk by name and returns directory information (date of creation, and so forth)
- o Performs a wildcard file search

For example, with a file name and type in the FIRQB of *.TXT it searches an account for a file with any file name and a type of .TXT. By incrementing an index and reexecuting LOKFQ, you can search through an entire directory for all such files.

Note

> For disk directory lookup on a PPN other than that of the caller, LOKFQ returns only files to which the caller has read or execute access.

Privileges Required

DEVICE to perform a disk directory lookup on the caller's account if the referenced device is restricted. You need read or execute access (by protection code, GREAD, or WREAD) to perform a disk directory lookup on another account.

Data Passed - Disk Directory Lookup

```
                                 FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
              +----------------------+----------------------+
          1   |/////////////////////////////////////////////|  0
              +----------------------+----------------------+
FQFUN     3   |     LOKFQ (= 22)     |//////////////////////|  2
              +----------------------+----------------------+
          5   |          (must equal 177777 octal)          |  4 FQFIL
              +----------------------+----------------------+
          7   |   project number     |  programmer number   |  6 FQPPN
              +----------------------+----------------------+
         11   |                  file name                  | 10 FQNAM1
              +               in RAD50 format               +
         13   |                  (2 words)                  | 12
              +----------------------+----------------------+
         15   |     file type in RAD50 format (1 word)      | 14 FQEXT
              +----------------------+----------------------+
         17   |/////////////////////////////////////////////| 16
              +----------------------+----------------------+
         21   |/////////////////////////////////////////////| 20
              +----------------------+----------------------+
         23   |/////////////////////////////////////////////| 22
              +----------------------+----------------------+
         25   |/////////////////////////////////////////////| 24
              +----------------------+----------------------+
         27   |/////////////////////////////////////////////| 26
              +----------------------+----------------------+
         31   |  device name (disk) (2 ASCII characters)    | 30 FQDEV
              +----------------------+----------------------+
         33   | ≠0, device no. real  | device unit number   | 32 FQDEVN
              +----------------------+----------------------+
         35   |/////////////////////////////////////////////| 34
              +----------------------+----------------------+
         37   |/////////////////////////////////////////////| 36
              +----------------------+----------------------+
```

FIRQB+FQFUN	The function code LOKFQ (octal value = 22).
FIRQB+FQFIL	The word beginning at this location must be set to 177777 (octal) to indicate the "disk directory lookup by file name" option of LOKFQ.
FIRQB+FQPPN	The PPN for the file to be looked up. A value of zero for this word defaults to the PPN under which the calling program is running.
FIRQB+FQNAM1	The file name to be looked up; two words in RAD50 format.

FIRQB+FQEXT	The file type of the file to be looked up; one word in RAD50 format.
FIRQB+FQDEV	The device name (must be disk), as two ASCII characters. If both bytes are zero, the public disk structure (SY:) is used.
FIRQB+FQDEVN	The disk device unit number is passed here in binary. A nonzero value in FQDEVN+1 indicates an explicit device unit number. A zero value in FQDEVN+1 indicates the system default.

Data Returned - Disk Directory Lookup by File Name

```
                                 FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|   0
          +---------------------+---------------------+
        3 |/////////////////////|  job number * 2     |   2 FQJOB
          +---------------------+---------------------+
        5 |    same as data passed (177777 octal)     |   4 FQFIL
          +---------------------+---------------------+
        7 |   project number    | programmer number   |   6 FQPPN
          +---------------------+---------------------+
       11 |          file name in RAD50 format        |  10 FQNAM1
          +                                           +
       13 |                (2 words)                  |  12
          +---------------------+---------------------+
       15 |    file type in RAD50 format (1 word)     |  14 FQEXT
          +---------------------+---------------------+
       17 |  LSB of file length in 512-byte blocks    |  16 FQSIZ
          +---------------------+---------------------+
       21 |  MSB of file length |   protection code   |  20 FQNAM2
          +---------------------+---------------------+
       23 |            date of last access            |  22 FQMODE
          +---------------------+---------------------+
       25 |             date of creation              |  24 FQFLAG
          +---------------------+---------------------+
       27 |             time of creation              |  26 FQPFLG
          +---------------------+---------------------+
       31 |     device name (2 ASCII characters)      |  30 FQDEV
          +---------------------+---------------------+
       33 | ≠0, unit no. real   | device unit number  |  32 FQDEVN
          +---------------------+---------------------+
       35 |             file clustersize              |  34 FQCLUS
          +---------------------+---------------------+
       37 |         file identification index         |  36 FQNENT
          +---------------------+---------------------+
```

FIRQB+FQJOB	The current job number times two.

FIRQB+FQFIL	The word at this location is the same as the data passed, in this case, 177777 (octal).
FIRQB+FQPPN	The PPN of the file (same as data passed).
FIRQB+FQNAM1	The file name; two words in RAD50 format (same as data passed).
FIRQB+FQEXT	The file type; one word in RAD50 format (same as data passed).
FIRQB+FQSIZ	The LSB of the file's size in 512-byte blocks. This word is combined with the byte at FIRQB+21 to form a 24-bit field giving the file size.
FIRQB+FQNAM2	The file's protection code, in binary, is returned in this byte.
FIRQB+21	The MSB of the file size in 512-byte blocks. It is combined with the word at FIRQB+FQSIZ to form a 24-bit field giving the file size.
FIRQB+FQMODE	The date the file was last accessed, in system internal format: [(year - 1970) * 1000.] + day-within-year. See the .DATE directive for a discussion of the system internal format for dates.
FIRQB+FQFLAG	The date the file was created, in system internal format (see FIRQB+FQMODE).
FIRQB+FQPFLG	The time the file was created, in system internal format: minutes until midnight, with 1440 equal to midnight.
FIRQB+FQDEV	The device name, as two ASCII characters. Always a specific name, even if zero was passed here.
FIRQB+FQDEVN	The device unit number, in binary. A specific number is always returned here; FIRQB+FQDEVN+1 is always nonzero.
FIRQB+FQCLUS	The file cluster size is returned in this word.
FIRQB+FQNENT	The file identification index is returned in this word. You can use the file identification index instead of the file name to open a file on disk with the OPNFQ subfunction of CALFIP. To do so, specify an explicit device name at FIRQB+FQDEV, a device unit number at FIRQB+FQDEVN, an explicit PPN at FIRQB+FQPPN, a zero

word at FIRQB+FQNAM1, and the file identification index at FIRQB+FQNAM1+2. (The file identification index is used by utilities that access software subroutines in the RMS libraries, for example.)

Note that there is no performance gain in using the file identification index instead of the file name. The file identification index is provided for compatibility with RSX. Furthermore, the file identification index is changed when the REORDR utility is run on the directory (see the *RSTS/E System Manager's Guide*).

Errors

BADNAM	The file name in bytes FIRQB+FQNAM1 through FIRQB+13 is missing or invalid.
NOSUCH	The device specified at FIRQB+FQDEV is not disk, or the file specified does not exist on the specified disk. This error also occurs when a user does not have read or execute access to the specified file.

Example

The following code looks for the file MATRIX.DAT on the system disk:

```
CALL    CLRFQB                          ;CLEAR FIRQB
MOV     #LOKFQ,FIRQB+FQFUN              ;SET FUNCTION CODE
MOV     #177777,FIRQB+FQFIL             ;FILENAME LOOKUP
CLR     FIRQB+FQPPN                     ;CALLER'S ACCOUNT
MOV     #^RMAT,FIRQB+FQNAM1             ;SET FILENAME
MOV     #^RRIX,FIRQB+FQNAM1+2           ;AND TYPE
MOV     #^RDAT,FIRQB+FQEXT              ;TO "MATRIX.DAT"
CLR     FIRQB+FQDEV                     ;SYSTEM DISK
CLR     FIRQB+FQDEVN                    ;DEVICE
CALFIP
```

See Programming Hints for information on the CLRFQB routine.

Data Passed -- Disk Wildcard Directory Lookup

```
                                FIRQB
Mne-   Octal                                                Octal   Mne-
monic  Offset                                               Offset  monic
            +---------------------+---------------------+
          1 |///////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN     3 |     LOKFQ (= 22)    |/////////////////////|  2
            +---------------------+---------------------+
          5 | index: n means search for n+1 occurrence  |  4 FQFIL
            +---------------------+---------------------+
          7 |   project number    | programmer number   |  6 FQPPN
            +---------------------+---------------------+
         11 |             wildcard file name            | 10 FQNAM1
            +        specification in RAD50 format      +
         13 |                 (2 words)                 | 12
            +---------------------+---------------------+
         15 |wildcard file type in RAD50 format (1 word)| 14 FQEXT
            +---------------------+---------------------+
         17 |///////////////////////////////////////////| 16
            +---------------------+---------------------+
         21 |///////////////////////////////////////////| 20
            +---------------------+---------------------+
         23 |/////////////////////|  marked-for-delete  | 22 FQMODE
            +---------------------+---------------------+
         25 |///////////////////////////////////////////| 24
            +---------------------+---------------------+
         27 |///////////////////////////////////////////| 26
            +---------------------+---------------------+
         31 |  device name (disk) (2 ASCII characters)  | 30 FQDEV
            +---------------------+---------------------+
         33 | ≠0, device no. real | device unit number  | 32 FQDEVN
            +---------------------+---------------------+
         35 |///////////////////////////////////////////| 34
            +---------------------+---------------------+
         37 |///////////////////////////////////////////| 36
            +---------------------+---------------------+
```

FIRQB+FQFUN	The function code LOKFQ (octal value = 22).
FIRQB+FQFIL	An index number specifying the occurrence of the file name meeting the wildcard specifications. A value of zero in this word causes the monitor to search for the first file name in the directory that meets the specification. A value of one causes the monitor to search for the second file name, and so forth.
FIRQB+FQPPN	The PPN of the account whose directory of disk files is to be searched.

FIRQB+FQNAM1	The wildcard file name; as two words in RAD50 format. Either an asterisk (*) character can replace the entire file name, or a question mark (?) character can replace any character in the file name. For example, a file name of FILE?? would cause the monitor to search the directory for any file name beginning with the characters FILE. An * character indicates that the file name does not matter in the search.
FIRQB+FQEXT	The wildcard file type; as one word in RAD50 format. An * character can replace the entire file type, or a ? character can replace any character in the file type. For example, a file type of BA? causes the monitor to search the directory for any file type beginning with the characters BA. An * character indicates that the file type does not matter in the search.
FIRQB+FQMODE	If bit 14 is set, LOKFQ returns information about marked-for-delete files.
FIRQB+FQDEV	The name of the device to be searched (must be disk). A value of 0 in this word indicates the public disk structure (SY:).
FIRQB+FQDEVN	The device unit number in binary. A nonzero value in FIRQB+FQDEVN+1 indicates an explicit device unit number. A zero value in FIRQB+FQDEVN+1 indicates the system default.

Data Returned -- Disk Wildcard Directory Lookup

```
                               FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////|  0
          +--------------------+--------------------+
        3 |////////////////////|  job number * 2    |  2 FQJOB
          +--------------------+--------------------+
        5 |               (as passed)               |  4 FQFIL
          +--------------------+--------------------+
        7 |  project number    | programmer number  |  6 FQPPN
          +--------------------+--------------------+
       11 |       file name in RAD50 format         | 10 FQNAM1
          +                                         +
       13 |                (2 words)                | 12
          +--------------------+--------------------+
       15 |   file type in RAD50 format (1 word)    | 14 FQEXT
          +--------------------+--------------------+
       17 |  LSB of file length in 512-byte blocks  | 16 FQSIZ
          +--------------------+--------------------+
       21 | MSB of file length |  protection code   | 20 FQNAM2
          +--------------------+--------------------+
       23 |     date of last access (disk only)     | 22 FQMODE
          +--------------------+--------------------+
       25 |            date of creation             | 24 FQFLAG
          +--------------------+--------------------+
       27 |            time of creation             | 26 FQPFLG
          +--------------------+--------------------+
       31 |                                         | 30 FQDEV
          +               (as passed)               +
       33 |                                         | 32 FQDEVN
          +--------------------+--------------------+
       35 |      file clustersize (disk only)       | 34 FQCLUS
          +--------------------+--------------------+
       37 |    USTAT byte      |////////////////////| 36 FQNENT
          +--------------------+--------------------+
```

FIRQB+FQJOB	The current job number times two.
FIRQB+FQPPN	The PPN of the file (same as data passed).
FIRQB+FQNAM1	The actual file name of a file meeting the wildcard specification in the data passed; two words in RAD50 format.
FIRQB+FQEXT	The actual file type of a file meeting the wildcard specification in the data passed; one word in RAD50 format.

FIRQB+FQSIZ	The LSB of the file's size in 512-byte blocks. This word is combined with the byte at FIRQB+21 to form a 24-bit field giving the file's size.
FIRQB+FQNAM2	The file's protection code, in binary, is returned in this byte.
FIRQB+21	The MSB of the file's size in 512-byte blocks. The byte is combined with the word at FIRQB+FQSIZ to give the file's size.
FIRQB+FQMODE	The date the file was last accessed, in system internal format: [(year - 1970) * 1000.] + day-within-year See the .DATE directive for a discussion of the system internal format for dates.
FIRQB+FQFLAG	The date the file was created, in system internal format (see FIRQB+FQMODE).
FIRQB+FQPFLG	The time the file was created, in system internal format: minutes until midnight, with 1440 equal to midnight.
FIRQB+FQDEV	The device name, as two ASCII characters. Always a specific name, even if zero was passed here.
FIRQB+FQDEVN	The device unit number, in binary. A specific number is always returned here; FIRQB+FQDEVN+1 is always nonzero.
FIRQB+FQCLUS	The file cluster size is returned in this word.
FIRQB+FQNENT+1	Internal flag information:

Bit	Meaning When Set
1	File is placed
2	Some job has write access now
3	File is open in update mode
4	File is contiguous; no extend allowed
5	No delete or rename allowed
7	File is marked for deletion

CALFIP
LOKFQ

Errors

BADNAM	No file specification appears at FIRQB+FQNAM1 or the file name is invalid.
NOSUCH	Either the device specified at FIRQB+FQDEV and FIRQB+FQDEVN is not a disk, or no match exists for the occurrence specified in the word at FIRQB+FQFIL.
PAKLCK	The disk is restricted and the caller does not have DEVICE privilege.

Example

The following code asks the monitor to search the directory for account [2,130] for the first occurrence of a file specification beginning with the letter X:

```
CALL    CLRFQB                        ;MAKE SURE FIRQB
CALL    CLRXRB                        ;AND XRB ARE CLEAR
MOV     #FILNAM,XRB+XRLOC             ;POINT TO FILE NAME
MOV     #NAMSIZ,XRB+XRLEN             ;SET ITS LENGTH
MOV     #NAMSIZ,XRB+XRBC              ;AND AGAIN
.FSS                                  ;CONVERT TO FIRQB FORMAT
MOVB    #LOKFQ,FIRQB+FQFUN            ;SET FUNCTION CODE
CALFIP

FILNAM:   .ASCII "[2,130]X?????.*"    ;STRING FOR .FSS TO CONVERT
NAMSIZ = ._FILNAM                     ;AND ITS LENGTH
```

See Programming Hints for information about the CLRFQB and CLRXRB routines.

OPNFQ (Open a File/Device on a Channel)

Form

```
  MOVB    #OPNFQ,FIRQB+FQFUN
          .
          .
          .
  (Set parameters appropriate to file or device)
          .
          .
          .
  CALFIP
```

Function

The OPNFQ function has the same effect as an OPEN FOR INPUT statement in BASIC-PLUS; it opens a device or existing file on a channel. Parameters defining the device, file name and type, protection code, and mode are passed to the monitor in the FIRQB. If a file name is given in the FIRQB, a file-structured open for input is performed. If no file name is given, a non-file-structured open for input is performed. The *RSTS/E Programming Manual* describes file- and non-file-structured OPEN FOR INPUT statement and the actions taken for the mode parameter (MODE modifier in BASIC-PLUS) for each device.

Note

> Privileges are required to open a disk for non-file-structured processing. Whenever you use a disk as a non-file structured device, be aware that all RSTS/E data structures you access are subject to change at any time. The same applies if you open a UFD as a file.

Privileges Required

No privileges are required to open the caller's UFD for read-only access. Privileges are needed to perform the following:

- GREAD to open the UFD of any other account in the caller's group for read-only access
- WREAD to open the UFD of any account outside the caller's group for read-only access
- DEVICE to open a restricted device
- Read access (protection code, GREAD, or WREAD) to open a disk file for read-only access

- o Write access (protection code, GWRITE, WWRITE, and/or SYSIO) to open a disk file for write access
- o EXQTA to specify detach on close option on a pseudo keyboard
- o TUNE to select data caching mode on open
- o RDNFS to gain non-file-structured read access to a disk
- o SYSMOD to gain non-file-structured write access to a mounted disk
- o WRTNFS to gain non-file-structured write access to a disk, or to gain write access to a UFD

Data Passed

```
                                FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
             +---------------------+---------------------+
        1    |///////////////////////////////////////////|   0
             +---------------------+---------------------+
FQFUN   3    |     OPNFQ (= 2)     |/////////////////////|   2
             +---------------------+---------------------+
        5    |/////////////////////| channel number * 2  |   4 FQFIL
             +---------------------+---------------------+
        7    |   project number    |  programmer number  |   6 FQPPN
             +---------------------+---------------------+
       11    |          file name in RAD50 format        |  10 FQNAM1
             +                                           +
       13    |                 (2 words)                 |  12
             +---------------------+---------------------+
       15    |     file type in RAD50 format (1 word)    |  14 FQEXT
             +---------------------+---------------------+
       17    |    receive buffer size for DMC11/DMR11    |  16 FQSIZ
             +---------------------+---------------------+
       21    |///////////////////////////////////////////|  20
             +---------------------+---------------------+
       23    |                    mode                   |  22 FQMODE
             +---------------------+---------------------+
       25    |///////////////////////////////////////////|  24
             +---------------------+---------------------+
       27    |///////////////////////////////////////////|  26
             +---------------------+---------------------+
       31    |      device name (2 ASCII characters)     |  30 FQDEV
             +---------------------+---------------------+
       33    |  ≠0, unit no. real  | device unit number  |  32 FQDEVN
             +---------------------+---------------------+
       35    | no. rec. bfrs. to allocate for DMC11/DMR11|  34 FQCLUS
             +---------------------+---------------------+
       37    |///////////////////////////////////////////|  36
             +---------------------+---------------------+
```

FIRQB+FQFUN	The function code OPNFQ (octal value = 2).
FIRQB+FQFIL	Channel number times two; defines the channel on which the file is to be opened.
FIRQB+FQPPN	The PPN of the file to be opened. The project number is in the high byte (FQPPN+1), and the programmer number in the low byte (FQPPN). A value of zero defaults to the PPN under which the calling program is running. (Zero for non-file-structured open.)

FIRQB+FQNAM1	The file name to be opened; two words of RAD50 data. May also be specified as a zero word followed by the file identification index (see LOKFQ). (Must be two zero words for a non-file-structured open.)
FIRQB+FQEXT	The file type; one word of RAD50 data. (Must be zero for a non-file-structured open.)
FIRQB+FQSIZ	This parameter has the same function as the FILESIZE option in BASIC-PLUS. It is only used for the DMC11/DMR11, where it specifies the receive buffer size. You can specify a value between 1 and 632 (1170 octal). See the *RSTS/E Programming Manual* for more information.
FIRQB+FQMODE	The mode with which the file is to be opened; values and actions taken are as described for the MODE modifier in OPEN FOR INPUT and non-file-structured OPEN statements for various devices, as described in the *RSTS/E Programming Manual.* If you use a mode value at all, you must set bit 15 of this word to 1.
FIRQB+FQDEV	The device name is passed here as two ASCII characters. A value of zero indicates "SY", public disk.
FIRQB+FQDEVN	The device unit number is passed here in binary. A nonzero value in the high byte of this word (FIRQB+FQDEVN+1) indicates an explicit device unit number. A zero value in FIRQB+FQDEVN+1 indicates no unit number.
FIRQB+FQCLUS	This parameter has the same function as the CLUSTERSIZE option in BASIC-PLUS. For OPNFQ, it is used only for the DMC11/DMR11, where it specifies the number of receive buffers to allocate. You can specify a value between 1 and 127, but values greater than four are not recommended. See the *RSTS/E Programming Manual* for more information.

Data Returned

```
                                FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                 Offset  monic
             +---------------------+---------------------+
        1    |///////////////////////////////////////////|  0
             +---------------------+---------------------+
        3    |/////////////////////| current job no. * 2 |  2 FQJOB
             +---------------------+---------------------+
FQSIZM  5    |  MSB of file size   |   channel no. * 2   |  4 FQFIL
             +---------------------+---------------------+
        7    |  project number     | programmer number   |  6 FQPPN
             +---------------------+---------------------+
       11    |        file name in RAD50 format          | 10 FQNAM1
             +                                           +
       13    |                (2 words)                  | 12
             +---------------------+---------------------+
       15    |    file type in RAD50 format (1 word)     | 14 FQEXT
             +---------------------+---------------------+
       17    |             LSB of file size              | 16 FQSIZ
             +---------------------+---------------------+
       21    |    reasonable buffer size for device      | 20 FQBUFL
             +---------------------+---------------------+
       23    |               (as passed)                 | 22 FQMODE
             +---------------------+---------------------+
       25    |            device description             | 24 FQFLAG
             +---------------------+---------------------+
FQPROT 27    |  protection code    | clustersize, mod256 | 26 FQPFLG
             +---------------------+---------------------+
       31    |    device name (2 ASCII characters)       | 30 FQDEV
             +---------------------+---------------------+
       33    |     flag byte       | device unit number  | 32 FQDEVN
             +---------------------+---------------------+
       35    |        file identification index          | 34 FQCLUS
             +---------------------+---------------------+
       37    |///////////////////////////////////////////| 36 FQNENT
             +---------------------+---------------------+
```

Note

For nondisk devices, the relevant information returned with the OPNFQ subfunction is in the two words at FIRQB+FQBUFL and FIRQB+FQFLAG. All other words are returned as passed.

FIRQB+FQJOB	The current job number times two.
FIRQB+FQFIL	Channel number times two; defines the channel on which the file is open.

FIRQB+FQSIZM	For large disk files (greater than 65,535 blocks), this byte contains the MSB of the file's size in 512-byte blocks. This byte is combined with the word at FIRQB+FQSIZ to form a 24-bit field giving the file size.
FIRQB+FQPPN	The PPN under which the file is open. An actual PPN is returned here even if this word was passed as zero.
FIRQB+FQNAM1	The file name; two words of RAD50 data.
FIRQB+FQEXT	The file type; one word of RAD50 data.
FIRQB+FQBUFL	Reasonable buffer size for this device, in bytes. If you are doing device-independent I/O (that is, if you do not want to keep track of which device is being opened and perform specific opens, reads, and writes, depending on the device), this value is the monitor's best guess for a buffer size to use in subsequent reads and writes on the opened channel. (See the .READ and .WRITE directives.)
FIRQB+FQFLAG	Description of the device just opened. The low byte contains the device's handler index. There is one unique handler index for all device types. The high byte contains a set of status flags to allow for device-independent I/O routines.

```
                      DDNFS
                   DDRLO   |
                DDWLO   |  |
             FLGPOS |   |  |
          FLGMOD |  |   |  |
       FLGFRC |  |  |   |  |
    FLGKB  |  |  |  |   |  |
FLGRND |   |  |  |  |   |  |
   |   |   |  |  |  |   |  |
  15  14  13 12 11 10   9  8| 7                  0
  +-----------------------+--------------------+
  |   device-type flags   | device handler index |
  +-----------------------+--------------------+
```

High Byte -- Device-Type Flags

The bits in the high byte of the flags word are set to indicate the type of file or device just opened.

FLGRND = 1 The device or file is random-access.
       = 0 The device or file is sequential.

FLGKB	= 1	The file or device is a terminal-type file or device (or is generically a terminal).
	= 0	The file or device is not a terminal-type file or device.
FLGFRC	= 1	The file or device is byte-oriented. That is, the .READ and .WRITE directives handle data in byte units.
	= 0	The file or device is block-oriented. The .READ and .WRITE directives handle data in block units.
FLGMOD	= 1	The file or device accepts modifiers in .READ and .WRITE directives.
	= 0	The file or device does not accept modifiers in .READ and .WRITE directives.
FLGPOS	= 1	The file or device keeps track of its horizontal position and expands characters such as TAB into whatever is appropriate for the file or device. You can determine the current horizontal position with the .POSTN directive.
	= 0	The file or device does not keep track of its horizontal position.
DDWLO	= 1	The file or device has been write-locked (with the protection code or mode value in the open) or is generically a read-only device.
	= 0	The file or device is not write-locked.
DDRLO	= 1	The file or device has been read-locked (with the protection code in the open) or is generically a write-only device.
	= 0	The file or device is not read-locked.
DDNFS	= 1	The file or device is non-file-structured (or is generically not a file-structured device).
	= 0	The file or device is file-structured.

Low Byte -- Device Handler Index

Bits 0-7 of the flags word contain a handler index that indicates the generic kind of device. The currently defined values follow.

Octal Value	Symbol	Meaning
0	DSKHND	All disks
2	TTYHND	All terminals
4	DTAHND	DECtape
6	LPTHND	All line printers
10	PTRHND	Paper tape reader
12	PTPHND	Paper tape punch
14	CDRHND	Card reader
16	MTAHND	Magnetic tape
20	PKBHND	Pseudo keyboards
22	RXDHND	Flexible diskettes
24	RJEHND	2780 remote job entry
26	NULHND	The null device
30	DMCHND	The DMC11/DMR11 DDCMP interface
36		Reserved
40	KMCHND	KMC11
42	IBMHND	IBM interconnect
46	DMPHND	DMP11/DMV11 device

FIRQB+FQPFLG	The file cluster size, modulo 256. That is, a file cluster size of 256 is indicated a zero.
FIRQB+FQPROT	The protection code of the file.
FIRQB+FQDEV	The device name; two ASCII characters. (For disk, the actual device name is returned, even if a zero word was passed in this word.)
FIRQB+FQDEVN	The device unit number. (For disk devices, the actual unit number is returned here, even if FIRQB+FQDEVN+1 was passed as zero.)

FIRQB+33 For a file-structured open, this byte contains two relevant bits:

Bit 0 = 0	The device is in the public structure
Bit 0 = 1	The device is a private disk
Bit 1 = 0	A specific device was not specified
Bit 1 = 1	A specific device was specified

These bits are meaningless for a non-file-structured open.

FIRQB+FQCLUS The file identification index of this file. This word is significant mainly in that it can be used in place of the file name in subsequent opens of the file on disk. You can open the file with the OPNFQ subfunction by using an explicit PPN in FIRQB+FQPPN, a zero word in FIRQB+FQNAM1, and the file identification index in FIRQB+FQNAM1+2.

Note that there is no performance gain in using the file identification index instead of the file name. The file identification index is provided for compatibility with RSX. Furthermore, the file identification index is changed when the REORDR utility is run on the directory (see the *RSTS/E System Manager's Guide*).

Errors

NOTCLS	The specified channel is already open. It must be closed before it can be opened again.
PRVIOL	You do not have the necessary privilege and you tried to: o Open a disk for non-file-structured access. o Open a device that the system manager has restricted to users with DEVICE privilege.
xxxxx	All other possible errors are device-dependent. See Appendix A for a full list of errors.

Example

The following MACRO code sets up the FIRQB for the OPNFQ function. A non-file-structured open of magnetic tape unit 2 is done on channel 3:

```
  MAG:   .ASCII  MT
           .
           .
           .
         CALL    CLRFQB                 ;CLEAR FIRQB
         MOVB    #OPNFQ,FIRQB+FQFUN     ;SET FUNCTION CODE
         MOVB    #3*2,FIRQB+FQFIL       ;SET CHANNEL = 3
         MOV     MAG,FIRQB+FQDEV        ;SET DEVICE = MT
         MOVB    #377,FIRQB+FQDEVN+1    ;SET FLAG DEVICE NO. EXPLICIT
         MOVB    #2,FIRQB+FQDEVN        ;SET DEVICE NO. = 2
         CALFIP
```

See Programming Hints for information on the CLRFQB routine.

RENFQ (Rename a File)

Form

```
  MOVB      #RENFQ,FIRQB+FQFUN
     .
     .
     .
  (Define file to be renamed and new name in FIRQB)
     .
     .
     .
  CALFIP
```

Function

The RENFQ function renames an existing file on disk or DECtape and, if requested, deletes any existing file with the new name.

Privileges Required

DEVICE is required if the device is restricted. Create/rename access (GWRITE, WWRITE, and/or SYSIO) to rename a disk file outside the caller's account. SYSIO to set the privilege bit (bit 7 in FQPROT).

Data Passed

```
                                FIRQB
Mne-   Octal                                              Octal   Mne-
monic  Offset                                             Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|   0
          +---------------------+---------------------+
FQFUN   3 |     RENFQ (= 10)    |/////////////////////|   2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|   4
          +---------------------+---------------------+
        7 |   project number    | programmer number   |   6  FQPPN
          +---------------------+---------------------+
       11 |                 file name                 |  10  FQNAM1
          +              in RAD50 format              +
       13 |                 (2 words)                 |  12
          +---------------------+---------------------+
       15 |     file type in RAD50 format (1 word)    |  14  FQEXT
          +---------------------+---------------------+
       17 |   -1 to delete existing file (disk only)  |  16  FQSIZ
          +---------------------+---------------------+
       21 |               new file name               |  20  FQNAM1
          +              in RAD50 format              +
       23 |                 (2 words)                 |  22
          +---------------------+---------------------+
       25 |  new file type in RAD50 format (1 word)   |  24  FQFLAG
          +---------------------+---------------------+
FQPROT 27 |    file protection  |  = 0 if no change   |  26  FQPFLG
          +---------------------+---------------------+
       31 |  device name (disk) (2 ASCII characters)  |  30  FQDEV
          +---------------------+---------------------+
       33 | ≠0, device no. real | device unit number  |  32  FQDEVN
          +---------------------+---------------------+
       35 |///////////////////////////////////////////|  34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////|  36
          +---------------------+---------------------+
```

FIRQB+FQFUN	The function code RENFQ (octal value = 10).
FIRQB+FQPPN	The PPN of the existing file to be renamed. The project number is in the high byte (FIRQB+FQPPN+1), and the programmer number is in the low byte (FIRQB+FQPPN). A value of zero defaults to the PPN under which the calling program is running.
FIRQB+FQNAM1	The old (existing) name for the file; two words of RAD50 data.
FIRQB+FQEXT	The old (existing) type for the file; one word of RAD50 data.

FIRQB+FQSIZ	This word is set to -1 to indicate that any existing file on the specified device with the new name is to be deleted. If any other value is given here and a file already exists with the new name, the RENFQ function will return an error.
FIRQB+FQNAM2	The new file name and type; three words of RAD50 data. You can use RENFQ to change the protection code on a file by setting FQPROT and making these three words the same as the three words beginning at FIRQB+FQNAM1.
FIRQB+FQPROT	The new protection code for the file, if any, is specified in this byte. To retain the old protection code, this entire word (FIRQB+FQPFLG and FIRQB+FQPROT) must be zero. If the word is nonzero, the high byte will be used as the new protection code.
FIRQB+FQDEV	The device name is passed here as two ASCII characters. It must be a disk or DECtape device. A value of zero in this word indicates the public disk structure (_SY:).
FIRQB+FQDEVN	The device unit number is passed here in binary. A nonzero value in FIRQB+FQDEVN+1 indicates an explicit device unit number. A zero value in FIRQB+FQDEVN+1 indicates the system default.

Data Returned

Other than a possible error in byte 0 of the FIRQB, the RENFQ function of CALFIP does not return any meaningful data.

Errors

FIEXST	The new file name specified already exists.
NOSUCH	The old file specified cannot be found.

Example

The following code renames the file OLDNAM.TXT to NEWNAM.TXT on the public disk structure under the caller's account. FQSIZ is set to -1, so any existing file named NEWNAM.TXT is deleted:

```
CALL    CLRFQB                          ;CLEAR FIRQB
MOVB    #RENFQ,FIRQB+FQFUN              ;SET FUNCTION CODE
MOV     #^ROLD,FIRQB+FQNAM1             ;SET OLD FILE NAME
MOV     #^RNAM,FIRQB+FQNAM1+2           ;AND TYPE
MOV     #^RTXT,FIRQB+FQEXT              ;TO "OLDNAM.TXT"
MOV     #-1,FIRQB+FQSIZ                 ;DELETE EXISTING FILE
MOV     #^RNEW,FIRQB+FQNAM2             ;SET NEW FILE NAME
MOV     #^RNAM,FIRQB+FQNAM2+2           ;AND TYPE
MOV     #^RTXT,FIRQB+FQNAM2+4           ;TO "NEWNAM.TXT"
CALFIP
```

See Programming Hints for information about the CLRFQB routine.

RSTFQ (Reset a Channel)

Form

```
  MOVB     #RSTFQ,FIRQB+FQFUN
     .
     .
     .
  (Define channel to be reset)
     .
     .
     .
  CALFIP
```

Function

The RSTFQ function closes a channel, all channels, or all channels except one without performing any of the normal clean-up operations. For example, no trailer tape is written to paper tape punch; no form feed is given on the line printer; and no trailer labels are written to magnetic tape. This function is useful as a backup to a normal close operation. If a normal close fails, RSTFQ will close the channel regardless. You can also use RSTFQ to closes a channel on which a tentative file is open if you do not want to make the file permanent. The RSTFQ directive functions the same as a CLOSE statement with a negative channel number in BASIC-PLUS. See the *RSTS/E Programming Manual* for description of tentative files.

Privileges Required

None

Data Passed

```
                              FIRQB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
          +--------------------+--------------------+
        1 |////////////////////////////////////////| 0
          +--------------------+--------------------+
FQFUN   3 |    RSTFQ (= 20)    |////////////////////| 2
          +--------------------+--------------------+
        5 |////////////////////| channel number * 2 | 4 FQFIL
          +--------------------+--------------------+
        7 |////////////////////////////////////////| 6
          +--------------------+--------------------+
       11 |////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |////////////////////////////////////////| 14
          +--------------------+--------------------+
       17 |////////////////////////////////////////| 16
          +--------------------+--------------------+
       21 |////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQFUN	The function code RSTFQ (octal value = 20).
FIRQB+FQFIL	The value is set as follows:

- o To reset one channel, set FIRQB+FQFIL to the channel number to reset times two
- o To reset all channels, set FIRQB+FQFIL to zero
- o To reset all but one channel, set FIRQB+FQFIL to -(channel number times two); that is, the negative of two times the channel number to remain open

.CCL -- Check String for CCL Command

Form

```
.CCL
```

Function

The .CCL directive asks the monitor to check a string (defined in the XRB) to see if it is a valid Concise Command Language (CCL) command. If the string is a valid CCL command, the monitor passes control to the appropriate run-time system for that CCL (as defined by the system manager), using the equivalent of a .RUN directive. If the directive is successful, control does not return in line.

Control passes to the run-time system at the location specified in the P.RUN word in the pseudovector region (Chapter 2). Data in the job's CORCMN, XRB, FIRQB, and KEY areas in the low segment are passed to the run-time system because they are of interest to the run-time system being entered as a result of a .CCL, not to the caller. The file containing the program to be run as a result of the .CCL command is opened on channel 15. See Chapter 2 for a description of these areas in the P.RUN word.

If the string is not a valid CCL command, the monitor returns control to the caller (the run-time system or user job image that issued the .CCL directive) with no error. Control resumes with the instruction following the .CCL. Since control would not be returned here otherwise, the program does whatever housekeeping it deems necessary here for an unsuccessful CCL.

The system manager defines the CCL command, an abbreviation point, the name of the file that is to be executed when the command is given, and an entry point for the program (see the *RSTS/E System Manager's Guide*).

The command can be a string from one to nine characters long. The allowed single-character commands are A through Z, @, $, and #. Commands that are longer than one character must begin with a letter; the remaining characters can be letters or digits.

The abbreviation point defines how many characters must be specified before the command is accepted as valid. For example, if "DIRECTORY" were defined as a CCL command, the system manager could indicate three characters as the abbreviation point for the command. Then DIR, DIRE, DIREC and so forth, up to the full DIRECTORY, would all be interpreted by the monitor as correct CCL commands. (The monitor always fills in the full CCL command in CORCMN when it passes control to the appropriate run-time system.)

When you issue a .CCL directive, the monitor compares the indicated string with the commands defined by the system manager:

- o All null (ASCII code 000) and delete (ASCII code 177) characters are ignored and are never passed on to the run-time system in CORCMN.

- o ASCII code 200 is ignored and is never passed on to the run-time system in CORCMN.

- o Leading spaces (ASCII code 040) and tabs (ASCII code 011) are ignored and are never passed on to the run-time system in CORCMN.

- o When not enclosed by the quote characters " (ASCII code 042) and ' (ASCII code 047):

  - All tabs (ASCII code 011) are changed to spaces (ASCII code 040).

  - All control characters (ASCII codes 001 through 037, inclusive, and ASCII codes 201 through 237, inclusive) are ignored and never passed in CORCMN.

  - Adjacent spaces (ASCII code 040) are merged into a single space.

  - All lowercase alphabetics (ASCII codes 141 through 172, inclusive, and ASCII codes 341 through 376, inclusive) are changed into their uppercase equivalents (ASCII codes 101 through 132, inclusive, and ASCII codes 301 through 336, inclusive).

- o When enclosed by the quote characters " (ASCII code 042) and ' (ASCII code 047), all characters are kept as is and passed on to the run-time system in CORCMN.

Immediately following the CCL command text (for example, DIR/SI:n/DET) the monitor also analyzes two switches. These switches are passed on to the run-time system as status flags set in the XRB (see the P.RUN description in Chapter 2). The two switches may appear in either order, but must immediately follow the command. Both are optional switches.

The /SIZE switch has the format:

```
[space]/SI[Z[E]]:[+][#]n[.]
```

where n indicates the size, in K words, of the user job image that the program, when executed, will require. If the + sign is given, n indicates the additional amount of space, in K words, that the file will require over that indicated by the computed size or minimum size (see the description of the PF.CSZ bit in P.FLAGS word, Chapter 2). If the + sign is omitted, then n simply indicates the size, in K words, at which the invoked file should run. If the # sign is given, n is assumed to be octal. If the period (.)is given, n is assumed to be decimal. If both are given, an error is returned, and if neither is given, n is assumed to be decimal. The value of n must be between 1 and the system-wide maximum for a user job image (see the SWAP MAX discussion in the *RSTS/E System Installation and Update Guide*).

The /DETACH switch has the format:

```
[space]/DET[A[C[H]]]
```

This switch indicates that the invoked program should be run detached. In this state, channel 0 (the terminal associated with the job) is marked as detached while the program is running. This switch is useful for noninteractive programs; it frees the terminal for other use and prevents the user from interrupting the job by entering a CTRL/C.

Remember that the monitor simply examines these switches and passes the information on to the run-time system. The run-time system is responsible for doing the appropriate processing. For more information on the CCL facility, see the *RSTS/E Programming Manual*.

Note

> Do not use this directive from a user job image running under the RT11 run-time system, since the lowest 512. bytes are used by RT11 differently from other run-time systems. Instead, use .DOCCL (see Chapter 7).

Privileges Required

Execute access (by protection code, GREAD, or WREAD) is needed to execute a program for a CCL command.

Data Passed

```
                                  XRB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
          +--------------------+--------------------+
        1 |  length of proposed command string, bytes |  0 XRLEN
          +--------------------+--------------------+
        3 |  length of proposed command string, bytes |  2 XRBC
          +--------------------+--------------------+
        5 |    starting address of command string    |  4 XRLOC
          +--------------------+--------------------+
        7 |//////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |                                          | 10 XRBLK
          +               (passed to                 +
       13 |           new run-time system            | 12
          +               unaltered)                 +
       15 |                                          | 14
          +--------------------+--------------------+
```

XRB+XRLEN	The length of the proposed CCL command string, in bytes.
XRB+XRBC	The length of the proposed CCL command string, in bytes, is also passed in this word.
XRB+XRLOC	The starting address of the proposed CCL command string. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)
XRB+XRBLK	The remaining three words are unaltered here if the string is indeed a CCL command and the monitor takes over and does the equivalent of a .RUN directive. That is, the run-time system that is given control as a result of this command finds the same three words here that the caller left. These three words are also be unchanged if control returns back to the caller for any reason. See the .RUN monitor call in this Chapter and the P.RUN entry point in Chapter 2 for details.

Data Returned

No useful data is returned to the calling program. (The P.RUN description describes the data passed to the run-time system if the call is successful.) If the call is unsuccessful, the last three words of the XRB are unaltered, but the first four words are random. In addition, an error code is returned in the first byte of the FIRQB.

Errors

(none)	No error is returned if the command part of the string passed was not a valid CCL command. The contents of CORCMN have not been altered; the XRB (except for the last three words) has been altered.
BADCNT	The first three words of the XRB, which describe the CCL command string, are illegal.
BADSWT	An illegal switch was given in the CCL command string.
BDNERR	An illegal number was the argument to one of the switches found in the CCL command string. For example, the n value in the /SIZE switch was greater than the system-wide maximum for a user job image (see the SWAP MAX discussion in the *RSTS/E System Installation and Update Guide*).
LINERR	The indicated string is too long to be passed in CORCMN.
xxxxxx	Any other error returned results from the monitor's execution of a .RUN directive for the program. See the .RUN directive in this chapter.

Example

The following example asks the monitor to check a 72-byte string beginning at location BUFFER to see if it is a CCL command:

```
BUFFER: .BLKB   72.
          .
          .
          .
        CALL    CLRXRB                  ; Clear XRB
        MOV     #72.,XRB+XRLEN          ; Set length
        MOV     #72.,XRB+XRBC           ; Set length again
        MOV     #BUFFER,XRB+XRLOC       ; Set starting address
        .CCL
```

See Programming Hints for information on the CLRXRB routine.

.CHAIN - Execute Under Same RTS

Form

```
  .CHAIN
```

Function

The .CHAIN directive is the same as the .RUN directive, except that it returns an error if the program to be run would cause a new run-time system to be entered. That is, if the call succeeds, the current run-time system is entered at the P.RUN entry point (see Chapter 2). In addition, there is no change in the user job image size. Otherwise, the error code NORTS is returned in byte 0 of the FIRQB.

Use this call to bypass the special protection afforded by the compiled-file bit in the protection code. The run-time system can use this directive to access an executable file without the problem of possibly losing control (unlike .RUN, which can transfer control to a different run-time system). The .CHAIN directive reenters the run-time system so a user cannot take control of the file once it is open on channel 15.

See the .RUN directive, for Data Passed, Data Returned, and Errors. For the example, substitute .CHAIN for .RUN.

Note

> Do not use this directive from a user job image running under the RT11 run-time system. Use the RT11 .CHAIN directive (see Chapter 7) to transfer control to another program running under RT11.

Privileges Required

Execute access (by protection code, GREAD, or WREAD) is required to execute a program.

.CLEAR - Clear Keyword Bits

Form

.CLEAR

Function

Use the .CLEAR directive to clear certain bits in the keyword (KEY) location in the user job image. The XRB passes the bits to be cleared to the monitor.

Privileges Required

None

Data Passed

```
                                 XRB
Mne-  Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
         +---------------------+---------------------+
       1 |   bits to be cleared are set to 1 here     |  0 XRLEN
         +---------------------+---------------------+
       3 |///////////////////////////////////////////|  2
         +---------------------+---------------------+
       5 |///////////////////////////////////////////|  4
         +---------------------+---------------------+
       7 |///////////////////////////////////////////|  6
         +---------------------+---------------------+
      11 |///////////////////////////////////////////| 10
         +---------------------+---------------------+
      13 |///////////////////////////////////////////| 12
         +---------------------+---------------------+
      15 |///////////////////////////////////////////| 14
         +---------------------+---------------------+
```

XRB+XRLEN   The bits to be cleared are set to 1 here.

```
                  JFSPRI---------+
               JFFPP---------+   |
            JFPRIV-------+   |   |
         JFSYS-------+   |   |   |
      JFNOPR-----+   |   |   |   |
   JFBIG-----+   |   |   |   |   |
JFLOCK---+   |   |   |   |   |   |
         |   |   |   |   |   |   |
  15  14  13  12  11  10   9   8 | 7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

JFLOCK	Clearable by any caller. Clearing JFLOCK indicates that the job wants to be swapped. When JFLOCK is clear, the monitor swaps the job (that is, the user job image) to and from disk as necessary.
JFBIG	Not clearable by any caller; masked off. That is, the corresponding bits in KEY cannot be cleared by the job with the .CLEAR directive.
JFNOPR	Not clearable by any caller; masked off.
JFSYS	Clearable by any caller. If the job has temporary privileges and this bit is cleared in a .CLEAR call, the temporary privileges are temporarily lost.
JFPRIV	If a .CLEAR directive is issued with this bit indicated for clearing, any temporary privileges that the job has or had are permanently lost.
JFFPP	Clearable by any caller. Clearing JFFPP indicates that this job no longer wants the state of the hardware floating-point unit (if any) to be swapped along with the job's normal context information.
JFSPRI	Clearable by any caller. Clearing JFSPRI lowers the job's run priority by one-half step. That is, it clears bit 2 of the system- controlled low-order three bits of the run priority. See the SET JOB command *RSTS/E System Manager's Guide*.

All other bits in the XRB are masked off.

Data Returned

The .CLEAR directive does not return any meaningful data.

Errors

No errors are possible with the .CLEAR directive.

Example

The following code clears the JFLOCK bit:

```
  CALL    CLRXRB                        ;CLEAR XRB
  MOV     #JFLOCK,XRB+XRLEN             ;SET JFLOCK FOR CLEAR
  .CLEAR
```

See Programming Hints for information on the CLRXRB routine.

.CMDLN -- Read/Write Command Line

Form

.CMDLN

Function

The .CMDLN directive lets programs pass up to 1K words of data (2048 characters) when chaining to another program. DCL uses this capability to pass long command lines to some server programs. If possible, the .CMDLN directive stores the data in core common. If not, it stores the data in XBUF.

Note

The read function deletes the command line after it has been read, therefore the command line can only be read once.

Privileges Required

None

Data Passed (Write a Command Line)

```
                              XRB
Mne-  Octal                                     Octal   Mne-
monic Offset                                     Offset  monic
         +--------------------+--------------------+
       1 |      length of command line to write     |  0 XRLEN
         +--------------------+--------------------+
       3 |      length of command line to write     |  2 XRBC
         +--------------------+--------------------+
       5 |     address of command line to write     |  4 XRLOC
         +--------------------+--------------------+
       7 |//////////////////////////////////////////|  6
         +--------------------+--------------------+
      11 |//////////////////////////////////////////| 10
         +--------------------+--------------------+
      13 |//////////////////////////////////////////| 12
         +--------------------+--------------------+
      15 |//////////////////////////////////////////| 14
         +--------------------+--------------------+
```

XRB+XRLEN	The length of the command line to be written, in bytes.
XRB+XRBC	The length of the command line to be written, in bytes.
XRB+XRLOC	The starting address of the command line to be written. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)

Errors

BADCNT	XRB parameters were invalid, or the command line was too long.
NOBUFS	Not enough XBUF was available to store the command line.

Data Passed (Read a Command Line)

```
                                  XRB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +---------------------+---------------------+
        1 | length of buffer to receive command line  |   0 XRLEN
          +---------------------+---------------------+
        3 |                   zero                    |   2 XRBC
          +---------------------+---------------------+
        5 |      address to place command line        |   4 XRLOC
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|   6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////|  10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////|  12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////|  14
          +---------------------+---------------------+
```

XRB+XRLEN	The length of the buffer to receive the command line, in bytes.
XRB+XRBC	Zero to indicate a read operation.
XRB+XRLOC	Starting address of the area where the command line is placed. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)

Data Returned

XRB+XRBC	The length of the command line read, in bytes.

Errors

NOSUCH	There was no command line to be read.
LINERR	The command line was too long for the user's buffer.

.CORE -- Change Memory Size

Form

```
  .CORE
```

Function

The .CORE directive asks the monitor to change the amount of memory currently allocated for the user job image (low segment) for this job. The monitor preallocates space for a user job image at the time a .RUN directive is issued. The space is based on the size of the file (if PF.CSZ = 1 in the P.FLAG word of the pseudovector region) or is equal to the P.MSIZ word in the pseudovector region.

Note

> If the I&D space split is enabled, .CORE alters only the size of the D-space.

You can change this initial size with the .CORE directive as many times as you want, as long as the requested size:

- o Falls within a maximum and minimum value
- o Does not overlap any address windows created by the job for use with resident libraries (see the .PLAS directive in this chapter)

The monitor first checks the size requested against maximum and minimum values, using this algorithm:

```
                     +----------------+
1Kword ≤  P.MSIZ ≤   | size requested | ≤  private ≤  P.SIZE ≤  system
                     | with .CORE     |    maximum               maximum
                     +----------------+    for job
```

The monitor determines the maximum allowable amount of space for a user job image as follows:

- o Set <max> (the maximum size that a job image can be) to the system-wide maximum. This maximum is set by the system manager at startup time (see the SWAP MAX discussion in the *RSTS/E System Installation and Update Guide*).
- o If the maximum user job image size imposed by the run-time system (P.SIZE in the pseudovectors) is less than the current <max>, set <max> to P.SIZE.
- o If the job's private memory maximum is less than <max> and if the caller does not have EXQTA privilege, then set <max> to the job's private memory maximum. The system manager can set

a particular job's private memory maximum with the SET JOB/SIZE=n command. You can also set a private memory maximum with the UU.PRI subfunction of the .UUO directive (in this chapter). A job's private memory maximum is initially defaulted to the system maximum.

- The size of I-space and the size of D-space can not exceed 32K words each.

Thus, the size requested with .CORE is checked against <max>, as determined by the four previous steps. The size requested with .CORE is also checked against a minimum (the P.MSIZ word in the pseudo vectors, which must be greater than or equal to 1K words). Any size between P.MSIZ and <max>, inclusive, is legal.

There are two special cases:

- If the size requested with .CORE is exactly equal to the run-time system's minimum size (P.MSIZ), that request is considered legal even if the requested size is greater than the job's private maximum.

- If the size requested with .CORE is less than the job image's current size and within the allowable bounds for the run-time system, but it is still larger than the private maximum, that request is considered legal. This could happen if the caller had EXQTA privilege, allowing the current size to be greater than the private maximum, and then dropped EXQTA privilege. The monitor would still allow the size greater than the private maximum.

If .CORE requests a decrease in job image size, no further checks are made. If .CORE requests an expansion and the size is legal according to the cases previously described, the monitor then checks the base APRs of any address windows created by the job (see the CRAFQ subfunction of .PLAS in this chapter). Windows created by the CRAFQ subfunction of .PLAS require both I&D APRs for mapping. Therefore, if a task contained 20K words of I-space and 16K words of D-space, only 12K words is available for windows. If the size requested in the .CORE overlaps a created address window, the .CORE fails and returns an error.

If .CORE requests a legal expansion that cannot be made in place, that is, if there is not enough free memory available for the expansion, the job is swapped out and swapped back in at the larger size. (This swap occurs even when JFLOCK = 1 in the keyword (KEY).)

When a user job image expands, the content of the newly added memory is zeroed as protection against a malicious user reading memory to look for passwords.

Note

> Do not use this directive from a user job image running under the RT11 run-time system. Expanding memory size should be done through the RT11 emulator, using the appropriate RT11 directive.

When the image runs under the RSX run-time system, use the EXTK$ directive to extend the task size, so that subsequent GTSK$ directives can return the task size correctly.

Privileges Required

EXQTA to exceed private memory maximum.

Data Passed

```
                              XRB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
          +---------------------+---------------------+
        1 |       I&D flag      |desired size, K words|  0 XRLEN
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +-------------------------------------------+
        5 |///////////////////////////////////////////|  4
          +-------------------------------------------+
        7 |///////////////////////////////////////////|  6
          +-------------------------------------------+
       11 |///////////////////////////////////////////| 10
          +-------------------------------------------+
       13 |///////////////////////////////////////////| 12
          +-------------------------------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
```

XRB+XRLEN	The desired size for the user job image, in K words.
XRB+XRLEN+1	If this byte contains a -1, the user is requesting that the I&D APRs be split to map different physical memory. XRB+XRLEN is the requested size of the D-space. The I-space size is fixed at the job's current size. If this byte contains a -1 and XRB+XRLEN contains a zero, the I&D split is disabled.

Data Returned

Other than a possible error in the byte 0 of the FIRQB, the .CORE directive does not return any meaningful data.

Errors

EDBMCE	The requested user job image size is illegal. It is either too large or too small, or it overlaps a mapped window.
ERRERR	The program is requesting use of separate I&D space but the hardware does not exist on this system.
INUSE	There is outstanding asynchronous I/O for this job.

Example

The following code requests a user job image of 24K words:

```
  CALL    CLRXRB                          ;CLEAR THE XRB
  MOVB    #24.,XRB+XRLEN                  ;SET XRB TO INDICATE 24K WORDS
  .CORE
```

See Programming Hints for information on the CLRXRB routine.

.DATE - Return Current Date and Time

Form

.DATE

Function

The .DATE directive returns the current date and time, the current program name (as installed by the .NAME directive), and the current run-time system name in the XRB.

Privileges Required

None

Data Passed

The .DATE directive does not pass any meaningful data.

Data Returned

```
                              XRB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
         +--------------------+--------------------+
       1 |  current date, in system internal format |  0 XRLEN
         +--------------------+--------------------+
       3 |           minutes until midnight          |  2 XRBC
         +--------------------+--------------------+
       5 | ticks until second  | seconds until minute|  4 XRLOC
         +--------------------+--------------------+
       7 |       program name in RAD50 format        |  6 XRCI
         +                (2 words)                +
      11 |         (as installed with .NAME)         | 10
         +--------------------+--------------------+
      13 |    run-time system name in RAD50 format   | 12 XRTIME
         +                (2 words)                +
      15 |                                           | 14
         +--------------------+--------------------+
```

XRB+XRLEN The current date, in system internal format. The monitor calculates the date as:

[(year - 1970) * 1000] + (day-within-year)

where day-within-year is 1 for January 1, 2 for January 2, and so forth. (Every leap year, the day-within-year value for March 1 and following is one higher than in other years.)

XRB+XRBC	The number of minutes until midnight. A value of 1440 is midnight; 720 is noon; 0 is never returned.
XRB+XRLOC	The number of seconds until the next minute. A value of 60 is xx:xx:00; 1 is xx:xx:59; 0 is never returned.
XRB+5	The number of ticks until the next second. A tick is either 1/60th or 1/50th of a second, depending on the clock in use and/or the line frequency. (Systems running with the KW11P clock at crystal speeds, rather than at line frequency, have a tick of 1/50th of a second. Otherwise, if the system is operating with a 60 Hz power line, a tick is 1/60th of a second.)
XRB+XRCI	The current program name (as set by the most recent .NAME directive); two words in RAD50 format.
XRB+XRTIME	The current run-time system name; two words in RAD50 format.

Errors

No errors are possible with the .DATE monitor call.

Example

Since no data is passed to the monitor, the call is:

```
  .DATE
```

.ERLOG – Log an Error from RTS

Form

```
  .ERLOG
```

Function

You can issue the .ERLOG directive only from the high segment (run-time system). It allows the run-time system to log an error into the RSTS/E error log file, which can then be printed by the system manager (see the *RSTS/E System Manager's Guide*). For example, you might want to place an entry into the RSTS/E error logging scheme on a hardware floating-point unit exception that has an illegal error code because the monitor makes no such checks.

You can issue this directive only from the job's current run-time system (high segment). Since this call does not require privilege, the monitor does not to let users fill up the system error log with unimportant errors. If .ERLOG is issued from the user job image (low segment), RSTS/E ignores it.

Privileges Required

None

Data Passed

The .ERLOG directive records in the system error log file the contents of the program counter (PC) and program status word (PS) at the time of the call, as well as the contents of the general registers (R0 through R5). These registers will then be displayed at the system manager's request. Hence, the registers should contain whatever information you want to record when the .ERLOG is executed.

Data Returned

The .ERLOG directive does not return any meaningful data.

Errors

No error is possible with .ERLOG.

Example

If the general registers contain relevant information, the call is:

```
  .ERLOG
```

.EXIT -- Exit to Default Keyboard Monitor

Form

```
.EXIT
```

Function

The .EXIT directive returns control to the system default keyboard monitor (DCL) at the P.NEW entry point (see Chapter 2). When a program exits, it normally passes control to the job's keyboard monitor with the .RTS directive. You can use the .EXIT call as a backup to return control to the default keyboard monitor if the .RTS fails, or for any other reason when you want to enter the default keyboard monitor at the entry point specified by P.NEW. The .EXIT directive needs no arguments and never returns in line to the caller.

Privileges Required

None

Data Passed

The .EXIT directive needs no arguments; however, the three words beginning at XRB+10 are passed unaltered to the default keyboard monitor. The monitor also passes information to the default keyboard monitor when .EXIT is executed. See the discussion of the P.NEW entry point in Chapter 2.

Data Returned

The .EXIT directive does not return any meaningful data; control never returns in line.

Errors

No errors are possible with .EXIT.

Example

Since no data is passed or returned with .EXIT, the call is:

```
.EXIT
```

Note

Do not use this RSTS/E directive from a user job image running under the RT11 run-time system. The correct way to terminate such a program is to exit to the RT11 emulator, which then returns control to the job's keyboard monitor.

.FSS - Check File Specification String

Form

.FSS

Function

The .FSS directive examines a string of characters presumed to be a file specification and, if possible, converts it to the internal RSTS/E file specification format (FIRQB format). The monitor returns information to the XRB describing what it found in the string and returns the converted file specification to the FIRQB. Thus, programs that manipulate files can use .FSS to translate a user-typed string to the FIRQB format.

The monitor examines the string from left to right and stops without error when it finds:

- o The end of the string.
- o An equal sign (=), which is ASCII code 075.
- o A semicolon (;), which is ASCII code 073.
- o A slash (/), which is ASCII code 057, that is followed by anything other than the following switches, which the monitor translates to the FIRQB format:
  - /CL[USTERSIZE]:[-][#]n[.]

    where n is the cluster size used in opening files and devices. The variable n can specify a value ranging from -32768 through 32767.
  - /MO[DE]:[#]n[.]

    where n is the mode used in opening files and devices. The variable n can specify a value between 0 and 32767.
  - /FI[LESIZE]:[#]n[.] or /SI[ZE]:[#]n[.]

    where n is the filesize used in opening files and devices. The value of n defines the file's size in 512-byte blocks and can be in the range 0 to 8,388,607 blocks.
  - /PO[SITION]:n

    where n is the position used in creating files (to position block 1 of the file at a device cluster). The variable n can specify a value between -2 and 65,535.

- /PR[OTECTION]:[#]n[.]

  where n is the protection code used in opening or creating files. The variable n can specify a value between 0 and 255. The value of n determines the file's protection from users; see the *RSTS/E System User's Guide*.

  The brackets ([ ]) in the previous switches enclose optional characters. Where more than one character is enclosed in brackets, any or all of the enclosed characters can be omitted. For example, MO, MOD, and MODE would all be accepted and the following quantity translated to the mode location in the FIRQB.

  The value n is assumed to be decimal, unless the optional pound sign [#] appears, indicating that n is octal. The optional decimal point also indicates a decimal value.

o A comma (,), which is ASCII code 054. An exception is the comma separating the project and programmer numbers in a PPN.

The monitor translates the components in a file specification string as follows:

device name — A device name can be either a logical device name or a physical device name:

logical — A logical device name is a string of alphanumeric characters or $ terminated with a colon (:). The logical name can contain the $ character anywhere except as the first character. Only the first nine characters are examined; the remainder are ignored. If the logical name does not contain a $ character, the monitor checks the name against the user's own logical device name assignments in USRLOG (or its equivalent, as defined in the XRB). If the monitor finds a definition, it returns the physical device name associated with that logical device name to the FIRQB. If the logical name is not found in the user-logical area or if the logical name contains a $ character, the monitor makes a search of system-wide logical names. If the logical name is not found, the monitor returns the logical device name and sets a flag in the XRB to indicate that it could make no association. For a logical device name beginning with an underscore (_), the monitor does not attempt any translation to a physical device name.

physical	A physical device name consists of two alphabetic characters optionally followed by digits and ended with a colon (:). The digits are translated as decimal and must have a value between 0 and 127 (decimal). Leading zeros are allowed. For some devices (for example, terminals) a physical name can also be specified as a two-letter device name, a one-letter controller identifier, and a number specifying the unit on that controller. For example, TTG2: is the third terminal on a DZ11 controller.

account or PPN A PPN can be expressed either as a single special character or as two separate numbers enclosed in square brackets ([]) or parentheses () and separated by a comma.

The following special characters are translated as:

$	The dollar sign ($) is equivalent to [1,2].
!	The exclamation point (!) is equivalent to [1,3].
%	The percent sign (%) is equivalent to [1,4].
&	The ampersand (&) is equivalent to [1,5].
#	The number sign (#) is translated to the caller's group library. It is always equivalent to [proj,0] where proj is the project number of the user issuing the .FSS directive.
@	The at sign (@) is translated to the caller's assignable PPN, the USRPPN value (see Chapter 2). If USRPPN is set, its value is placed in the FIRQB at offset FQPPN. If USRPPN is zero, a string with an @ causes an error.

Note

DIGITAL does not recommend the use of the special characters !, %, &, #, or @. They are provided only for compatibility with pre-V9.0 releases of RSTS/E.

[n,m]	This is the explicit construct for a PPN. The value n specifies the project number, and m specifies the programmer number. The variables n and m can specify any value from 0 through 254, except for [0,0]. If a number sign (#) precedes either n or m, the string is assumed to specify an octal value. Either n or m, or both, can also be the asterisk (*). The * is converted to 255 and placed in its corresponding FIRQB location. The asterisk character indicates a wildcard specification. Either n or m, or both, can be omitted; in that case, the monitor uses the corresponding part of the user's PPN. For example, [,] or [] specifies the user's PPN; [,0] is equivalent to #.
(n,m)	This is an alternate way to specify an explicit PPN. The same rules for n and m apply as when they are enclosed by brackets. **Note** DIGITAL does not recommend the use of parentheses in the PPN specification. This format is provided only for compatibility with pre-V9.0 releases of RSTS/E.
file name	A file name can consist of alphanumeric characters and the question mark (?). It is the only field in the file specification with no explicit delimiter. Only the first six characters are examined; the rest are ignored. The asterisk character (*) is also an acceptable file name. It is translated to two words of RAD50, where each RAD50 character is the unused code (29). Each ? character is also converted to this unused code. This indicates that the file name field is a wildcard. (The LOKFQ subfunction of CALFIP and UU.LOK subfunction of .UUO can be used to look up wildcard files.)
type	A file type can consist of alphanumeric characters and the ? character, preceded by a . character. The * character is also an acceptable file type. It is translated to one word of RAD50, where each RAD50 code is the unused code (29). Each ? character in the file type is also converted to this unused code. This indicates that the file type field, or character in the file type field, is a wildcard.

	If given, the file type must always follow the file name.
protection code	A file protection code can consist of numeric digits enclosed by angle brackets (<>). The general form of a protection code is <nnn>, where n may be numeric characters indicating a value from 0 through 255. If the numeric characters are preceded by a # character, they are converted as specifying an octal value. Alternatively, the protection code can be specified using the /PROTECTION switch. If no file protection code is specified in the string and a default value has been assigned in USRPRT (see Chapter 2), the default value is placed in the FIRQB.

These special characters can appear in the string in any order, with the exception of the file type and PPN. The file type must follow the file name, if specified.

In addition, if the device name is a system or user logical device name that has an account (PPN) associated with it, the position of an explicit PPN in the file specification string is significant. If the order is device:[PPN], then the explicit PPN overrides the PPN associated with the logical device name. If the order is [PPN]device:, RSTS/E displays the error message ?Illegal device name.

Note

> Do not use this directive from a user job image running under the RT11 run-time system, since the user logical area is not in the standard location.

Privileges Required

None

Data Passed

```
                              XRB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +--------------------+--------------------+
        1 |      length of the string, bytes       |  0 XRLEN
          +--------------------+--------------------+
        3 |      length of the string, bytes       |  2 XRBC
          +--------------------+--------------------+
        5 |     starting address of the string     |  4 XRLOC
          +--------------------+--------------------+
        7 |////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |   length of nonstandard user defaults  | 10 XRBLK
          +--------------------+--------------------+
       13 |  starting address of nonstandard users | 12 XRTIME
          +--------------------+--------------------+
       15 |////////////////////////////////////////| 14
          +--------------------+--------------------+
```

XRB+XRLEN	The length of the character string to be translated, in bytes.
XRB+XRBC	The length of the character string to be translated, in bytes.
XRB+XRLOC	The starting address of the string to be translated. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)
XRB+XRBLK	If the user logical information (USRPPN, USRPRT, and USRLOG) is in its standard location, this word is passed as zero. If some nonstandard set of locations is being used, then the length of that information, in bytes, is specified here.
XRB+XRTIME	If the word at XRB+XRBLK is nonzero, then this word defines the starting location for the user logical information. (The order of the information is assumed to be the same as in its standard location; that is, the user logical PPN, user logical protection code, user logical device name table. The format is also expected to be the same (see the USRPPN, USRPRT, and USRLOG descriptions in Chapter 2 for details).

Data Returned

```
                                  XRB
Mne-  Octal                                                   Octal   Mne-
monic Offset                                                   Offset  monic
             +---------------------+---------------------+
           1 |/////////////////////////////////////////////|  0
             +---------------------+---------------------+
           3 |number of untranslated characters in string|  2 XRBC
             +---------------------+---------------------+
           5 |  address of first untranslated character  |  4 XRLOC
             +---------------------+---------------------+
           7 |/////////////////////////////////////////////|  6
             +---------------------+---------------------+
          11 |                flag word 2                 | 10 XRBLK
             +---------------------+---------------------+
          13 |                flag word 1                 | 12 XRTIME
             +---------------------+---------------------+
          15 |             device description             | 14 XRMOD
             +---------------------+---------------------+
```

XRB+XRBC	A count of the untranslated characters in the string. If all characters were translated, the value of this word is zero.
XRB+XRLOC	The address of the first untranslated byte of the string. If XRB+XRBC is zero, this word identifies the end of the string. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)
XRB+XRBLK	Bit flags describing the translated string. Note that this word is the same as "flag word 2" for the BASIC-PLUS file name string scan SYS call (see the *RSTS/E Programming Manual*).

Bit	Setting	Meaning
0	Set	A file name was found in the source string; two words in RAD50 format at FIRQB+FQNAM1.
	Clear	No file name was found (and bits 1 and 2 of this word are also 0).
1	Set	The translated file name consisted of a single * character and has been translated to two words at FIRQB+FQNAM1 consisting of the RAD50 representation of the string "??????".
	Clear	The translated file name was not an * character.
2	Set	The file name contained at least one ? character.

Bit	Setting	Meaning
	Clear	The file name did not contain any ? characters.
3	Set	A period (.) was found in the source string.
	Clear	No period was found, implying that no file type was specified (and bits 4, 5, and 6 of this word are also 0).
4	Set	A file type was found; that is, the field after the period was not null.
	Clear	No file type was found (the field after the period was null), and bits 5 and 6 of this word are also 0.
5	Set	The file type was an * character and is returned in the word at FIRQB+FQEXT as the RAD50 representation of the string "???".
	Clear	The file type was not an * character.
6	Set	The file type contained at least one ? character.
	Clear	The file type did not contain any ? characters.
7	Set	A PPN was found in the source string.
	Clear	No PPN was found (and bits 8 and 9 of this word are also 0).
8	Set	The project number was an * character. That is, the PPN was of the form [*,n]. RSTS/E returns a value of 377 at FIRQB+FQPPN+1.
	Clear	The project number was not an * character.
9	Set	The programmer number was an * character. That is, the PPN was of the form [n,*]. RSTS/E returns a value of 377 at FIRQB+FQPPN.
	Clear	The programmer number was not an * character.
10	Set	A valid protection code was found.
	Clear	No protection code was found.
11	Set	No file protection code was found in the string, but there was a default output file protection code in location USRPRT. The default has been returned in the FIRQB.

Bit	Setting	Meaning
	Clear	The user-assignable default protection code (at location USRPRT) was not used. Either zero or the protection code given in the string is returned to the FIRQB.
12	Set	A colon (:), but not necessarily a device name, was found in the source string.
	Clear	No colon was found (no device was specified); bits 13, 14, and 15 of this word are also 0.
13	Set	A device name was found in the source string.
	Clear	No device name was found; bits 14 and 15 of this word are also 0.
14	Set	The device name in the string was a logical device name.
	Clear	The device name in the string was an actual device name; bit 15 of this word is also 0.
15	Set	This bit set indicates an invalid device name. (The characters that were specified are simply returned at FIRQB+FQDEV, FIRQB+FQDEVN, and FIRQB+FQFLAG as three words in RAD50 format.) This bit can be set in one of two ways: o If the device name contained an underscore but was not a recognizable device name for any device on the system, this bit is set. o If the device name did not contain an underscore but the name could not be translated to a physical device name, this bit is set.
	Clear	The device name specified, if any, was either an actual device name or a logical device name to which a physical device has been assigned. The physical device name has been returned to the word at FIRQB+FQDEV as two ASCII characters, and the unit information has been returned appropriately at FIRQB+FQDEVN.

XRB+XRTIME Remaining bit flags describing what was translated. Some of these bits duplicate information returned at XRB+XRBLK. DIGITAL recommends that you use the bits at XRB+XRBLK to allow for enhancements in future releases. Note that this word is the same as "flag word 1" for the BASIC-PLUS file name string scan SYS call, see the *RSTS/E Programming Manual.*

Bit	Setting	Meaning
0	Set	The /CLUSTERSIZE:n switch was specified.
	Clear	The /CLUSTERSIZE:n switch was not specified.
1	Set	Either the /MODE:n or /RONLY switch was specified.
	Clear	Neither the /MODE:n nor the /RONLY switch was specified.
2	Set	The /FILESIZE:n or /SIZE:n switch was specified.
	Clear	Neither the /FILESIZE:n nor the /SIZE:n switch was specified.
3	Set	The /POSITION:n switch was specified.
	Clear	No /POSITION:n switch was specified.
4-7		(Not currently used.)
8	Set	A file name was found in the source string; two words in RAD50 format at FIRQB+FQNAM1. Note that this is the same meaning as for bit 0 at XRB+XRBLK.
	Clear	No file name was found in the source string.
9	Set	A period (.) was found in the source string. Note that this is the same meaning as for bit 3 at XRB+XRBLK.
	Clear	No period was found in the source string, implying that no file type was specified either.
10	Set	A PPN was found in the source string. Note that this is the same meaning as for bit 7 at XRB+XRBLK.
	Clear	No PPN was found.

Bit	Setting	Meaning
11	Set	A valid protection code was found. Note that this is the same meaning as for bit 10 at XRB+XRBLK.
	Clear	No protection code was found.
12	Set	A colon (but not necessarily a device name) was found in the source string. Note that this is the same meaning as for bit 12 at XRB+XRBLK.
	Clear	No colon was found, implying that no device could have been specified.
13	Set	Device name was specified and was a logical device name. Note that this is the same meaning as for bit 14 at XRB+XRBLK.
	Clear	Device name (if specified) was an actual device name. (If device name was not specified, this bit is also 0.)
14		(Not currently used.)
15	Set	Source string contained wildcard characters (either ?, *, or both) in file name, type, or PPN fields. In addition, the device name specified, although a valid logical device name, does not correspond to any of the logical device assignments currently in effect or contains an underscore as the first character. Flag word 2 contains more specific information.
	Clear	None of the above.

XRB+XRMOD The device description (the same information returned by the BASIC-PLUS STATUS variable and returned at FIRQB+FQFLAG when a file or device is opened with the OPNFQ or CREFQ subfunctions of CALFIP). The device handler index is in the low byte and descriptive flags are in the high byte.

```
                                FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
             +---------------------+---------------------+
           1 |///////////////////////////////////////////|  0
             +---------------------+---------------------+
           3 |/////////////////////| current job no. * 2 |  2 FQJOB
             +---------------------+---------------------+
FQSIZM     5 |  MSB of file size   |/////////////////////|  4
             +---------------------+---------------------+
           7 |   project number    | programmer number   |  6 FQPPN
             +---------------------+---------------------+
          11 |         file name in RAD50 format         | 10 FQNAM1
             +                                           +
          13 |              (first 2 words)              | 12
             +---------------------+---------------------+
          15 |     file type in RAD50 format (1 word)    | 14 FQEXT
             +---------------------+---------------------+
          17 |             LSB of file size              | 16 FQSIZ
             +---------------------+---------------------+
          21 |///////////////////////////////////////////| 20
             +---------------------+---------------------+
          23 |              mode parameter               | 22 FQMODE
             +---------------------+---------------------+
          25 | 3rd word of logical name (not translated) | 24 FQFLAG
             +---------------------+---------------------+
FQPROT    27 |   protection code   | =255, explicit prot.| 26 FQPFLG
             +---------------------+---------------------+
          31 |      device name (2 ASCII characters)     | 30 FQDEV
             +---------------------+---------------------+
          33 |  ≠0, unit no. real  |  device unit number | 32 FQDEVN
             +---------------------+---------------------+
          35 |          cluster size parameter           | 34 FQCLUS
             +---------------------+---------------------+
          37 | position parameter (DCN for first block)  | 36 FQNENT
             +---------------------+---------------------+
```

Note

For each of the following field definitions that begin with the word "If," a corresponding statement applies: "If not, the field is left alone." That is, you can insert values in the FIRQB before executing the .FSS to serve as default values for fields when the .FSS returns no result.

FIRQB+FQJOB	The current job number times two.
FIRQB+FQSIZM	If a /FILESIZE:n or /SIZE:n switch was specified, the most significant bits of the file size are contained in this byte.

FIRQB+FQPPN	If a PPN was part of the translated string or if a logical name was found to be the same as a system or user logical name with an associated PPN, this word contains the binary value of that PPN. The project number is in the high byte; the programmer number in the low byte. Any value returned here by .FSS has been verified by the monitor as being within the range for PPNs on a RSTS/E system.
FIRQB+FQNAM1	If a file name was encountered, it is translated to two words of RAD50, beginning at this location. If less than 6 characters, the file name is left-justified and padded with blanks (0 RAD50 characters).
FIRQB+FQEXT	If a file type was encountered, it is translated to one word of RAD50, beginning at this location. If less than 3 characters, the file type is left-justified and padded with blanks (0 RAD50 characters).
FIRQB+FQSIZ	If a /FILESIZE:n or /SIZE:n switch was encountered, the value of n is translated to binary and the LSB of the value are placed in this word. The MSB are placed in the byte at FIRQB+FQSIZM.
FIRQB+FQMODE	If a /MODE:n switch was encountered, the value specified is translated to binary and returned in this word. Bit 15 is set to indicate that a mode switch was translated and to differentiate between a mode of 0 and no mode at all. (Note that bit 15 must be set for a mode value to work on opens.)
FIRQB+FQPFLG	If a file protection code was encountered, a word is returned here. The high byte (FIRQB+FQPROT) is the binary value of the protection code, and the low byte (FIRQB+FQPFLG) is 255. Setting the low byte indicates that a protection code was specified and differentiates a protection code of zero from no protection code at all.
FIRQB+FQDEV	If a device name was specified, it is returned here as two ASCII characters. If less than two characters were specified, the device name is left-justified and padded with spaces (ASCII 32.).
FIRQB+FQDEVN	If a device name but no explicit unit number was specified, this word is zero. If an explicit unit number was specified, then that unit number is in the low byte and 255 is in the high byte. Setting the high byte indicates an explicit device number and differentiates a device number of zero from no device number at all.

Note

> If a syntactically correct logical device name was encountered that could not be translated to a physical device name, then the first six characters of the logical device name is returned as two words of RAD50 starting at offset FQDEV and the next three characters are returned as one word of RAD50 at offset FQFLAG. A status bit in the XRB is set to indicate that this was done.

FIRQB+FQCLUS	If a /CLUSTERSIZE:n switch was encountered, the value of n is returned here, in binary.
FIRQB+FQNENT	If a /POSITION:n switch was encountered, the value of n is returned here, in binary. (The value n is the device cluster number for the first block of the file.)

Errors

BADCNT	The first three words of the XRB are illegal, or an odd or illegal address for nonstandard user logical table.
BADNAM	Some illegal specification occurred in the string. .FSS found a sign bit set on at least one character.
BADSWT	Some .FSS switch was encountered, but it was in an illegal format.
BDNERR	The numeric argument to one of the .FSS switches was illegal.

Example

The following code causes the monitor to scan a string beginning at location BUFFER as a possible file name. BUFFER is defined as an 80-byte area and is filled with zeros to terminate the string scan if what the user typed did not fill the buffer. For example:

```
  BUFFER:     .BLKW0   40.
                .
                .
                .
    (read string into BUFFER from terminal)
                .
                .
                .
                CALL    CLRXRB
                MOV     #80.,XRB+XRLEN          ;DEFINE LENGTH
                MOV     #80.,XRB+XRBC           ;DEFINE LENGTH AGAIN
                MOV     #BUFFER,XRB+XRLOC       ;START OF BUFFER
                .FSS
    (test for error; if none, try open)
```

.LOGS - Check for Logical Device Names

Form

```
.LOGS
```

Function

The .LOGS directive:

- o Translates a system logical device name to a physical device name
- o Verifies that a physical device name is valid
- o Obtains generic information about a particular device

You specify either a logical device name, a physical device designation (name and, if relevant, unit number), or both, in the XRB and the FIRQB. The monitor compares the logical device name specified, if any, against the system-wide logical device names defined by the system manager. (This directive does not check user-specified logical names.) If it finds a match, the monitor returns the device designation associated with the logical device name to the FIRQB. This physical device designation consists of a name, unit number, and in some cases, a PPN.

Next, the monitor checks the physical device designation (either the one passed or the one returned by the monitor in the logical-name translation) to be sure it is valid. If so, information describing the device is returned in the FIRQB.

Privileges Required

None

Data Passed

```
                                XRB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
         +--------------------+--------------------+
       1 |    logical device name in RAD50 format    |  0 XRLEN
         +                 (3 words)                 +
       3 |                                           |  2
         +                                           +
       5 |                                           |  4
         +--------------------+--------------------+
       7 |///////////////////////////////////////////|  6
         +///////////////////////////////////////////+
      11 |///////////////////////////////////////////| 10
         +///////////////////////////////////////////+
      13 |///////////////////////////////////////////| 12
         +///////////////////////////////////////////+
      15 |///////////////////////////////////////////| 14
         +--------------------+--------------------+
```

XRB+XRLEN	The logical device name to be checked is passed as two words in RAD50 format beginning at this location. If only a physical device name check or description is needed, then the first word of the XRB should be passed as zero. The physical device name is passed in the FIRQB.

```
                              FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
         +--------------------+--------------------+
       1 |////////////////////////////////////////////| 0
         +--------------------+--------------------+
       3 |////////////////////////////////////////////| 2
         +--------------------+--------------------+
       5 |////////////////////////////////////////////| 4
         +--------------------+--------------------+
       7 |////////////////////////////////////////////| 6
         +--------------------+--------------------+
      11 |////////////////////////////////////////////| 10
         +--------------------+--------------------+
      13 |////////////////////////////////////////////| 12
         +--------------------+--------------------+
      15 |////////////////////////////////////////////| 14
         +--------------------+--------------------+
      17 |////////////////////////////////////////////| 16
         +--------------------+--------------------+
      21 |////////////////////////////////////////////| 20
         +--------------------+--------------------+
      23 |////////////////////////////////////////////| 22
         +--------------------+--------------------+
      25 |////////////////////////////////////////////| 24
         +--------------------+--------------------+
      27 |////////////////////////////////////////////| 26
         +--------------------+--------------------+
      31 |    physical device name (ASCII format)     | 30 FQDEV
         +--------------------+--------------------+
      33 | ≠0, real dev. no.  | device unit number    | 32 FQDEVN
         +--------------------+--------------------+
      35 |////////////////////////////////////////////| 34
         +--------------------+--------------------+
      37 |////////////////////////////////////////////| 36
         +--------------------+--------------------+
```

FIRQB+FQDEV	The physical device name, as two ASCII characters. A value of 0 (and at offset FIRQB+FQDEVN) indicates the public disk structure (SY:). If you only want a translation from a logical device name to a physical device name, you can pass a value of -1 here. (A -1 is guaranteed not to be a valid physical device name.)
FIRQB+FQDEVN	The unit number of the physical device name is passed in this byte, in binary. To indicate an explicit device number, set the high byte (at FIRQB+FQDEVN+1) to some nonzero value. If the physical device name is of the form "XY:" (that is, no unit number is specified), then set the entire word at this offset to zero to indicate no explicit unit number.

Data Returned

```
                                XRB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////|  0
          +--------------------+--------------------+
        3 |/////////////////////////////////////////|  2
          +--------------------+--------------------+
        5 | logical device flag: -1=N, -2=Y, 0=N.A. |  4 XRLOC
          +--------------------+--------------------+
        7 |           device description            |  6 XRCI
          +--------------------+--------------------+
       11 |     reasonable buffer size for device   | 10 XRBLK
          +--------------------+--------------------+
       13 |/////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |/////////////////////////////////////////| 14
          +--------------------+--------------------+
```

XRB+XRLOC — If there was no logical device name specified or if the logical device name cannot be translated to a physical device name, this word is returned as zero. If the passed logical device name has a PPN associated with it, this word is returned as -2. If not, -1 is returned.

XRB+XRCI — Description of the device. The low byte contains the device's handler index. The high byte contains a set of status flags.

```
                     DDNFS
                  DDRLO   |
               DDWLO   |  |
            FLGPOS  |  |  |
         FLGMOD  |  |  |  |
      FLGFRC  |  |  |  |  |
   FLGKB   |  |  |  |  |  |
FLGRND  |  |  |  |  |  |  |
     |  |  |  |  |  |  |  |
   15 14 13 12 11 10  9  8| 7                  0
  +-----------------------+---------------------+
  |   device-type flags   | device handler index |
  +-----------------------+---------------------+
```

High Byte -- Device-Type Flags

The bits in the high byte of the flag word are set to indicate the type of device specified:

```
FLGRND = 1 The device is random-access.
       = 0 The device is sequential.

 FLGKB = 1 The device is a terminal-type device.
       = 0 The device is not a terminal-type device.

FLGFRC = 1 The device is byte-oriented.  That is, the
           .READ and .WRITE directives handle data in
           byte units.
       = 0 The device is block-oriented.  The .READ and
           .WRITE directives handle data in block
           units.

FLGMOD = 1 The device accepts modifiers in .READ and
           .WRITE directives.
       = 0 The device does not accept modifiers in
           .READ and .WRITE directives.

FLGPOS = 1 The device keeps track of its horizontal
           position and expands characters such as TAB
           to whatever is appropriate for the device.
           You can determine the current horizontal
           position with the .POSTN directive.
       = 0 The device does not keep track of its
           horizontal position.

 DDWLO = 1 The device is a read-only device.
       = 0 The device is not write-locked.

 DDRLO = 1 The device is a write-only device.
       = 0 The device is not read-locked.

 DDNFS = 1 The device is non-file-structured.
       = 0 The device is file-structured.
```

Low Byte -- Device Handler Index

Bits 0-7 of the flags word contain a handler index that indicates the generic kind of device. The currently defined values follow.

Octal Value	Symbol	Meaning
0	DSKHND	All disks
2	TTYHND	All terminals
4	DTAHND	DECtape
6	LPTHND	All line printers
10	PTRHND	Paper tape reader
12	PTPHND	Paper tape punch
14	CDRHND	Card reader
16	MTAHND	Magnetic tape
20	PKBHND	Pseudo keyboards
22	RXDHND	Flexible diskettes
24	RJEHND	2780 remote job entry
26	NULHND	The null device
30	DMCHND	The DMC11/DMR11 DDCMP interface
36		Reserved
40	KMCHND	KMC11
42	IBMHND	IBM interconnect
46	DMPHND	DMP11/DMV11 device

XRB+XRBLK

If the physical device name is valid (either the one returned by the monitor's translation of logical device name or the one passed), this word contains the monitor's "best guess" as a reasonable buffer size for this device. See the .READ and .WRITE directives in this Chapter.

```
                            FIRQB
Mne- Octal                                          Octal   Mne-
monic Offset                                        Offset  monic
          +----------------------+----------------------+
        1 |/////////////////////////////////////////////|  0
          +----------------------+----------------------+
        3 |/////////////////////////////////////////////|  2
          +----------------------+----------------------+
        5 |/////////////////////////////////////////////|  4
          +----------------------+----------------------+
        7 |   project number     |  programmer number   |  6 FQPPN
          +----------------------+----------------------+
       11 |/////////////////////////////////////////////| 10
          +----------------------+----------------------+
       13 |/////////////////////////////////////////////| 12
          +----------------------+----------------------+
       15 |/////////////////////////////////////////////| 14
          +----------------------+----------------------+
       17 |/////////////////////////////////////////////| 16
          +----------------------+----------------------+
       21 |/////////////////////////////////////////////| 20
          +----------------------+----------------------+
       23 |/////////////////////////////////////////////| 22
          +----------------------+----------------------+
       25 |/////////////////////////////////////////////| 24
          +----------------------+----------------------+
       27 |/////////////////////////////////////////////| 26
          +----------------------+----------------------+
       31 |      device name (2 ASCII characters)      | 30 FQDEV
          +----------------------+----------------------+
       33 |  ≠0, unit no. real   |  device unit number  | 32 FQDEVN
          +----------------------+----------------------+
       35 |/////////////////////////////////////////////| 34
          +----------------------+----------------------+
       37 |/////////////////////////////////////////////| 36
          +----------------------+----------------------+
```

FIRQB+FQPPN	If a logical device name was passed and it was translated to a device designation with an associated PPN, the PPN is returned in this word. Otherwise, this word is the same as before the .LOGS call was executed.
FIRQB+FQDEV	The physical device name, either the one returned when a successful translation of logical device name is made or the one passed, if no logical device name was passed. The physical device name is returned as two ASCII characters.
FIRQB+FQDEVN	The physical device unit number, either the one returned when a successful translation of logical device name is made or the one passed, if no logical device name was passed. The low byte contains the unit

number, in binary. The high byte (at FIRQB+FQDEVN+1) is either zero, to indicate no explicit device number, or nonzero, to indicate an explicit device number.

Errors

NODEVC	The physical device name (either the one passed or the one corresponding to the logical device name) is invalid.

Example

The following code asks the monitor to check the name SYSDEV to see if it is a defined system logical device name and, if so, to return the physical device name and characteristics to the XRB and FIRQB:

```
CALL    CLRXRB
MOV     #^RSYS,XRB+XRLEN              ;SET XRB TO TRANSLATE LOGICAL
MOV     #^RDEV,XRB+XRBC               ;DEVICE NAME "SYSDEV"
.LOGS
```

.MESAG -- Message Send/Receive

Form

```
          .
          .
          .
  (Load FIRQB and/or XRB for appropriate subfunction)
          .
          .
          .
  .MESAG
```

Function

The .MESAG directive provides access from a MACRO program to the RSTS/E local message send/receive services and, if your system is a DECnet/E system, to DECnet/E network message send/receive services.

This section contains FIRQB and XRB formats and error descriptions for local message send/receive. (Unless data passed and returned show specific values for the XRB, it should be all zeros.) For detailed information about each call, see the *RSTS/E Programming Manual*. For information about network message send/receive, see *RSTS/E DECnet/E Network Programming in MACRO-11*.

Declare Receiver Subfunction

Privileges Required

SYSIO to declare a receiver with an unrestricted name, network server, or local object. EXQTA to suppress RIB or message quota checks.

Data Passed

```
                            FIRQB
Mne- Octal                                     Octal   Mne-
monic Offset                                    Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////|  0
          +-------------------+-------------------+
        3 |///////////////////////////////////////|  2
          +-------------------+-------------------+
        5 |///////////////////| function code = 1 |  4 FQFIL
          +-------------------+-------------------+
        7 |                                       |  6 FQPPN
          +          receiver name in ASCII       +
       11 |                                       | 10
          +        (space fill to six bytes)      +
       13 |                                       | 12
          +-------------------+-------------------+
       15 |      access       |    object type    | 14 FQEXT
          +-------------------+-------------------+
       17 |            buffer maximum             | 16 FQSIZ
          +-------------------+-------------------+
       21 | inbound link max  |    message max    | 20 FQBUFL
          +-------------------+-------------------+
       23 |            packet maximum             | 22 FQMODE
          +-------------------+-------------------+
       25 |      pkts/msg     | outbound link max | 24 FQFLAG
          +-------------------+-------------------+
       27 |///////////////////////////////////////| 26
          +-------------------+-------------------+
       31 |///////////////////////////////////////| 30
          +-------------------+-------------------+
       33 |///////////////////|     RIB number    | 32 FQDEVN
          +-------------------+-------------------+
       35 |///////////////////////////////////////| 34
          +-------------------+-------------------+
       37 |///////////////////////////////////////| 36
          +-------------------+-------------------+
```

FIRQB+FQPPN	If the caller does not have SYSIO privilege, the fifth and sixth characters of the receiver name must be the caller's job number as two ASCII digits.
FIRQB+FQMODE	Specifies the maximum number of packets that can be queued at any one time. See the *RSTS/E Programming Manual* for more information. Used only in an EMT logging program.
FIRQB+FQFLAG+1	Specifies the number of packets that make up a complete message. See the *RSTS/E Programming Manual* for more information. Used only in an EMT logging program.

Data Returned

Except for a possible error code in byte 0 of the FIRQB, the declare receiver subfunction of .MESAG does not return any meaningful data.

Errors

BADCNT	The specified packet maximum is out of range.
BADFUO	The receiver name, object type, and access parameters passed are inconsistent.
BADNAM	This error can occur for one of the following reasons: o The receiver name passed contains nonprintable characters or leading or embedded spaces o A job without SYSIO privilege passed a nonblank receiver name whose fifth and sixth characters are not its job number o The specified local object type is invalid
ERRERR	The call you attempted requires an optional feature (such as EMT logging or DECnet) that is not available on your system.
FIEXST	The receiver name passed is being used by another receiver, or the local object type you specified is single instance and is already in use.
INUSE	The calling job already exists in the system's list of declared receivers. This error may indicate a logic error in the program or that a previous program running under the same job number failed to remove itself from the receiver list before terminating. In the last case, issue a remove receiver call, and then reissue the declare receiver. (It is common practice to code a remove receiver immediately before the declare receiver call.)
NOBUFS	There were no small buffers available to hold the arguments passed in the declaration. Since the system's use of small buffers is dynamic, a retry may succeed.
PRVIOL	This can occur for one of the following reasons: o The specified RIB number is out of range o The caller does not have the privilege required for a particular option

QUOTA You have exceeded the RIB or message quotas.

Remove Receiver Subfunction

Privileges Required

JOBCTL to remove the RIB of another job.

Data Passed

```
                              FIRQB
Mne-   Octal                                         Octal   Mne-
monic  Offset                                        Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |/////////////////////////////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 |  (See discussion.)  |  function code = 0  |  4 FQFIL
          +---------------------+---------------------+
        7 |/////////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |/////////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |/////////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |/////////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |/////////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |/////////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |/////////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 | ≠0, remove all RIBs |      RIB number     | 32 FQDEVN
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQSIZM The value is two times the job number of the receiver(s) you want to remove or zero if the receiver(s) belong to your own job. To specify a conditional remove, set the sign bit. For example, to conditionally remove receivers for your own job, set this byte equal to 200.

When the sign bit is set, the remove operation is rejected if there are any messages pending for the receiver.

Data Returned

Except for a possible error code in byte 0 of the FIRQB, the remove receiver subfunction of .MESAG does not return any meaningful data.

Errors

BADFUO	The argument at FIRQB+FQSIZM was odd. It must be zero to remove the calling program or job number times two to remove another job.
INUSE	Occurs on a conditional remove if there are messages pending for the receiver.
PRVIOL	The caller does not have JOBCTL privilege and has attempted to remove the receiver ID block (RIB) of another job (that is, FIRQB+FQSIZM is nonzero).

Send Local Data Message Subfunction

Privileges Required

SEND to send to a restricted receiver.

Data Passed

```
                             FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
           +---------------------+---------------------+
         1 |///////////////////////////////////////////|  0
           +---------------------+---------------------+
         3 |///////////////////////////////////////////|  2
           +---------------------+---------------------+
FQSIZM   5 |     destination     | function code = -1  |  4 FQFIL
           +---------------------+---------------------+
         7 |                                           |  6 FQPPN
           +          receiver name in ASCII           +
        11 |                                           | 10
           +        (space fill to six bytes)          +
        13 |                                           | 12
           +---------------------+---------------------+
        15 |                                           | 14 FQEXT
           +                                           +
        17 |     optional user parameter string --     | 16
           +      up to 20 bytes of additional         +
        21 |     user data can be specified here       | 20
           +                                           +
        23 |          (zero fill to 20 bytes)          | 22
           +                                           +
        25 |                                           | 24
           +                                           +
        27 |                                           | 26
           +                                           +
        31 |                                           | 30
           +                                           +
        33 |                                           | 32
           +                                           +
        35 |                                           | 34
           +                                           +
        37 |                                           | 36
           +---------------------+---------------------+
```

FIRQB+FQSIZM   You can specify the destination in one of three ways:

- o A zero value indicates that the destination is the receiver name that starts at FIRQB+FQPPN.

- o A value equal to the job number times two indicates that the destination is this job number. If you do not specify a name (FIRQB+FQPPN = 0), the system locates the receiver by job number; this works only when the the receiving job is receiving messages on RIB 0.

If you specify a name, the system sends to the specified receiver name. However, the system also checks that the receiver's job number matches the value in FIRQB+FQSIZM.

- o A value equal to the local object type (LOT) plus 200 indicates that the destination is the receiver for the specified local object type number. Only single-instance object types are valid (see the *RSTS/E Programming Manual*). Legal values are:

Lot	Receiver
1	Error logger
2	EMT logger
3	PBS -- command interpreter
4	PBS -- server interpreter
5	PBS -- user request packet
6	OPSER

FIRQB+FQPPN Specifies the receiver name. If FIRQB+FQSIZM is zero, this field must be nonzero.

XRB

```
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
         +--------------------+--------------------+
       1 |  length of output buffer, in bytes, 0-512 |  0 XRLEN
         +--------------------+--------------------+
       3 |number of bytes to send, 0 to buffer length|  2 XRBC
         +--------------------+--------------------+
       5 |        starting address of buffer         |  4 XRLOC
         +--------------------+--------------------+
       7 |///////////////////////////////////////////|  6
         +--------------------+--------------------+
      11 |///////////////////////////////////////////| 10
         +--------------------+--------------------+
      13 |///////////////////////////////////////////| 12
         +--------------------+--------------------+
      15 |///////////////////////////////////////////| 14
         +--------------------+--------------------+
```

XRB+XRLEN Length of the output buffer, in bytes. This value can range from zero through 512.

XRB+XRBC The number of bytes to be sent. This value can range from zero through the size of the buffer, as specified at XRB+XRLEN.

XRB+XRLOC Starting address of the output buffer. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)

Data Returned

```
                               FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////|  0
          +--------------------+--------------------+
        3 |/////////////////////////////////////////|  2
          +--------------------+--------------------+
FQSIZM  5 |job no.*2 of rec. job|///////////////////|  4
          +--------------------+--------------------+
        7 |/////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |/////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |/////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |/////////////////////////////////////////| 14
          +--------------------+--------------------+
       17 |/////////////////////////////////////////| 16
          +--------------------+--------------------+
       21 |/////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |/////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |/////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |/////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |/////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |/////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |/////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |/////////////////////////////////////////| 36
          +--------------------+--------------------+
```

Errors

BADCNT	The XRB+XRLEN value is illegal. It can range from 0 through 512.
BADFUO	The value at FIRQB+FQSIZM is odd. It must be zero or the receiver's job number times two.
NOBUFS	System buffers are currently not available to store this message for the intended local receiver. A later retry may proceed without error.
NOROOM	The number of pending messages for the intended local receiver is at its declared maximum, or the receiving program is hibernating. This program should try again later. If this error occurs repeatedly, the receiver is not processing messages often enough.
NOSUCH	The intended local receiver could not be located in the system's list of declared receivers. The receiver must be declared (with a declare receiver) before any data can be transmitted to it.
PRVIOL	Some access violation has occurred. Either the receiver does not allow any local senders, or the sender does not have SEND privilege and the receiver allows only restricted senders.

Receive Subfunction

Privileges Required

None

Data Passed

```
                                   FIRQB
Mne-   Octal                                               Octal   Mne-
monic  Offset                                              Offset  monic
             +-------------------+---------------------+
          1  |/////////////////////////////////////////|  0
             +-------------------+---------------------+
          3  |/////////////////////////////////////////|  2
             +-------------------+---------------------+
FQSIZM    5  |  receive modifier  | function code = 2  |  4 FQFIL
             +-------------------+---------------------+
          7  | qualifier (norm = 0)|  sender select    |  6 FQPPN
             +-------------------+---------------------+
         11  |/////////////////////////////////////////| 10
             +-------------------+---------------------+
         13  |/////////////////////////////////////////| 12
             +-------------------+---------------------+
         15  |/////////////////////////////////////////| 14
             +-------------------+---------------------+
         17  |/////////////////////////////////////////| 16
             +-------------------+---------------------+
         21  |/////////////////////////////////////////| 20
             +-------------------+---------------------+
         23  |         sleep time, in seconds          | 22 FQMODE
             +-------------------+---------------------+
         25  |/////////////////////////////////////////| 24
             +-------------------+---------------------+
         27  |/////////////////////////////////////////| 26
             +-------------------+---------------------+
         31  |/////////////////////////////////////////| 30
             +-------------------+---------------------+
         33  |///////////////////|     RIB number      | 32 FQDEVN
             +-------------------+---------------------+
         35  |/////////////////////////////////////////| 34
             +-------------------+---------------------+
         37  |/////////////////////////////////////////| 36
             +-------------------+---------------------+
                              XRB
Mne- Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
          +--------------------+--------------------+
        1 |       buffer size in bytes, or 0        |  0 XRLEN
          +--------------------+--------------------+
        3 |               must be 0                 |  2 XRBC
          +--------------------+--------------------+
        5 |       starting address of buffer        |  4 XRLOC
          +--------------------+--------------------+
        7 |/////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |/////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |/////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |/////////////////////////////////////////| 14
          +--------------------+--------------------+
```

XRB+XRLEN	The size of the receive buffer, in bytes. This word can be zero if no user data is desired on the receive. The amount of data transferred from a pending message will never be greater than the buffer size.
XRB+XRBC	This word must be passed as zero. The monitor returns the actual number of bytes of user data transferred in this word location, as shown in the Data Returned sections.
XRB+XRLOC	The starting address of the receive buffer. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)

Data Returned

The Receive call returns data to the FIRQB and XRB, identifying the type of message received and user data, if any, to the buffer defined in the data passed.

The FIRQB and XRB formats for the local data message follow.

Data Returned (Local Data Message)

```
                                  FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                               Offset  monic
             +---------------------+---------------------+
         1   |///////////////////////////////////////////|  0
             +---------------------+---------------------+
         3   |///////////////////////////////////////////|  2
             +---------------------+---------------------+
FQSIZM   5   |   job number * 2    |  function code = -1 |  4 FQFIL
             +---------------------+---------------------+
         7   |   project number    |  programmer number  |  6 FQPPN
             +---------------------+---------------------+
        11   |/////////////////////|sender KB no. or 377 | 10 FQNAM1
             +---------------------+---------------------+
        13   |remainder (number of bytes not transferred)| 12
             +---------------------+---------------------+
        15   |                                           | 14 FQEXT
             +                                           +
        17   |      Data passed as parameters by the     | 16
             +          sender of this message.          +
        21   |                                           | 20
             +                                           +
        23   |                                           | 22
             +                                           +
        25   |                                           | 24
             +                                           +
        27   |                                           | 26
             +                                           +
        31   |                                           | 30
             +                                           +
        33   |                                           | 32
             +                                           +
        35   |                                           | 34
             +                                           +
        37   |                                           | 36
             +---------------------+---------------------+
```

FIRQB+FQNAM1	A value of 377 means the sender is detached.
FIRQB+FQEXT	For an EMT logger message, the monitor returns three values:

- o Bytes 14-15 contain the number of data packets not transferred.
- o Bytes 16-17 contain the number of EMTs your program missed, either because it is not processing data packets quickly enough, or because not enough XBUF is available to store all the data packets that the monitor is creating.

o Bytes 20-21 contain the number of data packets transferred.

See the *RSTS/E Programming Manual* for more information.

```
                               XRB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
           +--------------------+--------------------+
        1  |////////////////////////////////////////|  0
           +--------------------+--------------------+
        3  |length (actual number of bytes transferred)|  2 XRBC
           +--------------------+--------------------+
        5  |////////////////////////////////////////|  4
           +--------------------+--------------------+
        7  |////////////////////////////////////////|  6
           +--------------------+--------------------+
       11  |////////////////////////////////////////| 10
           +--------------------+--------------------+
       13  |////////////////////////////////////////| 12
           +--------------------+--------------------+
       15  |////////////////////////////////////////| 14
           +--------------------+--------------------+
```

Errors

BADCNT	The buffer descriptor in the first three words of the XRB is invalid.
BADFUO	Not a declared receiver. Before any receive can succeed, you must execute a declare receiver call to define the RIB number you want to use.
NOSUCH	For a receive without sleep (bit 0 in receive modifier = 0), this error indicates that no appropriate messages are pending. For a receive with sleep (bit 0 in receive modifier = 1), this error is returned when the program is awakened from the sleep. The program must execute another receive call to retrieve any pending messages.

Send Privileges Subfunction

This subfunction provides a method for a program to tell another program about a job's current privileges and guarantees that the data cannot be falsified. Data can be sent at the same time.

Privileges Required

SEND to send to a restricted receiver.

Data Passed - Send Privileges

```
                              FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////| 0
          +-------------------+-------------------+
        3 |///////////////////////////////////////| 2
          +-------------------+-------------------+
FQSIZM  5 |    destination    | function code = -11 | 4 FQFIL
          +-------------------+-------------------+
        7 |                                       | 6 FQPPN
          +        receiver name in ASCII         +
       11 |                                       | 10
          +      (space fill to six bytes)        +
       13 |                                       | 12
          +-------------------+-------------------+
       15 |///////////////////////////////////////| 14 FQEXT
          +///////////////////////////////////////+
       17 |///////////////////////////////////////| 16
          +///////////////////////////////////////+
       21 |///////////////////////////////////////| 20
          +///////////////////////////////////////+
       23 |///////////////////////////////////////| 22
          +---------------------------------------+
       25 |                                       | 24
          +     optional user parameter string    +
       27 |                                       | 26
          +      (up to 12 bytes of additional    +
       31 |      user data can be specified here, | 30
          +          zero fill to 12 bytes)       +
       33 |                                       | 32
          +                                       +
       35 |                                       | 34
          +                                       +
       37 |                                       | 36
          +-------------------+-------------------+
```

FIRQB+FQSIZM You can specify the destination in one of three ways:

- o A zero value indicates that the destination is the receiver name that starts at FIRQB+FQPPN.
- o A value equal to the job number times two indicates that the destination is this job number. If you do not specify a name (FIRQB+FQPPN = 0), the system locates the receiver by job number; this works only when the the receiving job is receiving messages on RIB 0.

If you specify a name, the system sends to the specified receiver name. However, the system also checks that the receiver's job number matches the value in FIRQB+FQSIZM.

- A value equal to the local object type (LOT) plus 200 indicates that the destination is the receiver for the specified local object type number. Only single-instance object types are valid (see the *RSTS/E Programming Manual*). Legal values are:

Lot	Receiver
1	Error logger
2	EMT logger
3	PBS -- command interpreter
4	PBS -- server interpreter
5	PBS -- user request packet
6	OPSER

FIRQB+FQPPN   Specifies the receiver name. If FIRQB+FQSIZM is zero, this field must be nonzero.

XRB

Mne-monic	Octal Offset		Octal Offset	Mne-monic
	1	length of output buffer, in bytes, 0-512	0	XRLEN
	3	number of bytes to send, 0 to buffer length	2	XRBC
	5	starting address of buffer	4	XRLOC
	7	//	6	
	11	//	10	
	13	//	12	
	15	//	14	

XRB+XRLEN   Length of the output buffer, in bytes. This value can range from zero through 512.

XRB+XRBC   The number of bytes to be sent. This value can range from one through the size of the buffer, as specified at XRB+XRLEN. If the value in XRB+XRLEN is zero, this value must be zero.

.NAME

.NAME - Set Program Name

Form

```
.NAME
```

Function

The .NAME directive sets the program name in the monitor tables. The monitor enters the name in an internal table; otherwise, it makes no use of the program name. However, the SHOW USERS command uses the names in listing current information for jobs (under the "What" column) on the system. The BASIC-PLUS run-time system uses this directive when the user issues an OLD, NEW, or RENAME command.

The program name is passed as two words of RAD50 data in the FIRQB. Note that the data is passed in the same location in the FIRQB where the file name exists at the P.RUN entry point into a run-time system. If you are coding or modifying a run-time system, one of the first things to do on entry at P.RUN is to set the program's name. Thus, the file name's position in the FIRQB at this point is convenient for use.

Privileges Required

None

Data Passed

```
                               FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |       program name in RAD50 format        | 10 FQNAM1
          +                (2 words)                  +
       13 |                                           | 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQNAM1   The program name to be set; two words in RAD50 format.

Data Returned

The .NAME directive does not return any meaningful data.

Errors

No errors are possible with the .NAME directive.

Example

The following code sets the name PROGRM in the monitor tables:

```
  MOV     #^RPRO,FIRQB+FQNAM1              ; Set FIRQB to declare
  MOV     #^RGRM,FIRQB+FQNAM1+2            ; Name of "PROGRM"
  .NAME
```

.PEEK - Look at Monitor Memory

Form

.PEEK

Function

The .PEEK directive returns the contents of one word of the monitor's memory; that is, the memory mapped by the kernel mode APRs (see Chapter 2). Only a job with the required privileges can execute the .PEEK directive. (However, you should not base any of your code on the contents of monitor memory because DIGITAL reserves the right to change the monitor structure and internal addresses at any time, except for the addresses listed in Table 3-3.)

Privileges Required

RDMEM to read kernel memory other than the I/O page and WRTMEM to read the I/O page.

Data Passed

```
                                XRB
Mne-  Octal                                         Octal   Mne-
monic Offset                                         Offset  monic
         +--------------------+--------------------+
       1 |  virtual address of desired monitor word  |  0 XRLEN
         +--------------------+--------------------+
       3 |/////////////////////////////////////////////|  2
         +--------------------+--------------------+
       5 |/////////////////////////////////////////////|  4
         +--------------------+--------------------+
       7 |/////////////////////////////////////////////|  6
         +--------------------+--------------------+
      11 |/////////////////////////////////////////////| 10
         +--------------------+--------------------+
      13 |/////////////////////////////////////////////| 12
         +--------------------+--------------------+
      15 |/////////////////////////////////////////////| 14
         +--------------------+--------------------+
```

XRB+XRLEN	The virtual address of the data word in monitor memory whose contents are to be returned. The value must be even, since word addresses on the PDP-11 are always even. Peeking at data in the I/O page (kernel APR 7, or 111 (binary) in bits 15, 14, and 13) can cause unpredictable system results; DIGITAL does not recommend this. Furthermore, using .PEEK to obtain data in APRs 5 or 6 returns data in the file processor's private buffer pool.

Use .PEEK to examine addresses returned by get monitor tables calls or addresses of fixed monitor locations.

Table 3-3 shows fixed monitor locations and their addresses.

Table 3-3: Fixed Monitor Locations

Address (Octal)	Name	Meaning
44(word)	IDATE	The date when the system was last started by START.
46(word)	ITIME	The time of day when the system was last started by START.
1000(word)	DATE	Current system date.
1002(word)	TIME	Current time of day.
1006(byte)	JOB	Job number times 2 of the job currently running (always the user's own job number.)
1010(word)	JOBDA	Address of the job data block (JDB) of the currently running job (always the user's own JDB).
1012(word)	JOBF	Address of the JDFLG word in the JDB of the currently running job (always the user's own JDB).
1014(word)	IOSTS	Address of the JDIOST (low) byte and JDPOST (high) byte in the JDB of the currently running job (always the user's own JDB).

Data Returned

```
                                XRB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
          +---------------------+---------------------+
        1 |    contents of the monitor memory word    |  0 XRLEN
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
```

XRB+XRLEN	The contents of the requested monitor memory location.

Errors

B.4	The address specified caused a trap to the kernel mode vector at 4 (UNIBUS timeout, odd address, and so forth).
B.250	The address specified caused a memory management unit violation (trap to the kernel mode vector at 250).
PRVIOL	The caller does not have the required privilege.

Example

The following code obtains the contents of monitor memory location 1006 (the low byte of which is the current job number times two):

```
  MOV    #JOB,XRB+XRLEN      ; Set address to that of "JOB"
  .PEEK
```

.PLAS -- Access Resident Library

The .PLAS (Program Logical Address Space) directive has six subfunctions that allow a MACRO program to access a resident library. Resident libraries must be so defined by the system manager with the INSTALL LIBRARY command (see the *RSTS/E System Manager's Guide*). As noted in Chapter 2, the easiest way to do this is to link the resident library to your program using TKB, the Task Builder that links modules assembled or compiled under the RSX run-time system or its derivatives. However, you can use .PLAS subfunctions to directly access resident libraries.

Table 3-4 lists the .PLAS subfunctions by function code. The following subsections describe the subfunctions in alphabetical order.

Table 3-4: Summary of .PLAS Subfunctions

FQFUN Value (Octal)	Mnemonic	Action Performed
0	ATRFQ	Attach resident library. Attaches the job to a resident library; necessary before the job can map a window to the library.
2	DTRFQ	Detach resident library. Detach the job from a resident library.
4	CRAFQ	Create address window. Defines a range of virtual addresses to be a window for looking at all or some portion of a resident library. Optionally, CRAFQ maps the window to all or some portion of a resident library. (The mapping can be done separately with MAPFQ.) The CRAFQ subfunction reserves one or more APRs, so CRAFQ takes space in the job area even though the window may not actually be mapped.
6	ELAFQ	Eliminate address window. Releases the APRs used by a particular window.
10	MAPFQ	Map window. Map an already created address window of virtual addresses to actual memory locations in an attached resident library. The monitor loads the library from disk if necessary.
12	UMPFQ	Unmap address window. Releases a window of virtual addresses from a mapping to actual memory locations.

When a program exits or a user logs out, the monitor automatically detaches all libraries and unmaps and eliminates all windows for the job.

Privileges Required

Depends on the subfunction; refer to the individual descriptions that follow this introductory information on the .PLAS directive.

ATRFQ (Attach Resident Library)

Form

```
  MOVB   #ATRFQ,FIRQB+FQFIL
            .
            .
            .
  (set appropriate parameters)
            .
            .
            .
  .PLAS
```

Function

The ATRFQ subfunction of .PLAS declares your intent to access a resident library. The type of access is specified in the call. The number of resident libraries that can be attached to a job at any given time is unlimited (subject to the availability of system small buffers).

The job's ability to access the resident library depends upon the protection assigned to the library by the system manager when the library was installed. The default protection grants read access to all users and denies write access to all users.

If the calling job can access the library in the specified fashion, the monitor sets up its own internal tables, which lay the groundwork for the job to map windows to the library. Note, however, that the resident library does not take up space in the job area (virtual memory) with an attach. APRs are assigned (virtual memory in the job area is taken) when a window is created (using CRAFQ).

Privileges Required

Read-only or read/write access to the library. The protection code is interpreted exactly like a file protection code; as with files, access is granted either on the basis of the protection code or because of GREAD, GWRITE, WREAD, or WWRITE privileges.

Data Passed

```
                            FIRQB
Mne-  Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
          +--------------------+--------------------+
        1 |//////////////////////////////////////////| 0
          +--------------------+--------------------+
        3 |//////////////////////////////////////////| 2
          +--------------------+--------------------+
        5 |////////////////////|    ATRFQ (= 0)     | 4 FQFIL
          +--------------------+--------------------+
        7 |//////////////////////////////////////////| 6
          +--------------------+--------------------+
       11 |//////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |  resident library name in RAD50 format  | 12
          +               (2 words)                 +
       15 |                                         | 14
          +--------------------+--------------------+
       17 |//////////////////////////////////////////| 16
          +--------------------+--------------------+
       21 |//////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |               access mode               | 22 FQMODE
          +--------------------+--------------------+
       25 |//////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |//////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |//////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |//////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |//////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |//////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQFIL	The function code ATRFQ (octal value = 0).
FIRQB+12	The name of the resident library to which the job is to be attached; two words in RAD50 format. Resident libraries are made known to the monitor by the system manager with the INSTALL/LIBRARY command (see the *RSTS/E System Manager's Guide*). With this command, the system manager defines a file (filename.LIB) as a resident library. The monitor regards "filename" as the resident library's name.

FIRQB+FQMODE The low-order two bits of this word define the way the job wants to access the library:

Bit 0 = 1 Read-only access is desired.

Bit 1 = 1 Read/write access is desired.

Data Returned

```
                              FIRQB
Mne-  Octal                                         Octal   Mne-
monic Offset                                        Offset  monic
         +------------------+------------------+
       1 |//////////////////////////////////////|  0
         +------------------+------------------+
       3 |//////////////////| current job no. * 2 |  2 FQJOB
         +------------------+------------------+
       5 |//////////////////////////////////////|  4
         +------------------+------------------+
       7 |     resident library identification  |  6 FQPPN
         +------------------+------------------+
      11 | size, in 32-word blocks, of the library | 10 FQNAM1
         +------------------+------------------+
      13 |//////////////////////////////////////| 12
         +------------------+------------------+
      15 |//////////////////////////////////////| 14
         +------------------+------------------+
      17 |//////////////////////////////////////| 16
         +------------------+------------------+
      21 |//////////////////////////////////////| 20
         +------------------+------------------+
      23 |//////////////////////////////////////| 22
         +------------------+------------------+
      25 |//////////////////////////////////////| 24
         +------------------+------------------+
      27 |//////////////////////////////////////| 26
         +------------------+------------------+
      31 |//////////////////////////////////////| 30
         +------------------+------------------+
      33 |//////////////////////////////////////| 32
         +------------------+------------------+
      35 |//////////////////////////////////////| 34
         +------------------+------------------+
      37 |//////////////////////////////////////| 36
         +------------------+------------------+
```

FIRQB+FQJOB	The current job number times two.
FIRQB+FQPPN	This word is an identifier that must be used, rather than the resident library name, in subsequent calls to identify a resident library. Thus, you use this identifier to detach the job from the library (DTRFQ) and map and unmap windows to the library (MAPFQ and UMPFQ).
FIRQB+FQNAM1	The size of the resident library, in 32-word blocks.

Errors

NOSUCH — The resident library specified in the data passed is not known to the monitor. The system manager must install a resident library before it can be used.

PRVIOL — The attach did not succeed because the caller's privilege did not allow the access specified in the data passed. This could happen either:

- o Because the access code specified in the data passed is not compatible with the possible access defined when the library was installed by the system manager.
- o Because the protection code associated with the resident library file excludes access by the user.

Example

The following code attaches the job to a resident library called DATBAS. The access desired is defined as read/write:

```
MOVB    #ATRFQ,FIRQB+FQFIL           ;DEFINE FUNCTION CODE
MOV     #^RDAT,FIRQB+12              ;LIBRARY NAME IS
MOV     #^RBAS,FIRQB+FQEXT           ;DEFINED AS "DATBAS"
MOV     #2,FIRQB+FQMODE              ;ACCESS=READWRITE
.PLAS
```

CRAFQ (Create Address Window)

Form

```
  MOVB   #CRAFQ,FIRQB+FQFIL
           .
           .
           .
  (set up parameters)
           .
           .
           .
  .PLAS
```

Function

The CRAFQ subfunction of .PLAS can be used either to create a window (a range of virtual addresses) or to create a window of virtual addresses and map it to a range of actual addresses in an attached library. You define the range of addresses by:

- o Naming a base APR (which defines the starting address of the window)
- o Specifying the size of the window in 32-word blocks

Thus, a window always begins on a 4K-word boundary in virtual memory and always takes at least 4K words. It may take more than 4K words, depending on the size of the window.

If the address range overlaps the user job image, the call fails with an error. The address range cannot overlap the run-time system (high segment). However, if the job is currently running under RSX emulation, this is not a consideration. APR 7, normally used to map a run-time system, can be used instead to map a window to a resident library. If the address range overlaps an existing window, the previously created window is eliminated.

The difference between creating a window and creating and mapping a window is best shown by example. By using create without map, you can define one window, which can be mapped to a library or portion of a library and then remapped to another portion of the same library or another library, as many times as desired, using the MAPFQ subfunction of .PLAS.

For example, suppose your program takes up 24K words and you want to access a 24K-word resident library of data values. You can use create without map to set up a 4K-word window in APR 6. You can then map the window (using MAPFQ) to the first 4K words of the library, process the data, map to the next 4K words of the library, and so forth.

If, on the other hand, you had a 4K program and still wanted to access a 24K-word library, you could use CRAFQ to create a 24K-word window and map it to the entire library in APR 1 to 6.

A job can create a maximum of seven windows. A window takes at least one APR (it may take more, depending on the size you specify for the window). Thus, the maximum of seven assumes seven windows in APR 1 to 7. APR 0 can never be used to create a window, since the user program takes at least this much space. As mentioned previously, a window cannot overlap the user job image; thus, the size of the user job image determines the lowest base APR that can be used. If the program (user job image) is less than 4K words, APR 1 and up (to the limit imposed by the run-time system boundary) can be used to create windows. If the user job image is between 4K words and 8K words, APR 2 and up can be used to create windows, and so forth.

If a window is created that overlaps an already existing window, the old window is eliminated. For example, if you create a 6K-word window using a base APR of 5, the window uses APR 5 and APR 6. If you then create a 4K-word window using a base APR of 6, the entire old window is eliminated. APR 5 is then free for other use; APR 6 is used for the new window.

Privileges Required

RDMEM to map physical memory other than I/O page read-only. SYSMOD to map physical memory read/write or to map the I/O page read/only.

Data Passed

```
                              FIRQB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
          +----------------------+----------------------+
        1 |/////////////////////////////////////////////|  0
          +----------------------+----------------------+
        3 |/////////////////////////////////////////////|  2
          +----------------------+----------------------+
        5 |//////////////////////|     CRAFQ (= 4)      |  4 FQFIL
          +----------------------+----------------------+
        7 |   base APR (1-7)     |//////////////////////|  6
          +----------------------+----------------------+
       11 |/////////////////////////////////////////////| 10
          +----------------------+----------------------+
       13 |     size of windows, in 32-word blocks      | 12
          +----------------------+----------------------+
       15 |   library identification (for map only)     | 14 FQEXT
          +----------------------+----------------------+
       17 |  offset, in 32-word blocks (for map only)   | 16 FQSIZ
          +----------------------+----------------------+
       21 |  length, in 32-word blocks (for map only)   | 20 FQBUFL
          +----------------------+----------------------+
       23 |                 access flags                | 22 FQMODE
          +----------------------+----------------------+
       25 |/////////////////////////////////////////////| 24
          +----------------------+----------------------+
       27 |/////////////////////////////////////////////| 26
          +----------------------+----------------------+
       31 |/////////////////////////////////////////////| 30
          +----------------------+----------------------+
       33 |/////////////////////////////////////////////| 32
          +----------------------+----------------------+
       35 |/////////////////////////////////////////////| 34
          +----------------------+----------------------+
       37 |/////////////////////////////////////////////| 36
          +----------------------+----------------------+
```

FIRQB+FQFIL	The function code CRAFQ (octal value = 4).
FIRQB+7	The base APR of the window, 1 to 7. Implicitly defines the starting virtual address of the window. This byte cannot be zero; in addition, it cannot name an APR already being used to map the user job image.
FIRQB+12	The desired size of the window, in 32-word blocks. For example, a value of 128. equals 4K words.

FIRQB+FQEXT	The identifier of the resident library to which the window is to be mapped. (This is the value returned by the ATRFQ function of .PLAS at FIRQB+FQPPN.) This word is ignored for calls requesting a create without mapping (bit 7 at FIRQB+FQMODE equals 0). You can also map the window to physical memory by passing a value of -4. See MAPFQ for details.
FIRQB+FQSIZ	The offset, in 32-word blocks, from the start of the library where the mapping is to begin. This word is ignored if no mapping is requested (bit 7 at FIRQB+FQMODE equals 0). A value of zero for this word indicates the window is to be mapped beginning at the first byte of the library. A value of one indicates the window is to be mapped beginning at the 33rd word of the library (starting address + 64), and so forth.
FIRQB+FQBUFL	The length, in 32-word blocks, of the area to be mapped (ignored if bit 7 at FIRQB+FQMODE equals 0). This value cannot be greater than the size of the window specified at FIRQB+12. Furthermore, this value, combined with the offset value at FIRQB+FQSIZ, cannot indicate an address beyond the end of the library or into the high segment (run-time system). A value of zero for this word defaults to either the size of the window (specified at FIRQB+FQEXT) or the space remaining in the library, whichever is smaller.
FIRQB+FQMODE	Two bits in this word define whether the window is to be mapped and whether write access to the window is desired:

bit 1 = 1	Write access to the window is desired.
= 0	No write access to the window is desired.
bit 7 = 1	The window is to be mapped.
= 0	The window is not to be mapped.

The octal value to set bit 7 is 200; the value to set bit 1 is 2. Thus, an octal value of 202 for this word requests mapping and write access. A separate setting for write access in CRAFQ and in ATRFQ allows you to attach to a library read/write and map a portion of the library read-only.

Data Returned

```
                              FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +--------------------+--------------------+
        1 |//////////////////////////////////////////|  0
          +--------------------+--------------------+
        3 |////////////////////| current job no. * 2 |  2 FQJOB
          +--------------------+--------------------+
        5 |//////////////////////////////////////////|  4
          +--------------------+--------------------+
        7 |////////////////////|      window ID      |  6 FQPPN
          +--------------------+--------------------+
       11 |   starting virtual address of new window |  10 FQNAM1
          +--------------------+--------------------+
       13 |//////////////////////////////////////////|  12
          +--------------------+--------------------+
       15 |//////////////////////////////////////////|  14
          +--------------------+--------------------+
       17 |//////////////////////////////////////////|  16
          +--------------------+--------------------+
       21 |      mapped length, in 32-word blocks     |  20 FQBUFL
          +--------------------+--------------------+
       23 |               status flags                |  22 FQMODE
          +--------------------+--------------------+
       25 |//////////////////////////////////////////|  24
          +--------------------+--------------------+
       27 |//////////////////////////////////////////|  26
          +--------------------+--------------------+
       31 |//////////////////////////////////////////|  30
          +--------------------+--------------------+
       33 |//////////////////////////////////////////|  32
          +--------------------+--------------------+
       35 |//////////////////////////////////////////|  34
          +--------------------+--------------------+
       37 |//////////////////////////////////////////|  36
          +--------------------+--------------------+
```

FIRQB+FQJOB	The current job number times two.
FIRQB+FQPPN	The window ID. Use in later MAPFQ calls to map the newly created window; it must be used in any ELAFQ calls to eliminate the newly created windows. The value returned may be from 1 to 7.
FIRQB+FQNAM1	The starting virtual address of the new window.
FIRQB+FQBUFL	Length actually mapped by the window, in 32-word blocks.

FIRQB+FQMODE	Status flags. The monitor returns the status of the new window as follows:

bit 15 = 1	Window was created successfully.
= 0	Window was not created successfully.
bit 14 = 1	An existing window was unmapped because it overlapped the newly created mapping.
= 0	No existing windows were unmapped by this mapping.
bit 13 = 1	An existing window was eliminated because it overlapped the newly created window.
= 0	No existing windows were eliminated by this create.

Errors

BADFUO	Either the base APR and window length specified were invalid, or the offset and mapping length values specified were invalid. (For example, an offset indicating a starting address for the mapping that is beyond the end of the library or into the run-time system is invalid.)
NOBUFS	Creating a window requires a small buffer; a small buffer is not currently available.
NOROOM	You attempted to create more than seven address windows.
NOSUCH	The library ID specified for mapping is not a library currently attached to the job.
PRVIOL	The create was unsuccessful because the user privileges do not allow the access desired. At this point, since the library has been attached successfully with some access defined, this error means that the access requested in the CRAFQ is not allowed by the access requested in the ATRFQ.

Example

The following code creates a 4K-word address window and maps it to the beginning of a library whose ID (returned from a previous ATRFQ) has been stored at location LIBID:

```
  MOVB    #CRAFQ,FIRQB+FQFIL            ;DEFINE FUNCTION CODE
  MOV     #6,FIRQB+7                    ;BASE APR = 6
  MOV     #200,FIRQB+12                 ;WINDOW = 4K WORDS
  MOV     LIBID,FIRQB+FQEXT             ;SET LIBRARY ID
  CLR     FIRQB+FQSIZ                   ;OFFSET = 0
  CLR     FIRQB+FQBUFL                  ;MAP 4K WORDS OR TO
                                        ;END OF LIBRARY
  MOV     #200,FIRQB+FQMODE             ;MAP WINDOW, READ-ONLY
  .PLAS
```

DTRFQ (Detach Resident Library)

Form

```
  MOVB    #DTRFQ,FIRQB+FQFIL
          .
          .
          .
  (define library to be detached)
          .
          .
          .
  .PLAS
```

Function

The DTRFQ subfunction of .PLAS detaches a previously attached resident library. Any windows mapped to the library by the calling job are unmapped. If no other jobs are currently attached to the library and it was installed with the /UNLOAD option, the monitor removes the library from memory.

If this is a dynamic region and there are no other jobs currently attached to it, and the region is marked for deletion after the last job detaches, the monitor removes the region from memory and deletes it from the resident library list.

Privileges Required

None

Data Passed

```
                              FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////|  0
          +-------------------+-------------------+
        3 |///////////////////////////////////////|  2
          +-------------------+-------------------+
        5 |///////////////////|    DTRFQ (= 2)    |  4 FQFIL
          +-------------------+-------------------+
        7 |library identification (returned by ATRFQ) |  6 FQPPN
          +-------------------+-------------------+
       11 |///////////////////////////////////////| 10
          +-------------------+-------------------+
       13 |///////////////////////////////////////| 12
          +-------------------+-------------------+
       15 |///////////////////////////////////////| 14
          +-------------------+-------------------+
       17 |///////////////////////////////////////| 16
          +-------------------+-------------------+
       21 |///////////////////////////////////////| 20
          +-------------------+-------------------+
       23 |///////////////////////////////////////| 22
          +-------------------+-------------------+
       25 |///////////////////////////////////////| 24
          +-------------------+-------------------+
       27 |///////////////////////////////////////| 26
          +-------------------+-------------------+
       31 |///////////////////////////////////////| 30
          +-------------------+-------------------+
       33 |///////////////////////////////////////| 32
          +-------------------+-------------------+
       35 |///////////////////////////////////////| 34
          +-------------------+-------------------+
       37 |///////////////////////////////////////| 36
          +-------------------+-------------------+
```

FIRQB+FQFIL	The function code DTRFQ (octal value = 2).
FIRQB+FQPPN	The library identification returned at FIRQB+FQPPN by the ATRFQ that attached the job to the library.

Data Returned

```
                             FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                           Offset  monic
           +---------------------+---------------------+
         1 |/////////////////////////////////////////////|  0
           +---------------------+---------------------+
         3 |//////////////////////| current job no. * 2 |  2 FQJOB
           +---------------------+---------------------+
         5 |/////////////////////////////////////////////|  4
           +---------------------+---------------------+
         7 |/////////////////////////////////////////////|  6
           +---------------------+---------------------+
        11 |/////////////////////////////////////////////| 10
           +---------------------+---------------------+
        13 |/////////////////////////////////////////////| 12
           +---------------------+---------------------+
        15 |/////////////////////////////////////////////| 14
           +---------------------+---------------------+
        17 |/////////////////////////////////////////////| 16
           +---------------------+---------------------+
        21 |/////////////////////////////////////////////| 20
           +---------------------+---------------------+
        23 |                status flag                  | 22 FQMODE
           +---------------------+---------------------+
        25 |/////////////////////////////////////////////| 24
           +---------------------+---------------------+
        27 |/////////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |/////////////////////////////////////////////| 30
           +---------------------+---------------------+
        33 |/////////////////////////////////////////////| 32
           +---------------------+---------------------+
        35 |/////////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |/////////////////////////////////////////////| 36
           +---------------------+---------------------+
```

FIRQB+FQJOB	The current job number times two.
FIRQB+FQMODE	Bit 14 of this word is set if any windows were unmapped as a result of this detach.

Errors

NOSUCH	The library ID specified at FIRQB+FQPPN in the data passed does not identify any library currently attached to the job.

Example

The following code detaches the library whose ID is stored at LIBID:

```
  MOVB     #DTRFQ,FIRQB+FQFIL              ;SET FUNCTION CODE
  MOV      LIBID,FIRQB+FQPPN               ;SET LIBRARY ID
  .PLAS
```

ELAFQ (Eliminate Address Window)

Form

```
  MOVB    #ELAFQ,FIRQB+FQFIL
          .
          .
          .
  (set up parameters in FIRQB)
          .
          .
          .
  .PLAS
```

Function

The ELAFQ subfunction of .PLAS eliminates an address window that was created by the job, unmapping the window if necessary. ELAFQ frees the APRs used by the window and makes them available for creating another window or for expanding the user job image size.

Privileges Required

None

Data Passed

```
                              FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                               Offset  monic
          +--------------------+--------------------+
        1 |//////////////////////////////////////////| 0
          +--------------------+--------------------+
        3 |//////////////////////////////////////////| 2
          +--------------------+--------------------+
        5 |////////////////////|    ELAFQ ( = 6)    | 4 FQFIL
          +--------------------+--------------------+
        7 |////////////////////|     window ID      | 6 FQPPN
          +--------------------+--------------------+
       11 |//////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |//////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |//////////////////////////////////////////| 14
          +--------------------+--------------------+
       17 |//////////////////////////////////////////| 16
          +--------------------+--------------------+
       21 |//////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |//////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |//////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |//////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |//////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |//////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |//////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |//////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQFIL	The function code ELAFQ (octal value equals 6).
FIRQB+FQPPN	The ID of the window to be eliminated (returned at FIRQB+FQPPN by the CRAFQ that created the window).

Data Returned

```
                              FIRQB
Mne-   Octal                                             Octal   Mne-
monic Offset                                             Offset  monic
          +---------------------+---------------------+
       1  |///////////////////////////////////////////|  0
          +---------------------+---------------------+
       3  |/////////////////////| current job no. * 2 |  2 FQJOB
          +---------------------+---------------------+
       5  |///////////////////////////////////////////|  4
          +---------------------+---------------------+
       7  |///////////////////////////////////////////|  6
          +---------------------+---------------------+
      11  |///////////////////////////////////////////| 10
          +---------------------+---------------------+
      13  |///////////////////////////////////////////| 12
          +---------------------+---------------------+
      15  |///////////////////////////////////////////| 14
          +---------------------+---------------------+
      17  |///////////////////////////////////////////| 16
          +---------------------+---------------------+
      21  |///////////////////////////////////////////| 20
          +---------------------+---------------------+
      23  |               status flags                | 22 FQMODE
          +---------------------+---------------------+
      25  |///////////////////////////////////////////| 24
          +---------------------+---------------------+
      27  |///////////////////////////////////////////| 26
          +---------------------+---------------------+
      31  |///////////////////////////////////////////| 30
          +---------------------+---------------------+
      33  |///////////////////////////////////////////| 32
          +---------------------+---------------------+
      35  |///////////////////////////////////////////| 34
          +---------------------+---------------------+
      37  |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQJOB   The current job number times two.

FIRQB+FQMODE   Two bits in this word indicate the status of the window:

```
Bit 13 = 1  The window was successfully eliminated.
       = 0  The window was not eliminated.

Bit 14 = 1  The address window was mapped to a resident
            library and has been unmapped.
       = 0  The address window was not mapped; no
            unmapping was done.
```

.PLAS MAPFQ

Data Passed

```
                              FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////| 0
          +---------------------+---------------------+
        3 |/////////////////////////////////////////////| 2
          +---------------------+---------------------+
        5 |/////////////////////|     MAPFQ ( =10)    | 4 FQFIL
          +---------------------+---------------------+
        7 |/////////////////////|      window ID      | 6 FQPPN
          +---------------------+---------------------+
       11 |/////////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |/////////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |             resident library ID           | 14 FQEXT
          +---------------------+---------------------+
       17 |         offset, in 32-word blocks         | 16 FQSIZ
          +---------------------+---------------------+
       21 |  length, in 32-word blocks, to be mapped  | 20 FQBUFL
          +---------------------+---------------------+
       23 |             desired access mode           | 22 FQMODE
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |/////////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |/////////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFIL	The function code MAPFQ (octal value equals 10).
FIRQB+FQPPN	The ID of the window to be mapped (returned at FIRQB+FQPPN by the CRAFQ subfunction call that created the window).
FIRQB+FQEXT	The ID of the resident library to which the window is to be mapped (returned as a full word at FIRQB+FQPPN by the ATRFQ subfunction call that attached the job to the resident library).

FIRQB+FQSIZ	The offset from the start of the library where the mapping is to begin, in 32-word blocks. A value of zero for this word indicates the window is to be mapped beginning at the first byte of the library. A value of 1 indicates that the window is to be mapped beginning at the 33rd word of the library (starting address = 64), and so forth.
FIRQB+FQBUFL	The length of the area to be mapped, in 32-word blocks. This value cannot be greater than the size of the window (specified at FIRQB+12 in the CRAFQ which created the window). Furthermore, this value, combined with the offset value at FIRQB+FQSIZ, cannot indicate an address beyond the end of the library. A value of 0 for this word defaults to either the size of the window or the space remaining in the library, whichever is smaller.
FIRQB+FQMODE	Bit 1 of this word specifies whether the window is to be mapped read/write or read-only: Bit 1 = 1 Read/write access. = 0 Read-only access. A separate setting for access in MAPFQ and in ATRFQ allows you to attach to a library read/write and map a portion of the library read-only. You cannot, however, attach to a library read-only and then map to the library read/write.

Data Returned

```
                               FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                 Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |/////////////////////| current job no. * 2 |  2 FQJOB
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 | length, in 32-word blocks, actually mapped| 20 FQBUFL
          +---------------------+---------------------+
       23 |                status flag                | 22 FQMODE
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQJOB	The current job number times two.
FIRQB+FQBUFL	Length actually mapped by the call, in 32-word blocks.
FIRQB+FQMODE	Bit 14 of this word is a status flag:

Bit 14 = 1	The window specified was already mapped; the window was unmapped before this mapping was done.
= 0	The window specified had no previous mapping; no unmapping was done for this call.

Errors

BADFUO	The offset and length specified are inconsistent; either the mapping attempted to go beyond the end of the library or the length is greater than the created window.
NOSUCH	Either the resident library ID or the address window ID is incorrect. (The job is not currently attached to the specified resident library or no address window has been created with the specified window ID.)

Example

The following code maps a window whose ID is stored at WINID to the attached library whose ID is stored at LIBID. The offset of 256 indicates that the mapping is to begin 8K words from the start of the library. Access to the library is read-only. For example:

```
MOVB    #MAPFQ,FIRQB+FQFIL             ;SET FUNCTION CODE
MOVB    WINID,FIRQB+FQPPN              ;SET WINDOW ID
MOV     LIBID,FIRQB+FQEXT              ;SET LIBRARY ID
MOV     #256.,FIRQB+FQSIZ              ;OFFSET 8K WORDS
CLR     FIRQB+FQBUFL                   ;MAP WINDOW SIZE OR
                                       ;TO END OF LIBRARY
CLR     FIRQB+FQMODE                   ;READ-ONLY ACCESS
.PLAS
```

Map Physical Memory

You can also use MAPFQ to map a window to physical memory. The data passed is the same as for a resident library except FIRQB+FQEXT contains -4. The data returned is the same.

You can map to monitor memory, XBUF, user memory, and so on, but not to locked memory (including virtual disk), or non-existent memory. If you try to map to locked or non-existent memory, you get a privilege violation error.

Privileges

You need RDMEM to map read/write or read-only. You need SYSMOD to map read/write. If the starting address in memory is 17760000 or higher (block number 177600), you also need SYSMOD for read-only.

Data Passed (Map Physical Memory)

```
                                FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |/////////////////////|     MAPFQ ( =10)    |  4 FQFIL
          +---------------------+---------------------+
        7 |/////////////////////|      window ID      |  6 FQPPN
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |                    -4                     | 14 FQEXT
          +---------------------+---------------------+
       17 |    starting address, in 32-word blocks    | 16 FQSIZ
          +---------------------+---------------------+
       21 |  length, in 32-word blocks, to be mapped  | 20 FQBUFL
          +---------------------+---------------------+
       23 |            desired access mode            | 22 FQMODE
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFIL	The function code MAPFQ (octal value equals 10).
FIRQB+FQPPN	The ID of the window to be mapped (returned at FIRQB+FQPPN by the CRAFQ subfunction call that created the window).
FIRQB+FQEXT	-4 to indicate map to physical memory.

FIRQB+FQSIZ	The offset from the start of memory where the mapping is to begin, in 32-word blocks. A value of zero for this word indicates the window is to be mapped beginning at the first byte of physical memory. A value of 1 indicates that the window is to be mapped beginning at the 33rd word of physical memory (starting address = 64), and so forth.
FIRQB+FQBUFL	The length of the area to be mapped, in 32-word blocks. This value cannot be greater than the size of the window (specified at FIRQB+12 in the CRAFQ which created the window). Furthermore, this value, combined with the offset value at FIRQB+FQSIZ, cannot indicate an address beyond the end of physical memory. A value of zero for this word defaults to either the size of the window or the space remaining in physical memory, whichever is smaller.
FIRQB+FQMODE	Bit 1 of this word specifies whether the window is to be mapped read/write or read-only: Bit 1 = 1 Read/write access. = 0 Read-only access. You need SYSMOD privilege to map read/write. You need RDMEM privilege to map either way. If you want to map the I/O page by specifying a starting address of 17760000 or higher (block number 177600), you need SYSMOD privilege to map either way. You can map to monitor memory, XBUF, or user memory, but not to locked memory (including virtual disk), or nonexistent memory.

Errors

PRVIOL	The mapping was unsuccessful because user privileges did not allow the access desired or you attempted to map to locked or nonexistent memory.

UMPFQ (Unmap Address Window)

Form

```
  MOVB     #UMPFQ,FIRQB+FQFIL
           .
           .
           .
  (set up parameters)
           .
           .
           .
  .PLAS
```

Function

The UMPFQ subfunction of .PLAS unmaps a specified address window from a resident library. (Note that a MAPFQ on an already mapped window unmaps the existing windows.)

Privileges Required

None

Data Passed

```
                                FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
           +---------------------+---------------------+
         1 |/////////////////////////////////////////////| 0
           +---------------------+---------------------+
         3 |/////////////////////////////////////////////| 2
           +---------------------+---------------------+
         5 |//////////////////////|    UMPFQ (= 12)     | 4 FQFIL
           +---------------------+---------------------+
         7 |//////////////////////|     window ID       | 6 FQPPN
           +---------------------+---------------------+
        11 |/////////////////////////////////////////////| 10
           +---------------------+---------------------+
        13 |/////////////////////////////////////////////| 12
           +---------------------+---------------------+
        15 |/////////////////////////////////////////////| 14
           +---------------------+---------------------+
        17 |/////////////////////////////////////////////| 16
           +---------------------+---------------------+
        21 |/////////////////////////////////////////////| 20
           +---------------------+---------------------+
        23 |/////////////////////////////////////////////| 22
           +---------------------+---------------------+
        25 |/////////////////////////////////////////////| 24
           +---------------------+---------------------+
        27 |/////////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |/////////////////////////////////////////////| 30
           +---------------------+---------------------+
        33 |/////////////////////////////////////////////| 32
           +---------------------+---------------------+
        35 |/////////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |/////////////////////////////////////////////| 36
           +---------------------+---------------------+
```

FIRQB+FQFIL	The function code UMPFQ (octal value equals 12).
FIRQB+FQPPN	The ID of the window to be unmapped (returned at FIRQB+FQPPN by the CRAFQ that created the window).

Data Returned

```
                               FIRQB
Mne- Octal                                            Octal  Mne-
monic Offset                                           Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////|  0
          +-------------------+-------------------+
        3 |///////////////////| current job no. * 2 |  2 FQJOB
          +-------------------+-------------------+
        5 |///////////////////////////////////////|  4
          +-------------------+-------------------+
        7 |///////////////////////////////////////|  6
          +-------------------+-------------------+
       11 |///////////////////////////////////////| 10
          +-------------------+-------------------+
       13 |///////////////////////////////////////| 12
          +-------------------+-------------------+
       15 |///////////////////////////////////////| 14
          +-------------------+-------------------+
       17 |///////////////////////////////////////| 16
          +-------------------+-------------------+
       21 |///////////////////////////////////////| 20
          +-------------------+-------------------+
       23 |              status flag              | 22 FQMODE
          +-------------------+-------------------+
       25 |///////////////////////////////////////| 24
          +-------------------+-------------------+
       27 |///////////////////////////////////////| 26
          +-------------------+-------------------+
       31 |///////////////////////////////////////| 30
          +-------------------+-------------------+
       33 |///////////////////////////////////////| 32
          +-------------------+-------------------+
       35 |///////////////////////////////////////| 34
          +-------------------+-------------------+
       37 |///////////////////////////////////////| 36
          +-------------------+-------------------+
```

FIRQB+FQJOB	The current job number times two.
FIRQB+FQMODE	Bit 13 of this word is set if the unmapping was successful.

Errors

BADFUO	The window ID specified is invalid (not in the range 1 to 7).
NOSUCH	The window ID specified is in the range 1 to 7, but no such window is currently created for the job.

Example

The following code unmaps the window whose ID is stored at WINID:

```
  MOVB     #UMPFQ,FIRQB+FQFIL         ;SET FUNCTION CODE
  MOVB     WINID,FIRQB+FQPPN
  .PLAS
```

.POSTN - Return Current Horizontal Position

Form

.POSTN

Function

The .POSTN directive returns the maximum line width and the current horizontal position of devices for which this information is relevant (line printers and terminals). Data is passed in the XRB defining the channel where the device is currently opened. The information is returned in the XRB.

Privileges Required

None

Data Passed

```
                            XRB
Mne-  Octal                                      Octal   Mne-
monic Offset                                     Offset  monic
         +---------------------+---------------------+
       1 |/////////////////////////////////////////////|  0
         +---------------------+---------------------+
       3 |/////////////////////////////////////////////|  2
         +---------------------+---------------------+
       5 |/////////////////////////////////////////////|  4
         +---------------------+---------------------+
       7 |//////////////////////| channel number * 2  |  6 XRCI
         +---------------------+---------------------+
      11 |/////////////////////////////////////////////| 10
         +---------------------+---------------------+
      13 |/////////////////////////////////////////////| 12
         +---------------------+---------------------+
      15 |/////////////////////////////////////////////| 14
         +---------------------+---------------------+
```

XRB+XRCI	Channel number times two; defines the channel on which the file or device is open.

Data Returned

```
                                XRB
Mne-  Octal                                               Octal   Mne-
monic Offset                                               Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////| maximum line length |  0 XRLEN
          +---------------------+---------------------+
        3 |/////////////////////|  current position   |  2 XRBC
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
```

XRB+XRLEN	The file or device's maximum line length plus one. For example, a value of 81. would indicate a maximum line length of 80. bytes.
XRB+XRBC	The file or device's current horizontal position is returned here. The value can range from zero (leftmost character) to the value returned at XRB+XRLEN minus one (rightmost character). If the device does not keep track of its own horizontal position, then this value is zero. The FLGPOS status bit returned in the FIRQB when the file or device was opened indicates whether the value returned here is meaningful.

Errors

NOTOPN	No file or device is open on the specified channel.
DETKEY	The device is a terminal that is detached.

Example

The following code requests the current horizontal position of the device open on channel 4:

```
  MOVB    #4*2,XRB+XRCI          ;SET CHANNEL TO 4
  .POSTN
```

.READ - Read Data from File or Device

Form

```
.READ
```

Function

The .READ directive reads data from a file or device previously opened on a channel. The amount of data read depends on the device and the size of the buffer area, as defined in the XRB. The number of bytes transferred is always less than or equal to the buffer size. The actual number of bytes read is returned in the XRB when the directive is complete. Specific details for each device are given in Table 3-5. The "best guess" buffer sizes (returned by the monitor at FIRQB+FQBUFL when the device was opened) are also shown, for comparison.

Table 3-5: Data Input with .READ

Device	Block Size, Bytes (Decimal)	"Best Guess," Bytes (Decimal)	.READ's Intent to Deliver	What Happens When	
				Buffer Size $\geq$Intent	Buffer Size <Intent
Byte-Oriented Devices (FLGFRC=1, FLGRND=0)					
Keyboard (Terminal)	N/A	128	1 line *	1 line *	Fill buffer; next .READ reads next part of line.
Pseudo Keyboard	N/A	128	Full buffer	N/A	N/A
Paper Tape Reader	N/A	128	Full buffer	N/A	N/A
Card Reader	N/A	160	1 card	1 card	Fill buffer; next .READ reads next part of card.
Block-Sequential Devices (FLGFRC=0, FLGRND=1)					
Magnetic Tape	18 to 32,767	512	1 block	1 block	Fill buffer; next .READ starts with new block. (Error returned.)
DECtape (file-structured)	510	510	1 block	1 block	Fill buffer; next .READ starts with new block. (No error returned.)
DMC/DMR	1 to 632	512	1 message	1 message	Delivers partial message. Next .READ delivers next message.
Block-Random Devices (FLGFRC=0, FLGRND=0)					
Disk	512	512	Full buffer	Fills buffer; next .READ starts with new block. **	
Flexible Diskette	***	512	Full buffer	Fills buffer; next .READ starts with new block or sector. **	
DECtape (non-file-structured)	512	512	1 block	1 block	Fill buffer; next .READ starts with new block. (No error returned.)

* A line is any number of characters terminated by RETURN, LINE FEED, ESCAPE, FORM FEED, CTRL/Z, CTRL/D, CTRL/C, or a user-set private delimiter.

** If the buffer size is not a multiple of the blocksize, the remainder of the last block is not read. The next .READ starts with a new block: either the next sequential block (XRB+XRBLK=0) or the nth block (XRB+XRBLK=n). No error is returned.

*** 512(block mode) or 128 (RX01 or RX02 single-density sector mode) or 256 (RX02 double- density sector mode)

Privileges Required

None

Data Passed

XRB

Mne-monic	Octal Offset			Octal Offset	Mne-monic
	1	size of the buffer in bytes, must ≠ 0		0	XRLEN
	3	(must be 0)		2	XRBC
	5	starting address of buffer		4	XRLOC
XRBLKM	7	MSB of block number	channel number * 2	6	XRCI
	11	LSB of block number to begin (0 = next)		10	XRBLK
	13	wait time for terminal input		12	XRTIME
	15	device-dependent modifier		14	XRMOD

XRB+XRLEN	The length of the input buffer, in bytes. This word must be nonzero. The amount of data read depends on both the device and the buffer size. However, the amount of data is never more than the buffer size. See Table 3-5 for details on reads for specific devices.
XRB+XRBC	This word must be passed as zero. The monitor returns the actual number of bytes transferred in this word location (see the Data Returned section).
XRB+XRLOC	The starting address of the buffer. For disk, flexible diskette, and magtape devices, this address must be on a word boundary. For all other devices, the buffer can begin on an odd address. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)
XRB+XRCI	Channel number times two; defines the channel for the read, as previously defined in an open (OPNFQ, CRTFQ, CREFQ, or CRBFQ functions of CALFIP).
XRB+XRBLKM	For files on disk, this byte contains the MSB of the block number to begin the read. This byte is combined with the word at XRBLK to form a 24-bit field defining the block number. This byte is ignored for nondisk devices.
XRB+XRBLK	For channels opened as file-structured, this word defines the starting block number for this read. (For large files, this word forms the LSB of the block number to start the read.) The value performs the same

	action as the BLOCK option for disk and the RECORD option for flexible diskette and non-file-structured DECtape (see the *RSTS/E Programming Manual*). Note that this word is ignored if the device is not a random-access device.
	If the device is random-access and this field is nonzero, it is interpreted as the block number where the read is to start (1 to n, where n is the length of the file, in 512-byte blocks). If the field is zero, the next sequential block is read. For example, if a disk file is being read with a 1024-byte buffer size, a .READ with this parameter equal to four would cause the fourth and fifth blocks of the file (512 bytes each) to be read into the buffer.
	For channels opened on disk as non-file-structured MODE 0, this word defines the device cluster number where the read is to begin. In this case, each .READ begins with a new device cluster, so the buffer size at XRB+XRLEN should be a multiple of the device cluster size.
XRB+XRTIME	If positive, the maximum time to wait for terminal input data, in seconds. Zero indicates an infinite wait. A negative value (bit 15 equals 1) indicates an infinite keyboard monitor wait. This wait is used by run-time systems that act as keyboard monitors for

their command input. This type of wait time acts as a flag to the monitor and to the batch subsystem that the job is in a command input wait state.

XRB+XRMOD — Input operation modifier; significant only for card reader, terminal, DMC/DMR, or paper tape devices. (The monitor informs you with the FLGMOD bit of the flag word returned at FIRQB+FQFLAG on the open whether or not the device accepts modifiers.) This parameter performs the same action as the RECORD modifier in BASIC-PLUS for these devices (see the *RSTS/E Programming Manual*).

Data Returned

```
                                   XRB
Mne- Octal                                                Octal  Mne-
monic Offset                                              Offset monic
         +---------------------+---------------------+
       1 |/////////////////////////////////////////////|  0
         +---------------------+---------------------+
       3 |       actual number of bytes read          |  2 XRBC
         +---------------------+---------------------+
       5 |/////////////////////////////////////////////|  4
         +---------------------+---------------------+
XRBLKM 7 | MSB of block number |/////////////////////|  6
         +---------------------+---------------------+
      11 |   LSB of block number where .READ began    | 10 XRBLK
         +---------------------+---------------------+
      13 |/////////////////////////////////////////////| 12
         +---------------------+---------------------+
      15 |/////////////////////////////////////////////| 14
         +---------------------+---------------------+
```

XRB+XRBC — Actual number of bytes just read. The value is between zero and the value of XRB+XRLEN passed in the XRB. If an error is returned on the read (as indicated by byte 0 of the FIRQB), this word may or may not be zero. That is, data may be input even if an error occurs.

For example, if the card reader detects an illegal card column punch combination, it places the decoded card data in the input buffer, substituting a special character for the bad column. The XRB+XRBC field is correctly set for the number of characters input, and an error is returned.

XRB+XRBLKM — For disk files, this byte contains the MSB of the block number just read. (See XRB+XRBLK.)

XRB+XRBLK — For random-access devices (see Table 3-5), this word contains the block number of the block just input with this .READ. Block numbers range from 1 to n (where n is the length of the file, in 512-byte blocks); they define the order in which the file was written. (For disks opened non-file-structured MODE 0, this is the device cluster number.)

Errors

The possible device-independent errors are:

BADCNT	The first three words of the XRB describing the input buffer are illegal. (Illegal byte count for I/O.)
BSERR	The specified channel number is illegal.
NOTOPN	No file or device is open on the specified channel number.
PRVIOL	The file or device open on the specified channel is write-only, or the caller did not obtain read access to the file or device when it was opened.

All other errors are device-dependent. Some common errors are:

DATERR	Some data error occurred. There may or may not be any valid data in the input buffer. This error is issued for parity errors, bad card columns, and so on.
HNGDEV	Some hard device I/O error occurred. There is usually no data in the input buffer when this error occurs. (For example, a .READ for an off-line device causes this error.)

Example

In this example, we assume that a disk file has been opened on channel 2. Space for the buffer is allocated with the .BLKB directive, and the buffer size and location are defined. The next sequential block in the file is to be read (XRB+XRBLK is zero).

```
BUFFER: .BLKB   512.                   ;ALLOCATE SPACE FOR BUFFER
                .
                .
                .
        CALL    CLRXRB
        MOV     #512.,XRB+XRLEN        ;SET BUFFER SIZE TO 512. BYTES
        MOV     #BUFFER,XRB+XRLOC      ;STARTING ADDRESS OF BUFFER
        MOVB    #2*2,XRB+XRCI          ;CHANNEL 2 FOR INPUT
        .READ
```

.READA -- Read Data from a Device (Asynchronous)

Form

```
.READA
```

Function

The asynchronous read directive (.READA) performs the same basic functions as the synchronous read directive (.READ): both move data between a device and a user program. The difference lies in the completion of the request. While synchronous read requests stall the job's execution until the request completes, asynchronous read requests do not stall; the program continues to run while the I/O request completes in parallel with program execution. The mechanism that the RSTS operating system uses to notify the user job upon I/O completion is an asynchronous completion routine.

Privileges Required

TUNE is required to perform asynchronous read functions.

Data Passed

```
                              FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
           +---------------------+---------------------+
         1 |///////////////////////////////////////////|  0
           +---------------------+---------------------+
         3 |///////////////////////////////////////////|  2
           +---------------------+---------------------+
         5 |        Completion Routine Address         |  4 FQFIL
           +---------------------+---------------------+
         7 |          User-supplied parameter          |  6 FQPPN
           +---------------------+---------------------+
        11 |///////////////////////////////////////////| 10
           +---------------------+---------------------+
        13 |///////////////////////////////////////////| 12
           +---------------------+---------------------+
        15 |///////////////////////////////////////////| 14
           +---------------------+---------------------+
        17 |///////////////////////////////////////////| 16
           +---------------------+---------------------+
        21 |///////////////////////////////////////////| 20
           +---------------------+---------------------+
        23 |///////////////////////////////////////////| 22
           +---------------------+---------------------+
        25 |///////////////////////////////////////////| 24
           +---------------------+---------------------+
        27 |///////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |///////////////////////////////////////////| 30
           +---------------------+---------------------+
        33 |///////////////////////////////////////////| 32
           +---------------------+---------------------+
        35 |///////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |///////////////////////////////////////////| 36
           +---------------------+---------------------+
```

FIRQB+FQFIL	Identifies the address in the user job where program flow should begin when the asynchronous read completes. If FIRQB+FQFIL is zero, no AST completion routine is executed. In this case, it is understood that the user is using some other mechanism to determine whether the I/O request has successfully completed.
FIRQB+FQPPN	This field is used by the user job to communicate information to the AST completion routine with the user's program. If FIRQB+FQFIL is zero, this value is ignored.

```
                                XRB
Mne- Octal                                            Octal  Mne-
monic Offset                                          Offset monic
          +-------------------+--------------------+
        1 |  size of the buffer in bytes, must ≠ 0  |  0 XRLEN
          +-------------------+--------------------+
        3 |               (must be 0)               |  2 XRBC
          +-------------------+--------------------+
        5 |        starting address of buffer       |  4 XRLOC
          +-------------------+--------------------+
XRBLKM  7 | MSB of block number |  channel number * 2 |  6 XRCI
          +-------------------+--------------------+
       11 |  LSB of block number to begin (0 = next) | 10 XRBLK
          +-------------------+--------------------+
       13 |       wait time for terminal input      | 12 XRTIME
          +-------------------+--------------------+
       15 |        device-dependent modifier        | 14 XRMOD
          +-------------------+--------------------+
```

XRB+XRLEN	The length of the input buffer, in bytes. This word must be nonzero. The amount of data read depends on both the device and the buffer size. However, the amount of data can never be more than the buffer size. See Table 3-5 for details on reads for specific devices.
XRB+XRBC	This word must be passed as zero. The monitor returns the actual number of bytes transferred in this word location (see the Data Returned section).
XRB+XRLOC	The starting address of the buffer. For disk, flexible diskette, and magtape devices, this address must be on a word boundary. For all other devices, the buffer can begin on an odd address. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)
XRB+XRCI	Channel number times two; defines the channel for the read, as previously defined in an open (OPNFQ, CRTFQ, CREFQ, or CRBFQ functions of CALFIP).
XRB+XRBLKM	For files on disk, this byte contains the MSB of the block number to begin the read. This byte is combined with the word at XRBLK to form a 24-bit field defining the block number. This byte is ignored for nondisk devices.
XRB+XRBLK	For channels opened as file-structured, this word defines the starting block number for this read. (For large files, this word forms the LSB of the block number to start the read.) The value performs the same action as the BLOCK option for disk and the RECORD

option for flexible diskette and non-file-structured DECtape (see the *RSTS/E Programming Manual*). Note that this word is ignored if the device is not a random-access device.

If the device is random-access and this field is nonzero, it is interpreted as the block number where the read is to start (1 to n, where n is the length of the file, in 512-byte blocks). If the field is zero, the next sequential block is read. For example, if a disk file is being read with a 1024-byte buffer size, a .READA with this parameter equal to four would cause the fourth and fifth blocks of the file (512 bytes each) to be read into the buffer.

For channels opened on disk as non-file-structured MODE 0, this word defines the device cluster number where the read is to begin. In this case, each .READA begins with a new device cluster, so the buffer size at XRB+XRLEN should be a multiple of the device cluster size.

XRB+XRTIME	If positive, the maximum time to wait for terminal input data, in seconds. Zero indicates an infinite wait. A negative value (bit 15 equals 1) indicates an infinite keyboard monitor wait. This wait is used by run-time systems that act as keyboard monitors for their command input. This type of wait time acts as a flag to the monitor and to the batch subsystem that the job is in a command input wait state.
XRB+XRMOD	Input operation modifier. For disk asynchronous reads, a value of one in this byte indicates that a read check should be performed. A read check does not transfer any data. The buffer address specified in the XRB is compared with the data on the disk.

Data Returned on Completion of the Directive

No meaningful data is returned by the directive except for a possible error code in byte 0 of the FIRQB. Useful data is returned instead to the completion routine.

Data Returned to the Completion Routine

```
                                FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
              +--------------------+--------------------+
          1   |////////////////////|     error code     |   0
              +--------------------+--------------------+
          3   |////////////////////|  job number * 2    |   2  FQJOB
              +--------------------+--------------------+
          5   |/////////////////////////////////////////|   4
              +--------------------+--------------------+
          7   |        User-supplied parameter          |   6  FQPPN
              +--------------------+--------------------+
         11   |/////////////////////////////////////////|  10
              +--------------------+--------------------+
         13   |/////////////////////////////////////////|  12
              +--------------------+--------------------+
         15   |/////////////////////////////////////////|  14
              +--------------------+--------------------+
         17   |/////////////////////////////////////////|  16
              +--------------------+--------------------+
         21   |/////////////////////////////////////////|  20
              +--------------------+--------------------+
         23   |/////////////////////////////////////////|  22
              +--------------------+--------------------+
         25   |/////////////////////////////////////////|  24
              +--------------------+--------------------+
         27   |/////////////////////////////////////////|  26
              +--------------------+--------------------+
         31   |/////////////////////////////////////////|  30
              +--------------------+--------------------+
         33   |/////////////////////////////////////////|  32
              +--------------------+--------------------+
         35   |/////////////////////////////////////////|  34
              +--------------------+--------------------+
         37   |/////////////////////////////////////////|  36
              +--------------------+--------------------+
                              XRB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |        actual number of bytes read         |  2 XRBC
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
XRBLKM  7 | MSB of block number |/////////////////////|  6
          +---------------------+---------------------+
       11 |   LSB of block number where .READ began   | 10 XRBLK
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
```

XRB+XRBC	Actual number of bytes just read. The value is between zero and the value of XRB+XRLEN passed in the XRB. If an error is returned on the read (as indicated by byte 0 of the FIRQB), this word may or may not be zero. That is, data may be input even if an error occurs.
XRB+XRBLKM	For disk files, this byte contains the MSB of the block number just read. (See XRB+XRBLK.)
XRB+XRBLK	For random-access devices (see Table 3-5), this word contains the block number of the block just input with this .READA. Block numbers range from 1 to n (where n is the length of the file, in 512-byte blocks); they define the order in which the file was written. (For disks opened non-file-structured MODE 0, this is the device cluster number.)

Errors

There are three classes of errors that can cause the failure of an asynchronous I/O request: directive errors; system-driver errors; and device-dependent errors.

Directive errors are immediately visible upon examination of the FIRQB on return from the directive. These errors are similar to the syntax errors checked by a compiler. If an error is found, the I/O is not attempted, and the error is reported through the standard FIRQB+0 mechanism. Some common directive errors are:

BADCNT	The completion routine address is invalid or odd.
BSERR	The specified channel number is illegal.

NOTOPN	No file or device is open on the specified channel number.
PRVIOL	The file or device open on the specified channel is write-only, or the caller did not obtain read access to the file or device when it was opened.

System-driver errors occur before the asynchronous request is issued to the appropriate device driver. These errors are related to the lack of system resources, for example:

INUSE	Too much outstanding I/O for the job (over quota).
NOBUFS	The system could not get a small buffer for the request.

Any of the previous errors are reported on return from the .READA directive. If none of these errors occurred, the read operation is started and the directive returns a value of 0 in FIRQB+0. The read operation may still fail at a later point; its success or failure is indicated in FIRQB+0 when the completion routine is entered.

Other system-driver errors are reported within the AST completion routine:

EOF	End-of-file (EOF) on the specified device.
NOROOM	No room for the user on the specified device.

Device-dependent errors can only be found during the attempt of the actual I/O transfer. These errors can only be reported to the job in the asynchronous I/O completion routine. Some common device-dependent errors are:

DATERR	Some data error occurred. There may or may not be any valid data in the input buffer. This error is issued for parity errors, bad card columns, and so on.
HNGDEV	Some hard device I/O error occurred. There is usually no data in the input buffer when this error occurs. (For example, a .READA for an off-line device causes this error.)

Example

The following example performs an asynchronous disk-to-disk copy. The program initially issues three read requests (.READA) on the first disk and issues a corresponding write request (.WRITA) on the second device when each .READA completes. After each .WRITA completes, another advanced .READA is issued. Thus, there is always one outstanding .READA and at least one outstanding .WRITA. Each new I/O request within the routine uses the same buffer address as the request that has just finished. This is an easy way to keep track of the free buffers.

Note that the program does not handle bad blocks efficiently; nor does it prompt for the disk specifications. All errors are treated as fatal.

```
; Register usage:
; R2 = Input block number
; R4 = Buffer number (carried in the user-created parameter field)

.PSECT  CODE

START:  CALL    CLRFQB                  ; Clear FIRQB
;
; Open input disk device = DL0:
;
        MOVB    #OPNFQ,FIRQB+FQFUN      ; Function code
        MOVB    #3*2,FIRQB+FQFIL        ; Set input channel = 3
        MOV     #"DL,FIRQB+FQDEV        ;  on an RL02
        MOV     #377*400+0,FIRQB+FQDEVN ;  unit 0, device real
        MOV     #200*400+8192.,FIRQB+FQMODE ;  read-only mode
        CALFIP                          ; Go for it
        MOV     FIRQB,R0                ; Error?
        BNE     FATAL                   ; Stop and check it out
;
; Open output disk device = DL1:
;
        MOVB    #OPNFQ,FIRQB+FQFUN      ; Function code
        MOVB    #4*2,FIRQB+FQFIL        ; Set output channel = 4
        MOV     #"DL,FIRQB+FQDEV        ;  on an RL02
        MOV     #377*400+1,FIRQB+FQDEVN ;  unit 1, device real
        CALFIP                          ; Go for it
        MOV     FIRQB,R0                ; Error?
        BNE     FATAL                   ; Stop and check it out
;
; Issue the first three initial reads
;
        CALL    CLRFQX                  ; Clear FIRQB & XRB
        MOV     #3,R1                   ; 1 read + 2 read-aheads
        CLR     R2                      ; Start at virtual block zero
        MOV     #BUFFER,R4              ; Load 1st buffer address

10$:    MOV     #512.,XRB+XRLEN         ; Set buffer size to 512.
        CLR     XRB+XRBC                ; XRBC = 0
        MOV     R4,XRB+XRLOC            ; Set our next buffer address
        MOVB    #3*2,XRB+XRCI           ; Input channel times 2
        INC     R2                      ; Next input block number
        MOV     R2,XRB+XRBLK            ; Read that block
        MOV     #R$AST,FIRQB+FQFIL      ; Go to R$AST after read
        MOV     R4,FIRQB+FQPPN          ; User-created parameter
        .READA                          ; Do an asynchronous read
        MOV     FIRQB,R0                ; Directive error?
        BNE     FATAL                   ; Yes, stop
        ADD     #512.,R4                ; Get next buffer address
        SOB     R1,10$                  ; Issue next read-ahead

20$:    MOV     #100001, XRB+0          ; Conditional sleep
        .SLEEP
        BR      20$                     ; and do that until done
;
;All .READA's come here at completion time
;
R$AST:  MOV     FIRQB,R0                ; Error?
        BEQ     60$                     ; No, continue
        CMPB    #13,R0                  ; End-of-disk (EOD) ?
        BEQ     CLSIN                   ; Yes, close I/O disks
        BR      FATAL                   ; No, unexpected error, stop

60$:    MOV     #512.,XRB+XRLEN         ; Reset buffer size
        MOV     #512.,XRB+XRBC          ; Want to write entire buffer
        MOV     FIRQB+FQPPN,XRB+XRLOC ; Use the same buffer address
        MOVB    #4*2,XRB+XRCI           ; Output channel * 2
                                        ; XRBLK has block number
        MOV     #W$AST,FIRQB+FQFIL      ; Completion routine address
        .WRITA                          ; Write that block
        BR      ASTDON                  ; Join the common code

;
; All .WRITA's come here at completion time
;
W$AST:  MOV     FIRQB,R0                ; Error?
        BEQ     70$                     ; No, continue
        CMPB    #13,R0                  ; End-of-disk (EOF) ?
        BEQ     CLSIN                   ; Yes, close I/O disks
        BR      FATAL                   ; No, unexpected error, stop

70$:    MOV     #512.,XRB+XRLEN         ; Set buffer size
        CLR     XRB+XRBC                ; XRBC = zero
        MOV     FIRQB+FQPPN,XRB+XRLOC ; Use the same buffer address
        MOVB    #3*2,XRB+XRCI           ; Input channel * 2
        INC     R2                      ; Increment to next block
        MOV     R2,XRB+XRBLK            ; XRBLK has block number
        MOV     #R$AST,FIRQB+FQFIL      ; Completion routine address
        .READA                          ; Read that block
        BR      ASTDON                  ; Join the common code
;
; AST common code
;
ASTDON: MOV     FIRQB,R0                ; Any directive errors
        BNE     FATAL                   ; Yes, stop
        .ASTX                           ; No, just return to mainline
;
.SBTTL  FIRQB/XRB clearing routines
```

```
.ENABL  LSB

CLRFQX: PUSH    <R0,R1>                 ; Save R0, R1
        MOV     #FIRQB,R0               ; Point to FIRQB
        MOV     #<<FQBSIZ+XRBSIZ>/2>,R1 ; Compute how many words
        BR      200$

CLRFQB: PUSH    <R0,R1>                 ; Save R0, R1
        MOV     #FIRQB,R0               ; Point to FIRQB
        MOV     #<FQBSIZ/2>,R1          ; Compute how many words
        BR      200$

CLRXRB: PUSH    <R0,R1>                 ; Save R0, R1
        MOV     #XRB,R0                 ; Point to XRB
        MOV     #<XRBSIZ/2>,R1          ; Compute how many words

200$:   CLR     (R0)+                   ; Zero it out
        SOB     R1,200$                 ; Until all done
        POP     <R1,R0>                 ; Restore R0, R1
        RETURN
;
; Errors (code in R0) come here
;
FATAL:  BPT                             ; All errors come here
OUT:    .EXIT                           ; All done
;
; Close input disk (may still have outstanding .READAs)
;
CLSIN:  CALL    CLRFQB                  ; Clear FIRQB
        MOVB    #CLSFQ,FIRQB+FQFUN      ; CLOSE function
        MOVB    #3*2,FIRQB+FQFIL        ; Set channel = 3
        CALFIP                          ; Do it
;
; Close output disk (may still have outstanding .WRITAs)
;
CLSIN:  CALL    CLRFQB                  ; Clear FIRQB
        MOVB    #CLSFQ,FIRQB+FQFUN      ; CLOSE function
        MOVB    #4*2,FIRQB+FQFIL        ; Set channel = 4
        CALFIP                          ; Do it
        BR      OUT                     ; Then, exit

.PSECT  DATA

BUFFER: .BLKB   512.                    ; Buffer one
        .BLKB   512.                    ; Buffer two
        .BLKB   512.                    ; Buffer three
BUFEND:         ; Address for EOB

.END    START
```

.RTS - Pass Control to Run-Time System

Form

```
  .RTS
```

Function

The .RTS directive passes control to a run-time system at the P.NEW entry point, where the intent is not to run an executable file (see Chapter 2). .RTS performs one of four functions:

- o Passes control to the job's keyboard monitor
- o Passes control to a named run-time system
- o Passes control to a named run-time system and establishes it as the job's keyboard monitor
- o Passes control to a named run-time system without changing job context information

The first function is generally part of a run-time system's exit processing; you do not normally use .RTS for this purpose in a user program. The following code shows how .RTS is used in exit processing (for example, in the RT11 run-time system's .EXIT directive):

```
EXIT:   CALL    CLRFQB                  ; Clear FIRQB
        .RTS                            ; Exit to default KBM
        JMP     PROMPT                  ; If that's this KBM, then
                                        ; prompt for another command
```

The other three functions of .RTS apply to both user programs and run-time systems.

Note that once established, a job's keyboard monitor replaces the default keyboard monitor as the one to which control passes by default. This concept is best explained by example.

The SET JOB/KEYBOARD_MONITOR command uses the .RTS directive to establish the run-time system to which it passes control as the job's keyboard monitor (see the *RSTS/E System Manager's Guide*). Consider the following sequence:

```
$ SET JOB/KEYBOARD_MONITOR=RT11
.MAC
MAC>CTRL/C
.CTRL/C
.$SET JOB/KEYBOARD_MONITOR

$
```

The first line shows the DCL prompt ($), indicating that the user is in the DCL keyboard monitor. Then:

1. The user executes the SET JOB/KEYBOARD_MONITOR command to pass control to the RT11 run-time system. This command establishes RT11 as the job's keyboard monitor. The RT11 run-time system displays the period prompt.

2. The user runs MAC, the RSX assembler. Control passes to the RSX emulator, which loads the assembler and runs it. MAC displays the "MAC>" prompt.

3. The user, realizing the mistake in typing MAC instead of MACRO, enters CTRL/C to exit. Since RT11 has been established as the job's keyboard monitor, control passes back to RT11, and it displays the period prompt.

4. The user, wanting to get back to DCL, enters another CTRL/C. Since RT11 is the job's keyboard monitor, however, another period prompt appears.

5. The user finally issues the SET JOB/KEYBOARD_MONITOR command (with the $ prefix to indicate that it is a DCL command), leaving out the run-time system name. This command transfers control back to the system default keyboard monitor (DCL), establishing it once again as the job's keyboard monitor.

The four cases in which the .RTS directive can be used are:

- o In the first case, the .RTS directive is used to switch control back to the run-time system already established as the job's keyboard monitor. In this case, the name of the run-time system is not known. The word in the FIRQB that would contain the first part of the run-time system name is zero. If the run-time system that issues the .RTS is not itself the job's keyboard monitor, or the currently running program is a privileged program, control is passed to the job keyboard monitor at entry point P.NEW. If the run-time system that issues the .RTS is the job's keyboard monitor and the current program is not a privileged program, control returns to the instruction following the .RTS with no error indicated.

- o In the second case, the .RTS directive is used to switch control to the run-time system named in the FIRQB. (Run-time systems are made known to the monitor by the system manager with the INSTALL/RUNTIME_SYSTEM command. With this command, the system manager defines a file (filename.RTS) as an auxiliary run-time system. The monitor regards "filename" as the run-time system's name.) If the run-time system that issues the .RTS directive is not itself the named run-time system, or the currently running program is a privileged program, control passes to the named run-time system at the P.NEW entry point. If the run-time system that issues the .RTS directive is the named run-time system and the current program is not a privileged program, control returns to the instruction following the .RTS with no error indication. If the named run-time system does not exist or is not available for some reason, control returns to the instruction following the .RTS with an error in byte 0 of the FIRQB.

- In the third case, the .RTS directive is used to switch control to a named run-time system and establish it as the job's keyboard monitor. If the run-time system that issues the .RTS is not itself the named run-time system, or the currently running program is a privileged program, control passes to the named run-time system at the P.NEW entry point, and the named run-time system is established as the job's keyboard monitor. If the run-time system that issues the .RTS names itself as the run-time system and the current program is not a privileged program, control returns to the instruction following the .RTS with no error, and the run-time system is established as the job's keyboard monitor. If the named run-time system does not exist or is unavailable, control returns to the instruction following the .RTS with an error in byte 0 of the FIRQB.

- In the fourth case, the .RTS directive is used to switch control to a named run-time system preserving the job's current job-context information. Control passes to the named run-time system at the P.NEW entry point, but the monitor does not refresh the keyword, reset the stack pointer, or perform any of the initialization operations (see the discussion of P.NEW in Chapter 2).

In the first three cases, if the program currently running had temporary privileges in effect at one time (whether it has since dropped them or not), then the transfer of control to the P.NEW entry point is unconditional. In addition, all of the user's memory above NSTORG is cleared to ensure any restricted data kept by the program is erased from memory. Finally, the monitor drops temporary privileges.

Note

Do not use this directive from programs running under the RT11 run-time system, because RT11 uses the lowest 512 bytes differently than other run-time systems. The correct way to terminate a program running under RT11 is to exit through the RT11 emulator.

Although you can use this directive to terminate a program running under the RSX run-time system, DIGITAL recommends that you use the RSX EXIT$ or

EXST$ directive instead (see Chapter 5). Using .RTS may cause unexpected results.

Privileges Required

None

Data Passed

```
                              FIRQB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////|  0
          +--------------------+--------------------+
        3 |/////////////////////////////////////////|  2
          +--------------------+--------------------+
        5 |/////////////////////////////////////////|  4
          +--------------------+--------------------+
        7 |/////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |          run-time system name           | 10 FQNAM1
          +                                         +
       13 |            (2 words of RAD50)           | 12
          +--------------------+--------------------+
       15 | -1, Switch Kbm.; -2, don't change context | 14 FQEXT
          +--------------------+--------------------+
       17 |/////////////////////////////////////////| 16
          +--------------------+--------------------+
       21 |/////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |/////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |/////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |/////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |/////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |/////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |/////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |/////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQNAM1	If control is being passed to a named run-time system (cases 2, 3, and 4 described previously), the name of the run-time system is stored as two words in RAD50 format beginning here. If control is being passed to the job's keyboard monitor (case 1), the word beginning here must contain a value of zero.

FIRQB+FQEXT   To establish a named run-time system as the job's keyboard monitor (case 3), set this word to -1. To switch control to a named run-time system without altering job-context information (case 4), set this word to -2. When the word at FIRQB+FQNAM1 is zero, this word is ignored.

```
                                XRB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
          +--------------------+---------------------+
        1 |//////////////////////////////////////////| 0
          +--------------------+---------------------+
        3 |//////////////////////////////////////////| 2
          +--------------------+---------------------+
        5 |//////////////////////////////////////////| 4
          +--------------------+---------------------+
        7 |//////////////////////////////////////////| 6
          +--------------------+---------------------+
       11 |                                          | 10 XRBLK
          +                                          +
       13 |         unaltered whether or not         | 12
          +   control passes to new run-time system  +
       15 |                                          | 14
          +--------------------+---------------------+
```

XRB+XRBLK   The three words starting at this location pass unaltered to the new run-time system. They are also unaltered if control returns in line.

Data Returned

Since control usually passes to some other run-time system at its P.NEW entry point, no arguments as such are returned by .RTS. The three words starting at XRB+10 remain unaltered if control returns in line.

Errors

NORTS	No run-time system exists with the specified name.
PRVIOL	The named run-time system does exist but cannot be switched to for some reason. (For example, if the switch is made to a named run-time system with the intent of establishing it as the job's keyboard monitor and its PF.KBM bit in the P.FLAG word is not set to 1.)

Example

The following example establishes NEWRTS (which the system manager must have installed with INSTALL/RUNTIME_SYSTEM/KEYBOARD_MONITOR command) as the job's keyboard monitor:

```
MOV     #^RNEW,FIRQB+FQNAM1           ; Set run-time system
MOV     #^RRTS,FIRQB+FQNAM1+1         ; Name to "NEWRTS"
MOV     #-1,FIRQB+FQEXT               ; Establish as job kbm
.RTS
```

.RUN - Run a Program

Form

```
.RUN
```

Function

The .RUN directive searches for a binary (executable) file (defined in the FIRQB), opens it on channel 15, and passes control to the P.RUN entry point of the run-time system associated with the file. The associated run-time system is identified in the file's directory information. This directory information is initially set to indicate the run-time system under which the file was created. The system manager can change the run-time system associated with the file with the SET FILE/RUNTIME_SYSTEM command (see the *RSTS/E System Manager's Guide*).

The run-time system is responsible for loading and executing the file (see the P.RUN description in Chapter 2). Control is not returned in line unless an error occurs. The monitor drops any temporary privileges in effect under the previously running program.

Privileges Required

File execute access required; either by protection code or privilege (GREAD or WREAD).

Data Passed

```
                              FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |   project number    | programmer number   |  6 FQPPN
          +---------------------+---------------------+
       11 |          file name in RAD50 format        | 10 FQNAM1
          +                                           +
       13 |                 (2 words)                 | 12
          +---------------------+---------------------+
       15 |    file type in RAD50 format (1 word)     | 14 FQEXT
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |     device name (2 ASCII characters)      | 30 FQDEV
          +---------------------+---------------------+
       33 |  ≠0, unit no. real  | device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |              entry parameter              | 36 FQNENT
          +---------------------+---------------------+
```

FIRQB+FQPPN	The PPN of the file to be opened. The project number is specified in the high byte, the programmer number in the low byte. If this word is passed as zero, the PPN defaults to the PPN of the job that issues the .RUN directive.
FIRQB+FQNAM1	The file name of the file to be opened; two words in RAD50 format.
FIRQB+FQEXT	The file type for the file to be opened; one word in RAD50 format. If you set this word to -1, the monitor will search for the file name supplied, substituting the default file type for the currently installed set

	of run-time systems until a file with the given name and one of the default file types is found.
FIRQB+FQDEV	The device name of the file to be opened, as two ASCII characters. If the device name is not a disk, the .RUN directive returns in line with one of the soft errors described in the Errors section. If you pass a full word of zero here and in FIRQB+FQDEVN, the public disk structure (SY:) is searched for the named file.
FIRQB+FQNENT	A 16-bit parameter word can be passed to the run-time system here. If the calling program has temporary privileges in effect, bit 15 is unchanged. Otherwise, bit 15 is cleared. The monitor takes no other action as a result of the contents of this word; any processing is up to the run-time system. (See the P.RUN entry point, Chapter 2.)

Errors

When an error occurs in a .RUN, FIRQB+FQFLAG is set to zero to indicate a hard error or -1 to indicate a soft error.

A hard error means the .RUN failed. A soft error also means the .RUN failed, but the run-time system may be able to recover. For example, when you use the BASIC-PLUS RUN command, BASIC-PLUS first executes a .RUN. If the .RUN returns a soft error, BASIC-PLUS performs an OLD (which compiles the program) and then executes .RUN again.

Hard Errors

BADNAM	The specified file name was zero. This is an illegal file name for disk files.
DEVNFS	The specified disk device is currently being used non-file-structured.
NOTCLS	Channel 15 is currently open. It must be closed before any .RUN call. (This error occurs only if channel 15 is open on a nondisk device. If a disk file is open, the channel is closed automatically.)
NODEVC	The specified device does not exist or is in an illegal format.
NORTS	The run-time system named in the file's directory information has not been installed.
NOTMNT	The specified disk device is not now mounted.
PAKLCK	The disk pack on which the file exists is locked against further file opens.

Soft Errors

NOSUCH	The file was not found. Note that if FIRQB+FQEXT was -1 in the data passed, the monitor has looked for the given file name with all default runnable file types for the currently installed run-time systems. In this case, a source version of the file may still exist.
PRVIOL	The device is not disk but is still a legal device on the system. Or the file was found and is on disk but does not have the compiled program bit set in its file protection code (bit 6).

Example

The following example uses the .FSS directive to translate a user-typed string to the FIRQB format. If no errors in the .FSS occur, a test is made to see if a file type was specified. If not, a -1 is supplied in FIRQB+FQEXT so the monitor searches for the given file name with all possible default file types that are runnable.

```
(read user-typed line, set up FIRQB for .FSS)
              .
              .
              .
        .FSS
        TSTB    FIRQB                   ; Error on .FSS?
        BNE     ERRTN                   ; Branch to process error
        BIT     #100000,XRB+XRBLK       ; Invalid device name?
        BNE     BADDEV                  ; Branch to process error
        BIT     #10,XRB+XRBLK           ; File type given?
        BNE     SKIP1                   ; Yes, skip next
        MOV     #-1,FIRQB+FQEXT         ; No, search all
SKIP1:  CLR     FIRQB+FQNENT            ; Entry parameter is zero
        .RUN
```

.SET - Set Keyword Bits

Form

.SET

Function

The .SET directive sets certain bits in the keyword (KEY) location in the user job image (see Chapter 2). The bits to be set are passed to the monitor in the XRB.

Privileges Required

TUNE is required to set the special run priority bit (JFSPRI) or the lock in memory bit (JFLOCK).

Data Passed

```
                                  XRB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                 Offset  monic
           +---------------------+---------------------+
         1 |      bits to be set are set to 1 here     |  0 XRLEN
           +---------------------+---------------------+
         3 |///////////////////////////////////////////|  2
           +---------------------+---------------------+
         5 |///////////////////////////////////////////|  4
           +---------------------+---------------------+
         7 |///////////////////////////////////////////|  6
           +---------------------+---------------------+
        11 |///////////////////////////////////////////| 10
           +---------------------+---------------------+
        13 |///////////////////////////////////////////| 12
           +---------------------+---------------------+
        15 |///////////////////////////////////////////| 14
           +---------------------+---------------------+
```

```
                  JFSPRI---------+
               JFFPP---------+   |
            JFPRIV-------+   |   |
         JFSYS-------+   |   |   |
      JFNOPR-----+   |   |   |   |
   JFBIG-----+   |   |   |   |   |
JFLOCK---+   |   |   |   |   |   |
         |   |   |   |   |   |   |
    15  14  13  12  11  10   9   8 | 7   6   5   4   3   2   1   0
   +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
   +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

JFLOCK	Can be set only by jobs with TUNE privilege. When this bit is set, the monitor swaps the user job image out of memory only when: o The job issues a .CORE directive to expand the memory allocated for the user job image, and there is not sufficient room in physical memory to do the expansion. In this case, the job is swapped out to disk and back in at the indicated size. o A fatal error, such as a memory parity failure, occurs.
JFBIG	Cannot be set by any job; masked off, that is, the corresponding bits in KEY cannot be set by the job with the .SET directive.
JFNOPR	Cannot be set by any job; masked off.
JFSYS	Can be set by a job only if JFSYS was set at one time and the temporary privileges gained were only temporarily dropped. (See description of KEY, Chapter 2 and the .CLEAR directive.)
JFPRIV	Cannot be set by any job; masked off.
JFFPP	Can be set by any caller if the PDP-11/45 compatible hardware floating-point unit exists; masked off if it does not exist. When this bit is set, the monitor will save information in the floating-point unit as part of the job-context information kept when jobs are swapped in and out of memory.
JFSPRI	Can be set only by a caller with TUNE privilege. Setting JFSPRI raises the job's run priority by one-half step. (That is, it sets bit 2 of the system-controlled low-order three bits of the run priority. Priorities are normally set by the system manager.)

All other bits in the XRB word are masked off.

Data Returned

The .SET directive does not return any meaningful data.

Errors

No errors are possible with the .SET directive.

Example

The following code sets JFLOCK, allowing the job to remain locked in memory:

```
  MOV     #JFLOCK,XRB+XRLEN            ;SET JFLOCK
  .SET
```

.SLEEP - Suspend Job

Form

```
   .SLEEP
```

Function

The .SLEEP directive causes the monitor to suspend the job for some specified time interval or until an event occurs that the job should be aware of. Optionally, the monitor checks, before the job is suspended, to see if some event has already occurred which would cause it to awaken. If so, the job is not suspended. Control returns in line in either case.

When a .SLEEP is executed, execution of the job is suspended until one of the following happens:

- o The sleep time (specified in the XRB) expires.
- o A local or network message is queued for the job assuming that the job is using local or network send/receive services. (See the .MESAG directive.)
- o A delimiter is typed on a terminal that this job has opened or assigned.
- o The system manager disables logins. (This could occur if the system is being shut down.)
- o A state change occurs on a pseudo keyboard assigned to the job. (The job has printed output for the controlling job to read or has entered an input wait state.)
- o The DMC11/DMR11 driver (XM:) is open and a message is pending for the job.
- o An AST completion is queued.

If you request it, the monitor checks for the following conditions before suspending the job:

- o A delimiter has been typed on any terminal opened by the job or any terminal assigned to the job if the job also has a keyboard open on a nonzero channel.
- o A message has been queued for the job.
- o A state change has occurred on a pseudo keyboard opened by the job.

- o The job has a DMC11/DMR11 device driver open and a message has been received by that device driver.

- o One or more AST completion routines are pending.

If the monitor determines that any of these conditions are true, it does not execute the .SLEEP. This feature is known as conditional sleep.

Data Passed

```
                               XRB
Mne-  Octal                                           Octal   Mne-
monic Offset                                           Offset  monic
          +--------------------+--------------------+
        1 |          sleep time, in seconds          |  0 XRLEN
          +--------------------+--------------------+
        3 |/////////////////////////////////////////|  2
          +--------------------+--------------------+
        5 |/////////////////////////////////////////|  4
          +--------------------+--------------------+
        7 |/////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |/////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |/////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |/////////////////////////////////////////| 14
          +--------------------+--------------------+
```

XRB+XRLEN	This word defines the sleep interval, in seconds. If the value is zero, then .SLEEP returns immediately. If bit 15 is set, the monitor performs the conditional sleep checks. If any of the previously listed conditions are present, the .SLEEP is not executed.

Data Returned

The .SLEEP directive does not return any meaningful data.

Errors

No errors are possible with the .SLEEP directive.

Example

```
  MOV     #5,XRB+XRLEN                    ; Set timer to 5 seconds
  .SLEEP
```

.SPEC - Special Functions for I/O

Form

```
          .
          .
          .
  (set XRB for special function)
          .
          .
          .
     .SPEC
```

Function

The .SPEC directive performs special functions for:

- o Disks
- o Flexible diskettes
- o Line printers
- o Magnetic tapes
- o Pseudo keyboards
- o Terminals

.SPEC for Disk

For disk, the .SPEC directive lets you explicitly lock up to seven disk blocks on a file open for update (mode parameter). A locked block cannot be accessed by another user (or from another channel). This extends the implicit lock feature, by which the last block or blocks read on a file open for update cannot be accessed by anyone else. The disk special functions also let you release explicit and implicit locks. (All locks, both explicit and implicit, are released when the file is closed.)

There is also a .SPEC option to perform a comparison operation between the contents of a buffer and one or more blocks of a disk file.

Privileges Required

None

Data Passed (Disk - Except Compare)

```
                                 XRB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +---------------------+---------------------+
        1 |            special function code          |  0 XRLEN
          +---------------------+---------------------+
        3 |        LSB of block number (release)      |  2 XRBC
          +---------------------+---------------------+
        5 |/////////////////////|MSB of block no. (rl)|  4 XRLOC
          +---------------------+---------------------+
XRBLKM  7 |  DSKHND (=octal 0)  | channel number * 2  |  6 XRCI
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
```

XRB+XRLEN Defines function to be performed:

Code	Meaning
0	Release any implicit lock and all explicit locks. The monitor deallocates the extended internal table space it needed to do the explicit locks. (See code 3.)
1	Release implicit lock.
2	Make implicit lock into explicit lock.

3	Release the explicit lock on the block specified by the word at XRB+XRBC (LSB) and the byte at XRB+XRLOC (MSB). If all three bytes are zero, all explicit locks are released, but the monitor does not deallocate the extra space needed to do explicit locks. (This is useful if you intend to use the explicit lock feature again during this run. An error occurs if no space is available for this purpose.)
4	Make implicit lock into explicit lock and release the implicit lock.

XRB+XRBC, XRB+XRLOC	Both these bytes specify the starting block number for releasing an explicit lock. If these bytes are zero, all explicit locks are released, but the monitor retains the extended table area it needs to maintain these locks. (This is useful if you want to use this capability again during a run.)
XRB+XRCI	Channel number times two; defines the channel for the lock/unlock operation.
XRB+XRBLKM	Handler index for disk: DSKHND (octal value = 0).

Data Passed (Disk - Compare)

```
                              XRB
Mne- Octal                                             Octal   Mne-
monic Offset                                           Offset  monic
          +---------------------+---------------------+
        1 |                    -1                     |  0 XRLEN
          +---------------------+---------------------+
        3 |                buffer size                |  2 XRBC
          +---------------------+---------------------+
        5 |              buffer address               |  4 XRLOC
          +---------------------+---------------------+
XRBLKM  7 |     DSKHND (=0)     | channel number * 2  |  6 XRCI
          +---------------------+---------------------+
       11 |  LSB of block number to begin compare     | 10 XRBLK
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |/////////////////////| MSB of block number | 14 XRMOD
          +---------------------+---------------------+
```

XRB+XRBC	Size of the data buffer to be compared.
XRB+XRLOC	Address of the data buffer to be compared.
XRB+XRBLK	LSB of the first block number on the disk to be compared with the data buffer.

XRB+XRMOD	MSB of the first block number on the disk to be compared with the data buffer.

Data Returned

Except for a possible error code in byte 0 of the FIRQB, the disk subfunctions of .SPEC do not return any meaningful data.

Errors

For subfunction code 2:

INTLCK	Occurs if the implicit lock overlaps any current explicit lock. For example, if you read blocks 1 and 2 into a 1024-byte buffer in update mode, an implicit lock exists on blocks 1 and 2. If you explicitly locked these blocks, and then read blocks 2 and 3 and tried to explicitly lock blocks 2 and 3, you would get this error. An exact match is legal (for example, if the second read also read blocks 1 and 2) and results in a no operation.
NOBUFS	Occurs if the monitor needs to expand its internal table space but memory is not available.
NOROOM	There are already seven explicit locks on this channel.
PRVIOL	There is no current implicit lock; that is, no blocks have been read.

For subfunction code 3:

NOSUCH	The block number specified does not correspond to the first block number of an explicit lock.

For subfunction code -1:

DATERR	The data in the buffer does not match the data on the disk.

.SPEC for Flexible Diskette

For RX02 flexible diskette devices, the .SPEC directive lets you:

- o Obtain the density (single or double) of the current flexible diskette
- o Mount a new flexible diskette and recompute the density
- o Reformat an RX02 flexible diskette for a desired density

Because the RX02 flexible diskette drive supports single and double density flexible diskettes, the .SPEC function is especially useful for programmed flexible diskette operations. For example, .SPEC allows you to mount a series of single and double density flexible diskettes without having to close and reopen the device for each mount. Normally, the driver computes density once: during the initial open. If you insert a second flexible diskette that is incompatible with the initially computed density, a read or write operation fails. .SPEC lets you include an instruction in your program that causes the driver to recompute the density. In addition, for RX02 flexible diskette drives, .SPEC permits you to specify a density reformat operation.

.SPEC can require as much as 20 seconds to reformat the density of the RX02 flexible diskette and cannot be interrupted with CTRL/C. Note that if the operation is interrupted (by power failure or catastrophic error), the flexible diskette is rendered unusable. That is, the flexible diskette contains both single and double density. To recover, you must reformat the flexible diskette.

Privileges Required

None

.SPEC for Line Printer

Privileges Required

None

Data Passed (Line Counter Subfunction)

```
                              XRB
Mne-   Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
          +--------------------+--------------------+
        1 |           = 0 for line counter          |  0 XRLEN
          +--------------------+--------------------+
        3 |/////////////////////////////////////////|  2 XRBC
          +--------------------+--------------------+
        5 |/////////////////////////////////////////|  4 XRLOC
          +--------------------+--------------------+
XRBLKM  7 |    LPAHND (= 6)    | channel number * 2 |  6 XRCI
          +--------------------+--------------------+
       11 |/////////////////////////////////////////| 10 XRBLK
          +--------------------+--------------------+
       13 |/////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |/////////////////////////////////////////| 14
          +--------------------+--------------------+
```

XRB+XRLEN	The special line printer function to be performed.
XRB+XRCI	Channel number times two; defines the channel on which the line printer is currently open.
XRB+XRBLKM	Handler index for line printer: LPAHND (octal value equals 6).

Data Returned (Line Counter Subfunction)

```
                              XRB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
           +---------------------+---------------------+
        1  |///////////////////////////////////////////|  0
           +---------------------+---------------------+
        3  |      LSB of the 32-bit line counter       |  2 XRBC
           +---------------------+---------------------+
        5  |      MSB of the 32-bit line counter       |  4
           +---------------------+---------------------+
        7  |///////////////////////////////////////////|  6
           +---------------------+---------------------+
       11  |///////////////////////////////////////////| 10
           +---------------------+---------------------+
       13  |///////////////////////////////////////////| 12
           +---------------------+---------------------+
       15  |///////////////////////////////////////////| 14
           +---------------------+---------------------+
```

XRB+XRBC	The value of the 32-bit line counter. The LSB are returned in XRB+XRBC and the MSB are in XRB+XRBC+2. The counter is reset to zero when the line printer is opened.

Data Passed (Line Position Subfunction)

```
                              XRB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
           +---------------------+---------------------+
        1  |          = 1 for line position            |  0 XRLEN
           +---------------------+---------------------+
        3  |///////////////////////////////////////////|  2 XRBC
           +---------------------+---------------------+
        5  |///////////////////////////////////////////|  4 XRLOC
           +---------------------+---------------------+
XRBLKM  7  |    LPAHND (= 6)     | channel number * 2  |  6 XRCI
           +---------------------+---------------------+
       11  |///////////////////////////////////////////| 10 XRBLK
           +---------------------+---------------------+
       13  |///////////////////////////////////////////| 12
           +---------------------+---------------------+
       15  |///////////////////////////////////////////| 14
           +---------------------+---------------------+
```

XRB+XRLEN	The special line printer function to be performed.
XRB+XRCI	Channel number times two; defines the channel on which the line printer is currently open.

XRB+XRBLKM Handler index for line printer: LPAHND (octal value equals 6).

Data Returned (Line Position Subfunction)

```
                              XRB
Mne-  Octal                                        Octal   Mne-
monic Offset                                        Offset  monic
         +--------------------+--------------------+
       1 |////////////////////////////////////////////|  0
         +--------------------+--------------------+
       3 | curr horiz position | curr vert position  |  2 XRBC
         +--------------------+--------------------+
       5 |////////////////////////////////////////////|  4
         +--------------------+--------------------+
       7 |////////////////////////////////////////////|  6
         +--------------------+--------------------+
      11 |////////////////////////////////////////////| 10
         +--------------------+--------------------+
      13 |////////////////////////////////////////////| 12
         +--------------------+--------------------+
      15 |////////////////////////////////////////////| 14
         +--------------------+--------------------+
```

XRB+XRBC The current vertical line position. The value is in the range 0 (top of page) to L-1, where L is the page length.

XRB+XRBC+1 The current horizontal line position. The value is in the range 0 (left margin) to W-1, where W is the page width.

Errors (Line Printer)

BSERR An illegal channel number is specified at XRB+XRCI.

NOTOPN The channel specified at XRB+XRCI is not open.

PRVIOL Privilege violation; one of the following conditions has occurred:

- o The calling job does not own the specified printer and does not have DEVICE privilege.
- o The function code is not 0 or 1.
- o The device open on the channel specified at XRB+XRCI is not a line printer.

.SPEC for Magnetic Tape

Privileges Required

None

Data Passed (Magnetic Tape)

```
                                XRB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
             +---------------------+---------------------+
          1  |          special function code            |  0 XRLEN
             +---------------------+---------------------+
          3  |                parameter                  |  2 XRBC
             +---------------------+---------------------+
          5  |///////////////////////////////////////////|  4
             +---------------------+---------------------+
XRBLKM    7  |  MTAHND (=octal 16) |  channel number * 2 |  6 XRCI
             +---------------------+---------------------+
         11  |///////////////////////////////////////////| 10
             +---------------------+---------------------+
         13  |///////////////////////////////////////////| 12
             +---------------------+---------------------+
         15  |///////////////////////////////////////////| 14
             +---------------------+---------------------+
```

XRB+XRLEN	The special magnetic tape function to be performed (see Table 3-6). For a detailed discussion of these functions, see the discussion of the MAGTAPE function in the *RSTS/E Programming Manual.*
XRB+XRBC	The meaning of this word varies according to the special function code specified at XRB+XRLEN. Table 3-6 summarizes these values; for a detailed discussion, see the MAGTAPE function description in the *RSTS/E Programming Manual.*
XRB+XRCI	Channel number times two; defines the channel on which the tape is currently open.
XRB+XRBLKM	Handler index for magnetic tape: MTAHND (octal value equals 16).

Table 3-6: Special Functions for Magnetic Tape

Code	Action	Parameter	Value Returned
0	Rewind and off-line	Unused	0
1	Write end-of-file	Unused	0
2	Rewind	Unused	0
3	Skip record	# records to skip	# records not skipped
4	Backspace over record	# records to backspace	# records not backspaced
5	Set density and parity	D+P+S *	0
6	Tape status function	Unused	(see Table 3-7)
7	File characteristics	Unused	(see Table 3-7)
10	Rewind on close	Unused	0
11	End of volume	Unused	0
12	Error condition acknowledged	Unused	0

* Parameter word for function code 5:

D =	Density 14 = 800. bpi 400 = 1600. bpi, phase-encoded
P =	Parity (0 = odd, 1 = even) ** (800. bpi only)
S =	Stay 0 = MODE value specified in OPEN does not stay on CLOSE 20000 = MODE value specified in OPEN is retained after CLOSE

** DIGITAL recommends that you use odd parity. When you use even parity, you cannot write binary data. In addition, many operating systems and tape drives do not support even parity.

Data Returned (Magnetic Tape)

```
                              XRB
Mne-  Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////|  0
          +-------------------+-------------------+
        3 |         value (see Table 3-7)         |  2 XRBC
          +-------------------+-------------------+
        5 |///////////////////////////////////////|  4
          +-------------------+-------------------+
        7 |///////////////////////////////////////|  6
          +-------------------+-------------------+
       11 |///////////////////////////////////////| 10
          +-------------------+-------------------+
       13 |///////////////////////////////////////| 12
          +-------------------+-------------------+
       15 |///////////////////////////////////////| 14
          +-------------------+-------------------+
```

XRB+XRBC	The meaning of this word varies according to the value at XRB+XRLEN in the data passed. Table 3-7 summarizes these values. For a detailed discussion, see the MAGTAPE function description in the *RSTS/E Programming Manual*.

Table 3-7: Value Returned by .SPEC for Magnetic Tape

Bit	Octal Value	Meaning
Value Returned at XRB+XRBC for Function Code 6 (Magnetic Tape Status Word)		
15	100000	Last command caused an error.
14-13	If bit 3 = 0, these bits indicate density:	
	00000	Reserved
	20000	Reserved
	40000	800. bpi
	60000	Reserved
	If bit 3 = 1, these bits indicate density:	
	00000	1600. bpi
	20000	Reserved
	40000	Reserved
	60000	Reserved
12	00000	9-track tape
	10000	Reserved
11	0000	Odd parity
	4000	Even parity
10	2000	Magnetic tape is physically write-locked
9	1000	Tape is beyond end-of-tape (EOT) marker
8	400	Tape is at beginning-of-tape (BOT)
7	200	Last command detected an end-of-file (EOF)
6	100	The last command was .READ and the record read was longer than the I/O buffer size; that is, part of the record was lost
5	40	Unit is nonselectable (off-line)
4	00	Unit does not accept 1600. bpi
	20	Unit accepts 1600. bpi
3	00	See values for bits 14-13
	10	See values for bits 14-13

Table 3-7: Value Returned by .SPEC for Magnetic Tape (Cont.)

Bit	Octal Value	Meaning
Value Returned at XRB+XRBC for Function Code 6 (Magnetic Tape Status Word) (Cont.)		
2-0		Indicates last command issued: 0 = Off-line 1 = Read 2 = Write 3 = Write EOF 4 = Rewind 5 = Skip record 6 = Backspace record
Value Returned at XRB+XRBC for Function Code 7 (File Characteristics Word)		
word = 0, DOS format or ANSI U (undefined) format		
word ≠ 0, ANSI format, with bit meanings:		
15-14	40000	F (fixed-length)
	100000	D (variable-length)
	140000	S (spanned)*
13-12	00000	Carriage control embedded 'M'
	10000	FORTRAN carriage control 'A'
	20000	Implied LF/CR before record ' '
11-0	---	For Format F, this value is the record length, in bytes. For Format D, this value is the maximum record length, in bytes.

* ANSI format S is not supported by RSTS/E systems.

.SPEC for Pseudo Keyboards

For pseudo keyboards, the .SPEC function lets you:

- o Disable and enable echo at the controlled job's keyboard (that is, the KB side of the pseudo keyboard).
- o Read a flag word that tells you whether echo is ON or OFF at the controlled job's keyboard.
- o Determine the current exit status of the controlled job.

A pseudo keyboard receives two kinds of output from a controlled job: character echo, which is done by the RSTS/E monitor, and program output, which occurs when a program writes to the controlled job's keyboard. The .SPEC function affects only character echo, not program output.

Character echo is enabled by default. However, in some pseudo keyboard applications, it is more convenient to disable character echo. For example, in a pseudo keyboard application that uses both a terminal and a pseudo keyboard, you get character echo from the terminal. You also get character echo and program output from the pseudo keyboard. You can use this function to disable character echo at the pseudo keyboard.

Privileges Required

None

Data Passed

```
                                 XRB
Mne-  Octal                                         Octal   Mne-
monic Offset                                        Offset  monic
           +---------------------+---------------------+
        1  |          special function code            |  0 XRLOC
           +---------------------+---------------------+
        3  |                parameter                  |  2 XRBC
           +---------------------+---------------------+
        5  |///////////////////////////////////////////|  4
           +---------------------+---------------------+
XRBLKM  7  |    PKHND (= 20)     | channel number * 2  |  6 XRCI
           +---------------------+---------------------+
       11  |///////////////////////////////////////////| 10
           +---------------------+---------------------+
       13  |///////////////////////////////////////////| 12
           +---------------------+---------------------+
       15  |///////////////////////////////////////////| 14
           +---------------------+---------------------+
```

XRB+XRLOC	The special function codes; the possible values are:

Code	Meaning
0	Enable/disable echo or read flag.
1	Read exit status.

XRB+XRBC	The action to be performed for the special function code 0. If:

Value	Meaning
0	Read the flag word
255	Enable echo
-1	Disable echo

XRB+XRCI	Channel number times two; defines the channel on which the pseudo keyboard is open.
XRB+XRBLKM	Handler index for pseudo keyboard PKHND (octal value = 20).

Data Returned

```
                                XRB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
          +--------------------+--------------------+
        1 |////////////////////////////////////////////| 0
          +--------------------+--------------------+
        3 |         flag word or exit status         | 2 XRBC
          +--------------------+--------------------+
        5 |////////////////////////////////////////////| 4
          +--------------------+--------------------+
        7 |////////////////////////////////////////////| 6
          +--------------------+--------------------+
       11 |////////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |////////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |////////////////////////////////////////////| 14
          +--------------------+--------------------+
```

XRB+XRBC	For special function code 0: If bit 5 = 0 Keyboard echo is enabled. = 1 Keyboard echo is disabled. For special function code 1, the current exit status or the worst exit status for the job you are controlling. The current exit status value is returned in bits 0-2 and the worst exit status value is returned in bits 4-6. Both values have the following meanings:

0	Warning
1	Success
2	Error
4	Severe error

Errors

No errors are possible with the pseudo keyboard subfunction of .SPEC.

.SPEC for Terminal

The .SPEC directive for terminals has two forms. The first form lets you perform several different functions, such as cancel CTRL/O, set modes for tape, echo, and ODT, and cancel type-ahead. The second form lets you set, read, and clear private delimiters.

Privileges Required

SEND to broadcast a message to a terminal. SYSIO to force a command to a terminal.

Data Passed (Terminal - Except Private Delimiters)

```
                              XRB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
           +---------------------+---------------------+
         1 |        special function code              |  0 XRLEN
           +---------------------+---------------------+
         3 |KB number or number of bytes to send/force |  2 XRBC
           +---------------------+---------------------+
         5 | starting address of bytes to send/force   |  4 XRLOC
           +---------------------+---------------------+
XRBLKM   7 | TTYHND (=octal 2)   | channel number * 2  |  6 XRCI
           +---------------------+---------------------+
        11 |    KB number (XRB+XRLEN = 5, 6, or 12)    | 10 XRBLK
           +---------------------+---------------------+
        13 |///////////////////////////////////////////| 12
           +---------------------+---------------------+
        15 |///////////////////////////////////////////| 14
           +---------------------+---------------------+
```

XRB+XRLEN   Defines a special function:

Code	Meaning
0	Cancel CTRL/O (see the .TTRST directive)
1	Set tape mode (see the .TTAPE directive)
2	Enable echo and clear tape mode (see the .TTECH directive)
3	Disable echo (see the .TTNCH directive)
4	Set ODT mode (see the .TTDDT directive)
5	Force to keyboard (SYSIO privilege required)
6	Broadcast to keyboard (SEND privilege required)
7	Cancel all type-ahead
12	Read CTRL/C flag

For XRB+XRLEN = 12, if the caller owns the terminal being checked, the CTRL/C flag is cleared after it is read. Otherwise, the flag is just read.

XRB+XRBC	For XRB+XRLEN = 0, 1, 2, 3, 4, 7, or 12: When XRB+XRBC equals 0, these functions take place on the terminal currently open for this job. When XRB+XRBC does not equal 0, these functions take place on the keyboard number specified in XRB+XRBC. This keyboard must be assigned to but not opened by the calling job. For XRB+XRLEN = 5 or 6: XRB+XRBC is the number of bytes to send or force.
XRB+XRLOC	For XRB+XRLEN equal to 5 or 6, this word contains the starting address of the bytes to be sent or forced.
XRB+XRCI	Channel number times two; defines the channel for the terminal specified at XRB+XRBC.
XRB+XRBLKM	Handler index for terminals; TTYHND (octal value = 2).
XRB+XRBLK	For XRB+XRLEN = 5, 6, or 12, this word contains the keyboard number to which the data is to be sent or forced.

Data Returned (Terminal -- Except Private Delimiters)

```
                            XRB
Mne- Octal                                        Octal   Mne-
monic Offset                                      Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////|     CTRL/C flag     |  0 XRLEN
          +---------------------+---------------------+
        3 |        number of bytes not sent           |  2 XRBC
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
```

XRB+XRLEN	A value of zero indicates the CTRL/C flag was not set. A value of 377 indicates the CTRL/C flag was set. This value is only meaningful when the data passed in XRB+XRLEN was 12.

XRB+XRBC	Number of bytes that could not be sent (returned only when XRB+XRLEN in the data passed was six).

Errors (Terminal - Except Private Delimiter)

BSERR	An illegal channel number is specified at XRB+XRCI.
NOTOPN	The channel specified at XRB+XRCI is not open.
PRVIOL	Privilege violation; one of the following conditions has occurred:

- o The calling job does not own the specified keyboard and does not have DEVICE privilege
- o The function code is not 0 to 7 or 11 (returned for all .SPEC calls for terminals)
- o An illegal terminal number at XRB+XRBLK
- o The device open on the channel at XRB+XRCI is not a terminal

Private Delimiters

A private delimiter is a character used as a delimiter within a program. You can define any printing or nonprinting character to be a private delimiter, for example:

- o A letter
- o A function key such as DELETE
- o A control character such as CTRL/Z
- o A standard delimiter such as LINE FEED

A private delimiter is useful on a data entry terminal with a specialized keyboard. You can use a large or conveniently located key as the delimiter key. Private delimiters are also useful in keypad applications.

You can declare one character as a private delimiter on any RSTS/E system. In addition, an installation option allows the use of multiple private delimiters (see the *RSTS/E System Installation and Update Guide*). If your system has this feature, you can declare up to 256 private delimiters.

Multiple private delimiters let you do special character processing without using single character I/O. For example, by combining escape sequences with private delimiters, you can define your own function keys in keypad applications.

The rest of this section:

- o Provides general information about private delimiters
- o Shows the XRB layouts for setting, reading, and clearing private delimiters
- o Lists all private delimiter masks

Characteristics of Private Delimiters

Declaring a character as a private delimiter with the .SPEC directive overrides the existing interpretation for the character. Thus, unlike a standard delimiter such as RETURN or LINE FEED, a private delimiter does not echo at the terminal. In addition, a special character no longer performs its normal function. For example, when the DELETE key is a private delimiter, it does not erase the last character typed.

A private delimiter has basically the same characteristics as a standard delimiter. Like a standard delimiter, it:

- o Terminates a .READ on the terminal.
- o Cannot be deleted except by CTRL/X. The DELETE key and CTRL/U do not affect private delimiters in the type-ahead buffer. (CTRL/X cancels all type-ahead buffering.)
- o Causes the system to awaken a sleeping job when typed at a terminal that the job has open or assigned.

Once set, a private delimiter remains in effect for a terminal until one of the following occurs:

- o The program clears it
- o The job releases the terminal by deassigning it or by closing the I/O channel where the terminal is open
- o The job terminates

In addition, the system clears private delimiters when a dial-up line gets hung up or the job controlling the terminal is killed.

Private delimiters change the way characters are processed in binary mode (MODE 1). When a terminal is open in binary mode and no private delimiter is in use, the system terminates a read after every character. However, if one or more private delimiters are in use, the system terminates a read only when a private delimiter is typed.

The system processes private delimiters after processing CTRL/S and CTRL/Q (if the TTSYNC characteristic is set) and escape sequences (if the terminal is in escape sequence mode). This feature prevents a terminal from becoming permanently stalled, and it also lets you use private delimiters and escape sequences in the same program.

The system processes private delimiters before all other characters, including control characters such as CTRL/C. Thus, when you use a standard delimiter character as a private delimiter, it does not echo on the terminal.

Programming Hint

By combining escape sequences with private delimiters, you can define your own function keys without using single character I/O. For example:

1. Make sure the keypad is in the right mode for your application

2. Define each function as the PF1 key followed by a character

3. Define each character as a private delimiter so it does not echo on the terminal

For example, you might define PF1 + A as one function and PF1 + M as another function.

Data Passed (Terminal - Private Delimiters)

```
                              XRB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +---------------------+---------------------+
        1 |     = octal 11 for private delimiters      |  0 XRLEN
          +---------------------+---------------------+
        3 |  0 or byte count when XRB+XRMOD = 1 or 2   |  2 XRBC
          +---------------------+---------------------+
        5 |    0 or address when XRB+XRMOD = 1 or 2    |  4 XRLOC
          +---------------------+---------------------+
XRBLKM  7 |  TTYHND (=octal 2)  |  channel number * 2  |  6 XRCI
          +---------------------+---------------------+
       11 |      flag byte      |      KB number       | 10 XRBLK
          +---------------------+---------------------+
       13 |                  must = 0                  | 12 XRTIME
          +---------------------+---------------------+
       15 |       subfunction code = 0, 1, or 2        | 14 XRMOD
          +---------------------+---------------------+
```

XRB+XRLEN Function code; set to octal 11 for private delimiter mask.

XRB+XRBC For XRB+XRMOD equals zero, this word must be set to zero.

For XRB+XRMOD equals one (set private delimiter mask), this word is the byte count for the private delimiter bit mask.

For XRB+XRMOD equals two (read private delimiter mask), this word is the buffer length for a buffer into which the mask is to be read.

For XRB+XRMOD equals one or two, the value of XRB+XRBC must be less than or equal to 32.

XRB+XRLOC For XRB+XRMOD equals zero, this word must be set to zero.

For XRB+XRMOD equals one, this word is the address of the private delimiter mask. The mask itself can be up to 32 bytes long (32 bytes = 256 bits). Each bit in the mask represents an ASCII character (see Table 3-8). Setting a bit indicates that the associated ASCII character is to serve as a private delimiter.

For XRB+XRMOD equals two, this word is the address of a buffer into which the terminal's private delimiter mask is to be read.

Table 3-8: Private Delimiter Masks

Octal Value	ASCII Character	Byte, address+n	Bit	Byte Value Octal	Byte Value Decimal
0*	NUL	n=0	0	1	1
1	SOH	0	1	2	2
2	STX	0	2	4	4
3	ETX(CTRL/C)	0	3	10	8
4	EOT	0	4	20	16
5	ENQ	0	5	40	32
6	ACK	0	6	100	64
7	BEL	0	7	200	128

Table 3-8: Private Delimiter Masks (Cont.)

Octal Value	ASCII Character	Byte, address+n	Bit	Byte Value Octal	Byte Value Decimal
10	BS(backspace)	1	0	1	1
11	HT(horizontal tab)	1	1	2	2
12	LF(line feed)	1	2	4	4
13	VT(vertical tab)	1	3	10	8
14	FF(form feed)	1	4	20	16
15	CR(carriage return)	1	5	40	32
16	SO	1	6	100	64
17	SI(CTRL/O)	1	7	200	128
20	DLE	2	0	1	1
21	DC1(CTRL/Q)	2	1	2	2
22	DC2	2	2	4	4
23	DC3(CTRL/S)	2	3	10	8
24	DC4	2	4	20	16
25	NAK(CTRL/U)	2	5	40	32
26	SYN	2	6	100	64
27	ETB	2	7	200	128
30	CAN	3	0	1	1
31	EM	3	1	2	2
32	SUB(CTRL/Z)	3	2	4	4
33	ESC(ESCAPE)	3	3	10	8
34	FS	3	4	20	16
35	GS	3	5	40	32

Table 3-8: Private Delimiter Masks (Cont.)

Octal Value	ASCII Character	Byte, address+n	Bit	Byte Value Octal	Byte Value Decimal
36	RS	3	6	100	64
37	US	3	7	200	128
40	SP(space)	4	0	1	1
41	!	4	1	2	2
42	"	4	2	4	4
43	#	4	3	10	8
44	$	4	4	20	16
45	%	4	5	40	32
46	&	4	6	100	64
47	'(apostrophe)	4	7	200	128
50	(	5	0	1	1
51	)	5	1	2	2
52	*	5	2	4	4
53	+	5	3	10	8
54	,(comma)	5	4	20	16
55	-(dash/minus)	5	5	40	32
56	.(period)	5	6	100	64
57	/	5	7	200	128
60	0	6	0	1	1
61	1	6	1	2	2
62	2	6	2	4	4
63	3	6	3	10	8

Table 3-8: Private Delimiter Masks (Cont.)

Octal Value	ASCII Character	Byte, address+n	Bit	Byte Value Octal	Byte Value Decimal
64	4	6	4	20	16
65	5	6	5	40	32
66	6	6	6	100	64
67	7	6	7	200	128
70	8	7	0	1	1
71	9	7	1	2	2
72	:	7	2	4	4
73	;	7	3	10	8
74	<	7	4	20	16
75	=	7	5	40	32
76	>	7	6	100	64
77	?	7	7	200	128
100	@	10(8)	0	1	1
101	A	10(8)	1	2	2
102	B	10(8)	2	4	4
103	C	10(8)	3	10	8
104	D	10(8)	4	20	16
105	E	10(8)	5	40	32
106	F	10(8)	6	100	64
107	G	10(8)	7	200	128
110	H	11(9)	0	1	1
111	I	11(9)	1	2	2

Table 3-8: Private Delimiter Masks (Cont.)

Octal Value	ASCII Character	Byte, address+n	Bit	Byte Value Octal	Byte Value Decimal
112	J	11(9)	2	4	4
113	K	11(9)	3	10	8
114	L	11(9)	4	20	16
115	M	11(9)	5	40	32
116	N	11(9)	6	100	64
117	O	11(9)	7	200	128
120	P	12(10)	0	1	1
121	Q	12(10)	1	2	2
122	R	12(10)	2	4	4
123	S	12(10)	3	10	8
124	T	12(10)	4	20	16
125	U	12(10)	5	40	32
126	V	12(10)	6	100	64
127	W	12(10)	7	200	128
130	X	13(11)	0	1	1
131	Y	13(11)	1	2	2
132	Z	13(11)	2	4	4
133	[	13(11)	3	10	8
134	\(backslash)	13(11)	4	20	16
135	]	13(11)	5	40	32
136	^ or *	13(11)	6	100	64
137	_ or ⟵	13(11)	7	200	128

Table 3-8: Private Delimiter Masks (Cont.)

Octal Value	ASCII Character	Byte, address+n	Bit	Byte Value	
				Octal	Decimal
140	`(grave accent)	14(12)	0	1	1
141	a	14(12)	1	2	2
142	b	14(12)	2	4	4
143	c	14(12)	3	10	8
144	d	14(12)	4	20	16
145	e	14(12)	5	40	32
146	f	14(12)	6	100	64
147	g	14(12)	7	200	128
150	h	15(13)	0	1	1
151	i	15(13)	1	2	2
152	j	15(13)	2	4	4
153	k	15(13)	3	10	8
154	l	15(13)	4	20	16
155	m	15(13)	5	40	32
156	n	15(13)	6	100	64
157	o	15(13)	7	200	128
160	p	16(14)	0	1	1
161	q	16(14)	1	2	2
162	r	16(14)	2	4	4
163	s	16(14)	3	10	8
164	t	16(14)	4	20	16
165	u	16(14)	5	40	32

Table 3-8: Private Delimiter Masks (Cont.)

Octal Value	ASCII Character	Byte, address+n	Bit	Byte Value	
				Octal	Decimal
166	v	16(14)	6	100	64
167	w	16(14)	7	200	128
170	x	17(15)	0	1	1
171	y	17(15)	1	2	2
172	z	17(15)	2	4	4
173	{	17(15)	3	10	8
174	|(vertical)	17(15)	4	20	16
175	}	17(15)	5	40	32
176	~(tilde)	17(15)	6	100	64
177	DEL(RUBOUT)	17(15)	7	200	128
200-377	Can only be used for terminals with the /EIGHT_BIT characteristic set or open in binary mode; the 8th bit is cleared in normal mode for terminals with the /NOEIGHT_BIT characteristic.				

* Octal code 0 can be used only for terminals opened in binary mode because nulls are ignored in normal mode.

XRB+XRCI	Channel number times two. When this byte equals zero, it indicates the job's keyboard. When this byte does not equal zero, a keyboard must be open on the indicated channel.
XRB+XRBLKM	Handler index for terminals; TTYHND (octal value is 2).
XRB+XRBLK	Specifies the keyboard number that the subfunction is to take place on. If this byte and the byte at XRB+11 equal zero, then the subfunction is performed on the terminal open on the channel indicated by XRB+XRCI.
XRB+11	Flag byte. If bit 7 of this byte is set (byte equals 128), it indicates that the keyboard number at

XRB+XRBLK is real. (When XRB+XRBLK is zero, this bit set indicates that KB0: is the desired terminal. If this bit is cleared and XRB+XRBLK is zero, it indicates that the keyboard open on the channel indicated at XRB+XRCI is the desired terminal.) All other bits must equal zero.

XRB+XRTIME	This word must be set to zero.
XRB+XRMOD	Subfunction code:

0 = Clear private delimiter mask.
1 = Set private delimiter mask.
2 = Read private delimiter mask.

Once set, private delimiters remain in effect for a terminal until cleared (XRB+XRMOD = 0) or until implicitly cleared by:

- o Deassigning or closing the terminal
- o Killing the job or hanging up the line
- o Keyboard monitor read (negative wait time on .READ)

Note that clearing the delimiter mask to all zeros is not the same as issuing the clear subfunction (XRB+XRMOD = 0). Use the clear subfunction to free system resources used when any mask is set.

Data Returned (Terminal - Private Delimiters)

Except for a possible error in byte 0 of the FIRQB, the private delimiter subfunctions of .SPEC do not return any meaningful data.

Errors (Terminal - Private Delimiters)

BADCNT One of the following conditions has occurred:

- o Byte count at XRB+XRBC does not equal zero or is greater than 32
- o Invalid address at XRB+XRLOC

BSERR An illegal channel number is specified at XRB+XRCI.

ERRERR Multiple delimiter feature is not included in the monitor.

NOBUFS The monitor needs to expand its internal buffer space, but memory is not available; a retry may succeed.

NOSUCH	You are trying to read the private delimiter mask, but no private delimiters are set.
NOTOPN	The channel specified at XRB+XRCI is not open.
PRVIOL	Privilege violation; one of the following conditions has occurred: o Job does not own the specified keyboard and does not have the required privilege o Subfunction code at XRB+XRMOD is not 0, 1, or 2 o Function code is not 0 through 7 or 11 (returned for all .SPEC calls for terminals) o An illegal terminal number is specified at XRB+XRBLK o The device open on the channel specified at XRB+XRCI is not a terminal

.STAT - Return Job Statistics

Form

```
.STAT
```

Function

The .STAT directive returns current statistics on the job to the XRB.

Privileges Required

TUNE is required to read the current run priority or run burst settings.

Data Passed

The .STAT directive does not pass any meaningful data.

Data Returned

XRB

Mne-monic	Octal Offset		Octal Offset	Mne-monic
	1	current job image size, in K words	0	XRLEN
	3	current run-time system size, in K words	2	XRBC
	5	current private memory max, in K words	4	XRLOC
	7	maximum job image size, in K words	6	XRCI
	11	current PPN	10	XRBLK
	13	current run priority	12	XRTIME
	15	current run burst	14	XRMOD

XRB+XRLEN	The current size of the user job image for this job, in K words. If I&D space is enabled, this is the size of only the D-space of the task.
XRB+XRBC	The size of the current run-time system for this job, in K words.
XRB+XRLOC	The current private memory maximum for the user job image, in K words. If the job has an unlimited memory maximum or if its private memory maximum is larger than the possible maximum size allowed by its current run-time system, then the value returned here is the

	maximum size possible for the current run-time system, in K words. If the job's private maximum is less than the run-time system minimum, then the value returned here is the run-time system's minimum size (see the .CORE directive).
	In all cases, this value represents the maximum size of the user job image under its current run-time system.
XRB+XRCI	The maximum job image size possible (under the current run-time system), in K words. If I&D space is enabled, this is the maximum size to which D-space can expand.
XRB+XRBLK	The job's current PPN is returned here. The programmer number is returned as a binary value in the low byte (XRB+XRBLK), the project number as a binary value in the high byte (XRB+11). If the job is not logged in, a value of 0 is returned here.
XRB+XRTIME	The job's current run priority (if caller has TUNE privilege; otherwise, .STAT returns zero). Run priority may range from -128, indicating a suspended job, to +127, the highest priority. The monitor schedules jobs for time-shared execution according to this priority. When a user first logs in to RSTS/E, LOGIN is run with priority 0. LOGIN sets the user's job run priority to -8. Only in unusual cases should the run priority be changed. It can be changed by a user with TUNE privilege for any job with the SET JOB/PRIORITY command. It can also be changed with the UU.PRI subfunction of the .UUO directive. It can be modified by the job by one-half step (4) with the .SET and .CLEAR directives.
	The special-case value of -128 indicates that the job is never scheduled to run; it is suspended.
XRB+XRMOD	The job's current run burst (if caller has TUNE privilege; otherwise, .STAT returns zero). The run burst is the amount of time that the job is allowed to execute compute-bound before the next job in the schedule is given control.
	The units of run burst are 1/60ths or 1/50ths of a second, depending on the clock in use and/or the line frequency. (Systems running with the KW11-P clock at crystal speeds, rather than at line frequency, have a run burst unit of 1/50th of a second. Otherwise, if the system is operating with a 60 Hz power line, one run burst unit equals 1/60th of a second.) The range of values for run burst is from 1 to 127.

When a job is created, the monitor sets the run burst to a value of six. This value can be modified by the system manager for a particular job with the SET JOB/RUN_BURST command. It can also be modified with the UU.PRI subfunction of the .UUO directive.

Errors

No errors are possible on this directive.

Example

Since the .STAT directive does not pass any data to the monitor, the call is simply:

```
  .STAT
```

.TIME -- Return Timing Information

Form

```
  .TIME
```

Function

The .TIME directive returns job timing information: elapsed CPU time, elapsed time connected to a user terminal (channel 0), elapsed device time, and memory utilization.

Privileges Required

None.

Data Passed

The .TIME directive does not pass any meaningful data.

Data Returned

```
                                     XRB
Mne-  Octal                                              Octal   Mne-
monic Offset                                              Offset  monic
          +--------------------+--------------------+
        1 | low 16 bits of elapsed CPU time, .1 sec.  |  0 XRLEN
          +--------------------+--------------------+
        3 |  elapsed connect time to channel 0, min.  |  2 XRBC
          +--------------------+--------------------+
        5 |  low 16 bits of memory utilization, KCTs  |  4 XRLOC
          +--------------------+--------------------+
        7 |        elapsed device time, minutes       |  6 XRCI
          +--------------------+--------------------+
       11 | high 16 bits of elapsed CPU time, .1 sec. | 10 XRBLK
          +--------------------+--------------------+
       13 |///////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 | high 16 bits of memory utilization, KCTs  | 14 XRMOD
          +--------------------+--------------------+
```

XRB+XRLEN	The low-order 16 bits of the job's elapsed CPU time, in tenths of a second of CPU utilization.
XRB+XRBC	The elapsed time that the job has been connected to a channel 0 terminal, in minutes.
XRB+XRLOC	The low-order 16 bits of the job's memory utilization, in kilo-core-ticks (KCTs). A kilo-core-tick is the use of 1K of memory for one-tenth of a second.

XRB+XRCI	The job's elapsed device time. Device time is the use of an allocatable device for one minute. If a job owns two devices for one elapsed minute, then two units of device time are accrued.
XRB+XRBLK	The high-order 16 bits of the job's elapsed CPU time (see XRB+XRLEN).
XRB+XRMOD	The high-order 16 bits of the job's memory utilization (see XRB+XRLOC).

Errors

No errors are possible with the .TIME directive.

Example

Since the .TIME directive does not pass any data, the call is simply:

```
  .TIME
```

.TTAPE -- Enter Tape Mode

Form

```
.TTAPE
```

Function

The .TTAPE directive enters tape mode on the job's terminal (channel 0). Use this directive when it is necessary to read data from the low-speed paper tape reader available on some terminals. Make sure the terminal is open on channel 0, and call this directive before the .READ directive. Three things happen:

- o No incoming characters are echoed, preventing needless output at the terminal while the tape is being read.
- o The DELETE character (ASCII code 177) is ignored, rather than deleting the previous character.
- o A LINE FEED character (ASCII code 012) is not automatically appended to an incoming RETURN (ASCII code 015).

The .TTECH directive returns character processing to normal on channel 0.

Privileges Required

None

Data Passed

The .TTAPE directive does not pass any data.

Data Returned

The .TTAPE directive does not return any data.

Errors

DETKEY	Channel 0 is not currently available for this job; it is running detached.

Example

Since the .TTAPE directive does not pass or return any data, the call is simply:

```
.TTAPE
```

.TTDDT -- Disable Full-Line Buffering

Form

```
.TTDDT
```

Function

The .TTDDT directive disables the monitor's usual practice of buffering a full line of data from the user's terminal (channel 0) before passing it on to the job on a .READ directive.

The monitor accepts data typed at a user terminal and stores it in a buffer until the job associated with the terminal reads the data. Normally, a .READ causes the monitor to pass a line to the job's buffer. (A line is any number of characters ending with a RETURN, LINE FEED, ESCAPE, FORM FEED, or CTRL/D combination.) If a full line is not in the monitor's buffer, the monitor stalls the job until it gets a delimiter and then awakens the job and passes the line on to the job's buffer.

The .TTDDT directive tells the monitor that, when the next .READ on the user's terminal is issued, it is to pass on whatever is currently in the monitor's buffer, whether or not a delimiter has been typed. If no characters are in the monitor's buffer, the job is stalled until at least one character has been typed.

.TTDDT is a one-shot directive: it affects only the next .READ on the user's terminal. If you want to do this type of input consistently, you must execute a .TTDDT before each .READ.

This type of input is useful when you want to respond to each character that a user types. (Note that more than one character may be in the monitor's buffer. If you really want only one character, use .TTDDT before each .READ, and define a 1-character input buffer for the .READ.) For example, the ODT utility (see the *ODT Reference Manual*) uses this capability to accept commands without requiring that you type a delimiter. This type of input puts a high load on the system; DIGITAL does not recommend its use except in unusual circumstances.

Privileges Required

None

Data Passed

The .TTDDT directive does not pass any data.

Data Returned

The .TTDDT directive does not return any data.

Errors

DETKEY	Channel 0 is not currently available for this job; the job is running detached.

Example

Since the .TTDDT directive does not pass any data, the call is simply:

```
  .TTDDT
```

.TTECH -- Undo .TTAPE or .TTNCH

Form

```
.TTECH
```

Function

The .TTECH directive causes the monitor to resume normal character input processing on channel 0 when it has been disabled with either a .TTAPE directive or a .TTNCH directive.

Privileges Required

None

Data Passed

The .TTECH directive does not pass any data.

Data Returned

The .TTECH directive does not return any data.

Errors

DETKEY	Channel 0 is not available for this job; the job is running detached.

Example

Since the .TTECH directive does not pass any data, this call is simply:

```
.TTECH
```

.TTNCH -- Stop Echo

Form

```
.TTNCH
```

Function

The .TTNCH directive disables terminal echo on the job's terminal (channel 0). That is, whatever the user types is accepted, but it is not echoed back for display on the terminal. Otherwise, all normal character processing occurs. (The .TTECH directive returns character processing to normal on channel 0.)

Privileges Required

None

Data Passed

The .TTNCH directive does not pass any data.

Data Returned

The .TTNCH directive does not return any data.

Errors

DETKEY	Channel 0 is not available for this job; the job is running detached.

Example

Since the .TTNCH directive does not pass any data, the call is simply:

```
.TTNCH
```

.TTRST - Restart Output

Form

```
.TTRST
```

Function

The .TTRST directive restarts program output to the user's terminal when such output has been stopped by the user's typing a CTRL/O or CTRL/C.

Terminal service in the RSTS/E monitor maintains a "discard all program output" indicator for each terminal. When this indicator is set, the driver ignores a .WRITE directive to the terminal rather than display the data at the user's terminal. When the indicator is clear, a .WRITE to the terminal is processed normally. The .TTRST directive clears this indicator.

The driver sets the indicator when the user enters a CTRL/C combination. It reverses the indicator when the user enters a CTRL/O combination. A .READ directive to the user's terminal clears the indicator.

One use of .TTRST is in run-time systems with keyboard monitors. By issuing .TTRST before displaying any prompt, you can ensure that the prompt is actually displayed at the user's terminal.

Privileges Required

None

Data Passed

The .TTRST directive does not pass any data.

Data Returned

The .TTRST directive does not return any data.

Errors

DETKEY	The user's terminal is not currently available for this job; the job is running detached.

Example

Since the .TTRST directive does not pass any data, the call is simply:

```
.TTRST
```

.ULOG -- Allocate/Reallocate/Deallocate Device or Assign/Deassign User Logical

Form

```
.ULOG
```

Function

The .ULOG directive has three subfunctions. You select the particular action desired by setting a function field in the FIRQB (at offset FQFUN). Table 3-9 lists the subfunctions by function code. The following sections describe the subfunctions in alphabetical order.

Table 3-9: Summary of .ULOG Subfunctions

FQFUN Value (Octal)	Mnemonic	Action Performed
12	UU.ASS	Allocate/reallocate a device or assign user logical
13	UU.DEA	Deallocate a device or deassign a user logical
14	UU.DAL	Deallocate all devices and deassign all user logicals

Privileges Required

See the descriptions of each .ULOG subfunction.

UU.ASS (Allocate/Reallocate a Device, or Assign User Logical)

Form

```
  MOVB     #UU.ASS,FIRQB+FQFUN
           .
           .
           .
   (set up FIRQB and XRB)
           .
           .
           .
  .ULOG
```

Function

The UU.ASS subfunction of .ULOG lets you do one of three things:

- o Allocate a device to a job.
- o Reallocate a device to another job.
- o Enter a user logical. This feature lets you:
  - Assign a logical name to a device. The monitor then uses this assignment for logical-to-physical device translation by the .FSS directive.
  - Associate a PPN with a particular logical name. This user logical PPN is used if the associated logical name (but no PPN) is found in a string scanned by .FSS. Use this feature if you want to override a system-wide logical name with an associated PPN. For example, the logical name LB is commonly associated with a disk on the public structure and some specific PPN (usually [1,1]). With this feature, you can set up a different device and PPN for the logical name LB.
  - Assign a PPN to be substituted for an at sign (@) character encountered in a file specification string scanned by an .FSS directive.
  - Assign a protection code to be used as a default if no protection code is specified in a file specification string translated by an .FSS directive.

Note

Assigning user logicals assigns values to the USRPPN, USRPRT, and USRLOG areas in the low 1000 bytes of virtual memory (see Chapter 2). If these values are in a nonstandard location, you must specify where they are by setting the XRB. For all other uses of .ULOG, the XRB should be cleared to zeros.

Privileges Required

No privileges are required to manipulate user logicals or deallocate a device. DEVICE is required to allocate a restricted device. HWCTL is required to seize a device or reallocate a device to a job in another account.

Data Passed -- Allocate/Reallocate Device

```
                              FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                           Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////| 0
          +-------------------+-------------------+
FQFUN   3 | UU.ASS (= octal 12) |                 | 2
          +-------------------+-------------------+
        5 |///////////////////////////////////////| 4
          +-------------------+-------------------+
        7 |              (must be 0)              | 6  FQPPN
          +-------------------+-------------------+
       11 |    (must be 0)    | 0=assign; ≠0=job  | 10 FQNAM1
          +-------------------+-------------------+
       13 |///////////////////////////////////////| 12
          +-------------------+-------------------+
       15 |  DOS or ANS in RAD50 format (1 word)  | 14 FQEXT
          +-------------------+-------------------+
       17 |///////////////////////////////////////| 16
          +-------------------+-------------------+
       21 |///////////////////////////////////////| 20
          +-------------------+-------------------+
       23 |       100001 for seize; else 0        | 22 FQMODE
          +-------------------+-------------------+
       25 |///////////////////////////////////////| 24
          +-------------------+-------------------+
       27 |              (must be 0)              | 26 FQPFLG
          +-------------------+-------------------+
       31 |    device name (2 ASCII characters)   | 30 FQDEV
          +-------------------+-------------------+
       33 | ≠0, unit no. real | device unit number | 32 FQDEVN
          +-------------------+-------------------+
       35 |///////////////////////////////////////| 34
          +-------------------+-------------------+
       37 |///////////////////////////////////////| 36
          +-------------------+-------------------+
```

FIRQB+FQFUN   The function code UU.ASS (octal value = 12).

FIRQB+FQNAM1	For allocating a device, the two bytes beginning here must equal zero. For reallocating a device, this byte is the job number to which the device is to be reallocated. The byte at FIRQB+FQNAM1+1 must equal zero. If you do not have HWCTL privilege, you can reallocate a device only to a job that is logged in to the same account as your current account.
FIRQB+FQEXT	For allocating or reallocating a magnetic tape device, this word is either DOS or ANS (in RAD50 format) to indicate DOS or ANSI label format for the magnetic tape drive.
FIRQB+FQMODE	For allocating or reallocating a device, setting this field to 100001 octal indicates a "seize" operation. That is, it allocates or reallocates the device even if it is currently allocated to another job. This feature can be issued by a caller with HWCTL privilege. Furthermore, the target device must not be open, and the current owner cannot be performing a directory lookup on that device with the UU.DIR subfunction of the .UUO directive. If you do not want a seize operation, set this word to zero.
FIRQB+FQDEV	Device name to be allocated or reallocated, specified as two ASCII characters. If this word is zero, the public disk structure is assumed.
FIRQB+FQDEVN	Device unit number, passed as a binary value in byte FIRQB+FQDEVN. A nonzero value in byte FIRQB+FQDEVN+1 indicates an explicit device unit number. A zero value in byte FIRQB+FQDEVN+1 indicates no device unit number.

Note

The XRB should be cleared to zeros for an allocate or reallocate device operation.

Data Passed - Enter User Logical

```
                                FIRQB
Mne-   Octal                                            Octal   Mne-
monic Offset                                             Offset  monic
              +-------------------+-------------------+
          1   |///////////////////////////////////////|  0
              +-------------------+-------------------+
FQFUN     3   | UU.ASS (= octal 12) |///////////////////|  2
              +-------------------+-------------------+
          5   |///////////////////////////////////////|  4
              +-------------------+-------------------+
          7   |    project number   |  programmer number  |  6  FQPPN
              +-------------------+-------------------+
         11   |                                       | 10  FQNAM1
              + user logical device name in RAD50 format +
         13   |               (2 words)               | 12
              +-------------------+-------------------+
         15   |///////////////////////////////////////| 14
              +-------------------+-------------------+
         17   |///////////////////////////////////////| 16
              +-------------------+-------------------+
         21   |///////////////////////////////////////| 20
              +-------------------+-------------------+
         23   |///////////////////////////////////////| 22
              +-------------------+-------------------+
         25   |///////////////////////////////////////| 24
              +-------------------+-------------------+
FQPROT   27   |   protection code   |255=assign prot. code| 26  FQPFLG
              +-------------------+-------------------+
         31   |      device name (2 ASCII characters)     | 30  FQDEV
              +-------------------+-------------------+
         33   |  ≠0, unit no. real  |  device unit number | 32  FQDEVN
              +-------------------+-------------------+
         35   |///////////////////////////////////////| 34
              +-------------------+-------------------+
         37   |///////////////////////////////////////| 36
              +-------------------+-------------------+
```

FIRQB+FQFUN	The function code UU.ASS (octal value = 12).
FIRQB+FQPPN	The PPN for those features that use a PPN (see the discussion under Function at the beginning of this section). For a user-assignable PPN, this is the PPN to be used to replace an @ sign in a string parsed by an .FSS directive. In this case, the two bytes at FIRQB+FQNAM1 and FIRQB+FQNAM1+1 must be zero. If the four bytes FIRQB+FQNAM1 through FIRQB+FQNAM1+3 contain a logical device name, and a PPN is to be associated with that name, specify the PPN here. If no PPN is to be associated with the logical device name,

set this word to zero.

FIRQB+FQNAM1	To assign a user logical name to a device, set the two words beginning here to the logical name, in RAD50 format. Otherwise, set these bytes to zero.
FIRQB+FQPFLG	To assign a user-assignable default protection code, set this byte to nonzero. (See FIRQB+FQPROT.)
FIRQB+FQPROT	The protection code to be used as a default if no protection code is specified in a string scanned by an .FSS directive. The byte at FIRQB+FQPFLG must be nonzero, and the two words at FIRQB+FQPPN and FIRQB+FQNAM1 must be zero.
FIRQB+FQDEV	Device name for assigning a logical name to be associated with a device. Specify the device as two ASCII characters. If this word is zero, the public disk structure is assumed.
FIRQB+FQDEVN	Device unit number for assigning a logical name to be associated with a device. The device unit number is passed as a binary value in byte FIRQB+FQDEVN. A nonzero value in byte FIRQB+FQDEVN+1 indicates an explicit device unit number. A zero value in byte FIRQB+FQDEVN+1 indicates no device unit number.

XRB

```
Mne-  Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
         +----------------------+---------------------+
       1 |     length of user logical information     |  0 XRLEN
         +----------------------+---------------------+
       3 |     length of user logical information     |  2 XRBC
         +----------------------+---------------------+
       5 |  starting address for user logical info.   |  4 XRLOC
         +----------------------+---------------------+
       7 |////////////////////////////////////////////|  6
         +----------------------+---------------------+
      11 |////////////////////////////////////////////| 10
         +----------------------+---------------------+
      13 |////////////////////////////////////////////| 12
         +----------------------+---------------------+
      15 |////////////////////////////////////////////| 14
         +----------------------+---------------------+
```

XRB+XRLEN	If the user logical information is in its standard location (USRPPN, USRPRT, and USRLOG), this word is passed as zero. If some nonstandard set of locations is being used, then specify the length of that information, in bytes, here. Thus, when the user logical information is in a nonstandard location, this

word must be at least four (for user-assignable PPN and default protection code). The space for user logical names is optional. If no space is reserved for logical names in the nonstandard location, no such assignments are allowed.

XRB+XRBC	The length of the information (same as the word at XRB+XRLEN).
XRB+XRLOC	If the word at XRB+XRLEN is nonzero, then this word defines the starting location for the user logical information (the default PPN). You must specify an even value. See Chapter 2 for a description of the order and format of the information created by .ULOG.

Data Returned

FIRQB — The error code (if any).

FIRQB+FQFIL — The previous owner of the device as follows:

- o If you own a device already and allocate it, UU.ASS returns your own job number.
- o If you seize a device, you can see who you seized it from.
- o If the value in FIRQB+FQFIL is zero, then you know that the device was not previously owned.

Errors

BADCNT — This error has two possible causes:

- o The length specified at XRB+XRLEN and XRB+XRBC must be 4 + 8n, where n = 0, 1, 2, 3, or 4. If it is not, this error is returned. (Other lengths are not valid user logicals.)
- o The value at XRB+XRLOC is odd.

BDNERR — An attempt was made to reallocate a device to a nonexistent job. This error can occur only for a reallocate call.

INUSE — For allocating or reallocating a device, the specified device is currently open or has an open file. For assigning a user logical, no space is currently available for storing the information.

NODEVC	The device name specified at FIRQB+FQDEVN is a logical device name for which a physical device is not currently allocated.
NOTAVL	The device specified exists on the system, but the operation failed for one of the following reasons: o The device is currently reserved by another job (see description of FIRQB+FQMODE). o Ownership of the device requires DEVICE privilege, which the user does not have. o The device or its controller is disabled. o The device is a keyboard line for a pseudo keyboard only.
PRVIOL	You do not have the required privileges and tried to reallocate a device to a job that is logged in to an account other than your current account.

Example

The following code allocates the user logical name OUT to the public disk structure. Assume that the FIRQB and XRB have been previously cleared to zero. For example:

```
MOVB    #UU.ASS,FIRQB+FQFUN          ; Set function code
MOV     #^ROUT,FIRQB+FQNAM1          ; Set logical name as
MOV     #^R    ,FIRQB+FQNAM1+2       ;   two words RAD50
.ULOG
```

UU.DAL - Deallocate All Devices and Deassign User Logicals

Form

```
  MOVB     #UU.DAL,FIRQB+FQFUN
               .
               .
               .
   (set up FIRQB and XRB)
               .
               .
               .
  .ULOG
```

Function

The UU.DAL function of .ULOG deallocates all devices and deassigns user logicals for the calling program.

Privileges Required

None

Data Passed

```
                               FIRQB
Mne-   Octal                                                Octal   Mne-
monic Offset                                                Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |  UU.DAL (=octal 14) |/////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFUN   The function code UU.DAL (octal value equals 14).

```
                              XRB
Mne-  Octal                                        Octal   Mne-
monic Offset                                        Offset  monic
          +--------------------+--------------------+
        1 |     length of user logical information   |  0 XRLEN
          +--------------------+--------------------+
        3 |     length of user logical information   |  2 XRBC
          +--------------------+--------------------+
        5 |  starting address for user logical info. |  4 XRLOC
          +--------------------+--------------------+
        7 |//////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |//////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |//////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |//////////////////////////////////////////| 14
          +--------------------+--------------------+
```

XRB+XRLEN	If the user logical information is in its standard location (USRPPN, USRPRT, and USRLOG), set this word to zero. If some nonstandard set of locations is being used, then specify the length (in bytes) of that information.
XRB+XRBC	The length of the information (same as the word at XRB+XRLEN).
XRB+XRLOC	If the word at XRB+XRLEN is nonzero, then this word defines the starting location for the user logical information (the default PPN). See Chapter 2 for a description of the order and format of the information created by .ULOG.

Data Returned

The UU.DAL subfunction of .ULOG does not return any meaningful data.

Errors

BADCNT   This error has two possible causes:

- o The length specified at XRB+XRLEN and XRB+XRBC must be 4+8n, where n = 0, 1, 2, 3, or 4. If it is not, this error is returned. (Other lengths are not valid user logicals.)
- o The value at XRB+XRLOC is odd.

Example

The following code deallocates all devices and deassigns user logicals for the current job. The user logical information is in its standard location. For example:

```
  MOVB     #UU.DAL,FIRQB+FQFUN
  CLR      XRB+XRLEN
  CLR      XRB+XRBC
  CLR      XRB+XRLOC
  .ULOG
```

UU.DEA -- Deallocate a Device or Deassign a User Logical

Form

```
  MOVB    #UU.DEA,FIRQB+FQFUN
          .
          .
          .
  (set up FIRQB)
          .
          .
          .
  .ULOG
```

Function

The UU.DEA subfunction of .ULOG deallocates a device from the current job (releases it for use by other jobs) or deassigns a user-logical assignment.

Privileges Required

None

Data Passed for Deallocate Device

```
                              FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
            +--------------------+--------------------+
          1 |/////////////////////////////////////////|  0
            +--------------------+--------------------+
FQFUN     3 |  UU.DEA (=octal 13) |///////////////////|  2
            +--------------------+--------------------+
          5 |/////////////////////////////////////////|  4
            +--------------------+--------------------+
          7 |               (must be 0)               |  6 FQPPN
            +--------------------+--------------------+
         11 |               (must be 0)               | 10 FQNAM1
            +--------------------+--------------------+
         13 |/////////////////////////////////////////| 12
            +--------------------+--------------------+
         15 |/////////////////////////////////////////| 14
            +--------------------+--------------------+
         17 |/////////////////////////////////////////| 16
            +--------------------+--------------------+
         21 |/////////////////////////////////////////| 20
            +--------------------+--------------------+
         23 |/////////////////////////////////////////| 22
            +--------------------+--------------------+
         25 |/////////////////////////////////////////| 24
            +--------------------+--------------------+
         27 |               (must be 0)               | 26 FQPFLG
            +--------------------+--------------------+
         31 |     device name (2 ASCII characters)    | 30 FQDEV
            +--------------------+--------------------+
         33 |≠0, unit number real | device unit number | 32 FQDEVN
            +--------------------+--------------------+
         35 |/////////////////////////////////////////| 34
            +--------------------+--------------------+
         37 |/////////////////////////////////////////| 36
            +--------------------+--------------------+
```

FIRQB+FQFUN	The UU.DEA function code (octal value equals 13).
FIRQB+FQDEV	The name of the device to be deallocated; two ASCII characters.
FIRQB+FQDEVN	The device unit number is passed in this byte, in binary. A nonzero value in FIRQB+FQDEVN+1 indicates an explicit device unit number. A zero value in FIRQB+FQDEVN+1 indicates no device unit number.

Note

The XRB should be cleared to zeros for deallocating a device.

Data Passed for Deassign User Logical

```
                                 FIRQB
Mne-  Octal                                                    Octal   Mne-
monic Offset                                                   Offset  monic
             +---------------------+---------------------+
           1 |/////////////////////////////////////////////|  0
             +---------------------+---------------------+
FQFUN      3 |  UU.DEA (=octal 13) |/////////////////////////|  2
             +---------------------+---------------------+
           5 |/////////////////////////////////////////////|  4
             +---------------------+---------------------+
           7 | ≠0, deassign default ppn; else must be 0  |  6 FQPPN
             +---------------------+---------------------+
          11 |                                           | 10 FQNAM1
             + user logical device name in RAD50 format  +
          13 |                 (2 words)                 | 12
             +---------------------+---------------------+
          15 |/////////////////////////////////////////////| 14
             +---------------------+---------------------+
          17 |/////////////////////////////////////////////| 16
             +---------------------+---------------------+
          21 |/////////////////////////////////////////////| 20
             +---------------------+---------------------+
          23 |/////////////////////////////////////////////| 22
             +---------------------+---------------------+
          25 |/////////////////////////////////////////////| 24
             +---------------------+---------------------+
          27 |/////////////////////|≠0,deass def prot cde| 26 FQPFLG
             +---------------------+---------------------+
          31 |/////////////////////////////////////////////| 30
             +---------------------+---------------------+
          33 |/////////////////////////////////////////////| 32
             +---------------------+---------------------+
          35 |/////////////////////////////////////////////| 34
             +---------------------+---------------------+
          37 |/////////////////////////////////////////////| 36
             +---------------------+---------------------+
```

FIRQB+FQFUN	The UU.DEA function code (octal value equals 13).
FIRQB+FQPPN	Any nonzero value deassigns any previously assigned default PPN. (In this case, set the word at FIRQB+FQNAM1 to zero.) If deassigning the default protection code, set this word to zero.

FIRQB+FQNAM1	The logical device name to be deassigned; two words in RAD50 format. If deassigning a default PPN or protection code, set the word at FIRQB+FQNAM1 to zero.
FIRQB+FQPFLG	Set this byte to nonzero if deassigning a protection code.

```
                               XRB
Mne- Octal                                          Octal  Mne-
monic Offset                                        Offset monic
         +--------------------+--------------------+
       1 |     length of user logical information   |  0 XRLEN
         +--------------------+--------------------+
       3 |     length of user logical information   |  2 XRBC
         +--------------------+--------------------+
       5 |  starting address for user logical info. |  4 XRLOC
         +--------------------+--------------------+
       7 |//////////////////////////////////////////|  6
         +--------------------+--------------------+
      11 |//////////////////////////////////////////| 10
         +--------------------+--------------------+
      13 |//////////////////////////////////////////| 12
         +--------------------+--------------------+
      15 |//////////////////////////////////////////| 14
         +--------------------+--------------------+
```

XRB+XRLEN	If the user logical information is in its standard location (USRPPN, USRPRT, and USRLOG), this word is passed as zero. If some nonstandard set of locations is being used, then specify the length (in bytes) of that information.
XRB+XRBC	The length of the information (same as the word at XRB+XRLEN).
XRB+XRLOC	If the word at XRB+XRLEN is nonzero, then this word defines the starting location for the user logical information (the default PPN). See Chapter 2 for a description of the order and format of the information created by .ULOG.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.DEA subfunction of .ULOG does not return any meaningful data.

Errors

BADCNT	This error has two possible causes: o The length specified at XRB+XRLEN and XRB+XRBC must be 4+8n, where n = 0, 1, 2, 3, or 4. If it is not, this error is returned. (Other lengths are not valid user logicals.) o The value at XRB+XRLOC is odd.
NODEVC	The device or its type specified at the locations FIRQB+FQDEV and FIRQB+FQDEVN is not part of your system configuration.

Example

The following code deallocates MT0: from the current job:

```
  MOVB    #UU.DEA,FIRQB+FQFUN           ; Set function code
  CLR     FIRQB+FQNAM1                  ; Clear name
  MOV     #"MT,FIRQB+FQDEV              ; Set device
  MOVB    #0,FIRQB+FQDEVN               ; Set unit number
  MOVB    #377,FIRQB+FQDEVN+1           ; Unit number real
  .ULOG
```

.UUO - Execute BASIC-PLUS SYS Call

Form

.UUO

Function

The .UUO directive lets a MACRO program execute the BASIC-PLUS SYS calls to the File Processor (FIP). The *RSTS/E Programming Manual* describes the SYS calls. For the MACRO programmer's convenience, the FIRQB formats for the calls are shown here; however, for detailed descriptions see the *RSTS/E Programming Manual*.

Table 3-10 summarizes the FIP SYS calls; note that some of the calls are handled by directives other than .UUO. (.FSS, for example, handles the file name string scan in MACRO.) The mnemonic subfunction names are provided by the COMMON.MAC prefix file; the decimal values are the BASIC-PLUS FIP call subfunction codes and the .UUO subfunction codes that COMMON.MAC relates to the mnemonics listed.

The rest of this section gives the FIRQB formats for data passed and returned for the .UUO subfunctions; the subfunctions are organized in alphabetical order of the mnemonics provided by COMMON.MAC.

Table 3-10: .UUO Subfunctions - Calls to the File Processor (FIP)

Mnemonic	BASIC-PLUS SYS Call Code (Decimal)	Privileges That Apply	Function
UU.TB3	-29	None	Get monitor tables -- Part III
UU.SPL	-28	r/w file	Spooling
UU.DMP	-27	SYSIO	Snap shot dump
UU.FIL	-26	DATES,TUNE SYSIO,r/w	File placement and modification
UU.ATR	-25	r/w acct	Read or write attributes
UU.CCL	-24	INSTAL	Add/delete CCL command
(.FSS)	-23	None	Terminating file name string scan

Table 3-10: .UUO Subfunctions - Calls to the File Processor (FIP) (Cont.)

Mnemonic	BASIC-PLUS SYS Call Code (Decimal)	Privileges That Apply	Function
(.SET)	-22	TUNE	Set special run priority
(.SET/ .CLEAR)	-21	None	Drop/regain temporary privilege
(.SET/ .CLEAR)	-20	TUNE	Lock/unlock job in memory
UU.LOG	-19	SWCTL	Set number of logins
UU.RTS	-18	read file INSTAL	Install run-time system
		INSTAL	Load, remove, or unload run-time system
		INSTAL	Declare default keyboard monitor
		read file INSTAL	Install resident library
		INSTAL	Load, remove, or unload resident library
UU.NAM	-17	write file	Name run-time system
UU.DIE	-16	SHUTUP	System shutdown
UU.ACT	-15	write acct	Accounting dump
UU.DAT	-14	DATES	Change system date/time
UU.PRI	-13	TUNE	Change priority/run burst/job size
UU.TB2	-12	None	Get monitor tables -- Part II
UU.BCK	-11	DATES	Change file backup statistics

Table 3-10: .UUO Subfunctions -- Calls to the File Processor (FIP) (Cont.)

Mnemonic	BASIC-PLUS SYS Call Code (Decimal)	Privileges That Apply	Function
(.FSS)	-10	None	File name string scan
UU.HNG	-9	HWCTL	Hangup a dataset
UU.FCB	-8	None	Get open channel statistics
(internal to BASIC-PLUS)	-7	None	CTRL/C trap enable
UU.POK	-6	SYSMOD	Poke memory
(.SPEC)	-5	SEND	Broadcast to terminal
(.SPEC)	-4	SYSIO	Force input to terminal
UU.TB1	-3	None	Get monitor tables -- Part I
UU.NLG	-2	SWCTL	Disable logins
UU.YLG	-1	SWCTL	Enable logins
UU.PAS	0	write acct	Create user account
UU.DLU	1	write acct	Delete user account
UU.MNT	3	MOUNT	Disk pack status
UU.LIN	4	None	Login
UU.BYE	5	WACNT EXTQA	Logout
UU.ATT	6	write acct DEVICE	Attach Reattach Swap console
UU.DET	7	JOBCTL EXQTA	Detach

Table 3-10: .UUO Subfunctions - Calls to the File Processor (FIP) (Cont.)

Mnemonic	BASIC-PLUS SYS Call Code (Decimal)	Privileges That Apply	Function
UU.CHU	8	write acct JOBCTL HWCTL	Change password/quota Kill job Disable terminal
UU.ERR	9	None	Return error message
UU.ASS	10	DEVICE HWCTL	Allocate/reallocate device *
UU.DEA	11	None	Deallocate device *
UU.DAL	12	None	Deallocate all devices *
UU.ZER	13	DEVICE write acct	Zero a device
UU.RAD	14	write acct	Read or read-and-reset accounting data
UU.DIR	15	DEVICE execute or read file	Directory lookup on index Special magnetic tape directory lookup
UU.TRM	16	HWCFG	Set terminal characteristics
UU.LOK	17	DEVICE read file	Disk directory lookup on file name Disk wildcard directory lookup
UU.CHE	19	TUNE	Enable/disable disk caching
UU.CNV	20	None	Date and time conversion
UU.SLN	21	INSTAL	System logical names
(.MESAG)	22	SEND SYSIO	Message send/receive

Table 3-10: .UUO Subfunctions - Calls to the File Processor (FIP) (Cont.)

Mnemonic	BASIC-PLUS SYS Call Code (Decimal)	Privileges That Apply	Function
UU.SWP	23	r/w files INSTALL	Install, remove, list system files
UU.JOB	24	WACNT JOBCTL EXQTA	Create job
UU.PPN	25	DEVICE	Wildcard PPN lookup
UU.SYS	26	JOBCTL TUNE	Return job status information
UU.PRV	28	None	Set/clear/read current privileges
UU.STL	29	HWCTL	Stall/unstall system
UU.3PP	31	None	Third-party privileges
UU.CHK	32	None	Check access function
UU.ONX	33	DEVICE DATES r/w files	Open next disk file
UU.CFG	34	HWCFG SWCFG HWCTL	Change system characteristics and device defaults
* To assign or deassign user logicals, use .ULOG.			

UU.ACT (Accounting Information Dump)

Privileges Required

GACNT or WACNT is required for access to accounting data.

Data Passed

```
                              FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +--------------------+--------------------+
        1 |////////////////////////////////////////|  0
          +--------------------+--------------------+
FQFUN   3 |    UU.ACT (= -17)  |////////////////////|  2
          +--------------------+--------------------+
        5 |////////////////////////////////////////|  4
          +--------------------+--------------------+
        7 |  project number    | programmer number  |  6 FQPPN
          +--------------------+--------------------+
       11 |////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |////////////////////////////////////////| 14
          +--------------------+--------------------+
       17 |////////////////////////////////////////| 16
          +--------------------+--------------------+
       21 |////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQPPN   Zero in both bytes means the current account.

Data Returned

Except for a possible error in byte 0 of the FIRQB, The UU.ACT subfunction of .UUO does not return any meaningful data.

Errors

NOSUCH	The account specified does not exist.
PRVIOL	A caller with GACNT privilege specified an account outside the caller's group.

UU.ASS (Allocate/Reallocate Device)

Privileges Required

DEVICE is needed to allocate a restricted device. HWCTL is required to seize a device or reallocate a device to a job in another account.

Data Passed

```
                               FIRQB
Mne-  Octal                                            Octal  Mne-
monic Offset                                           Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.ASS (= 12)    |/////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |/////////////////////| 0,alloc.;≠0,realloc.| 10 FQNAM1
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |    DOS or ANS in RAD50 format (1 word)    | 14 FQEXT
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |          100001 for seize; else 0         | 22 FQMODE
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |      device name (2 ASCII characters)     | 30 FQDEV
          +---------------------+---------------------+
       33 |   ≠0,unit no. real  |  device unit number | 32 FQDEVN
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Data Returned

FIRQB	The error code (if any).
FIRQB+FQFIL	The previous owner of the device as follows: o If you own a device already and allocate it, UU.ASS returns your own job number. o If you seize a device, you can see who you seized it from. o If you reallocate the device, you can see who owned it before the reallocation. o If the value in FIRQB+FQFIL is zero, then you know that the device was not previously owned.

Errors

BDNERR	Your program attempted to reallocate a device to a nonexistent job.
INUSE	For a reallocate, the specified device is open or has an open file.
NODEVC	Device name at FIRQB+FQDEV is not a valid device name on this system.
NOTAVL	The device specified exists on the system, but the operation failed for one of the following reasons: o The device is currently reserved by another job (see description of FIRQB+FQMODE). o The device or its controller is disabled. o The device is a keyboard line for a pseudo keyboard only. o You do not have DEVICE privilege and the device is restricted.
PRVIOL	You do not have the required privileges and tried to reallocate a device to a job that is logged in to an account other that you current account.

.UUO
UU.ATR

NOROOM	Occurs only on a write. The User File Directory (UFD) of the account is full. Some files must be deleted to free entries for attributes.
NOTOPN	Channel specified at FIRQB+FQFIL must have file open.
PRVIOL	Job does not have read/write access to the file open on the channel, or a UFD is open on the channel.

Privileges Required (Read/Write/Delete Account Attributes)

No privileges are required to read attributes 1, 2, and 4 to 191 of your own account. That is, of the DIGITAL-defined attributes you can read all except the password. Of the user-defined attributes (128 to 255) you can read the first 64. To read attribute 3 (password) or attributes 192 to 255, you need GREAD, WREAD, GACNT, or WACNT privilege.

Read or accounting privileges (GREAD, GACNT, WREAD, or WACNT) are required to read accounting data of other accounts. Accounting access (GACNT or WACNT) is required to write or delete accounting data of any account.

Data Passed (Read/Write/Delete Account Attributes)

```
                              FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
            +--------------------+--------------------+
        1   |//////////////////////////////////////////|  0
            +--------------------+--------------------+
FQFUN   3   |   UU.ATR (= -31)   |////////////////////|  2
            +--------------------+--------------------+
FQSIZM  5   | attribute type code|  subfunction code  |  4 FQFIL
            +--------------------+--------------------+
        7   |   project number   | programmer number  |  6 FQPPN
            +-----------------------------------------+
       11   |                    |  (see discussion)  | 10 FQNAM1
            +                    +--------------------+
       13   |                                         | 12
            +           13 bytes of data              +
       15   |           (only on write)               | 14
            +                                         +
       17   |                                         | 16
            +                                         +
       21   |                                         | 20 FQNAM2
            +                                         +
       23   |                                         | 22
            +                                         +
       25   |                                         | 24
            +-----------------------------------------+
       27   |//////////////////////////////////////////| 26
            +-----------------------------------------+
       31   |   device name (2 ASCII characters)      | 30 FQDEV
            +-----------------------------------------+
       33   | ≠0, unit no. real  | device unit number | 32 FQDEVN
            +--------------------+--------------------+
       35   |//////////////////////////////////////////| 34
            +--------------------+--------------------+
       37   |//////////////////////////////////////////| 36
            +--------------------+--------------------+
```

FIRQB+FQFUN	The function code UU.ATR (octal value = -31).
FIRQB+FQFIL	The subfunction code:

-1	Read account attributes
-2	Write account attributes
-3	Delete account attributes

FIRQB+FQSIZM	The attribute type code: 0 Look-up by index 1 Quotas 2 Authorized Privilege Mask 3 Password 4 Date/time information 5 User name information 6 Nondisk quotas
FIRQB+FQPPN	The PPN of the account accessed. The project number is in the high byte (FQPPN+1), and the programmer number is in the low byte (FQPPN). A value of zero defaults to the caller's PPN.
FIRQB+FQNAM1	The index in a "look-up by index." Look-up by index is specified by using a type code of zero in FIRQB+FQSIZM and -1 in FIRQB+FQFIL. For programs like BACKUP, it is useful to read all the account attributes without having to try each of the 255 possible type codes. The value n of the index specifies that you want nth attribute type.
FIRQB+FQNAM1+1	The account attributes to be written to one attribute block. The data is written exactly as passed except for attribute types 2 and 4. For type 2 (authorized privilege mask), when writing the mask, any attempt to turn on privilege bits are ignored if the caller does not have the corresponding privilege currently in effect. The layout of the 13 bytes are: o FIRQB+11 -- reserved o FIRQB+12 through +21 -- authorized privilege mask o FIRQB+22 through +25 -- reserved For type 4 (date/time information), the last log-in fields are always left alone. This ensures that any logins to an account leave a trace that cannot be tampered with. The layout of the 13 bytes are: o FIRQB+11 -- keyboard of last login (-1 if detached) o FIRQB+12,+13 -- date of last login (RSTS internal format)

- o FIRQB+14,+15 -- time of last login (RSTS internal format)
- o FIRQB+16,+17 -- date of last password change
- o FIRQB+20,+21 -- time of last password change and flags
- o FIRQB+22,+23 -- date of account creation
- o FIRQB+24,+25 -- expiration date (-1 if none)

The flags in FIRQB+14,+15 are:

- o Bit 11 -- no password required ("guest" account)
- o Bits 12-15 -- reserved

The flags in FIRQB+20,+21 are:

- o Bit 11 -- password cannot be looked up ("hashed" password)
- o Bit 12 -- no dial-up logins allowed
- o Bit 13 -- no network logins allowed
- o Bit 14 -- no interactive logins allowed (spawn and batch only)
- o Bit 15 -- captive account

If bit 11 is set, then passwords can be up to 14 characters long and the system stores them in hashed form. The hash uses parts of the date/time attributes; so, altering any bits of bytes FIRQB+16 through FIRQB+23 (except for the flag bits) renders the password useless. The read accounting data directive (UU.RAD) returns two zero words if this bit is set.

If bit 11 is clear, then passwords can be up to six characters long and must consist of uppercase letters and numbers; that is, the RAD50 character set.

FIRQB+FQDEV	The device name passed; two ASCII characters. A value of zero indicates "SY", public disk.
FIRQB+FQDEVN	The device unit number passed in binary notation. A nonzero value in the high byte of this word (FIRQB+FQDEVN+1) indicates an explicit device unit number. A zero value in the high byte indicates no unit number.

Data Returned (Read/Write/Delete Account Attributes)

Other than a possible error in byte 0 of the FIRQB, data is returned only on a read subfunction (-1 in FIRQB+FQFIL).

```
                               FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +----------------------+----------------------+
        1 |//////////////////////////////////////////////|  0
          +----------------------+----------------------+
        3 |//////////////////////| current job no. * 2 |  2 FQJOB
          +----------------------+----------------------+
        5 | attribute type code |//////////////////////|  4
          +----------------------+----------------------+
        7 |   project number    | programmer number   |  6 FQPPN
          +---------------------------------------------+
       11 |                     |    attribute type   | 10 FQNAM1
          +                     +----------------------+
       13 |                                             | 12
          +                                             +
       15 |                                             | 14
          +                                             +
       17 |              13 bytes of data              | 16
          +                                             +
       21 |                                             | 20
          +                                             +
       23 |                                             | 22
          +                                             +
       25 |                                             | 24
          +---------------------------------------------+
       27 |//////////////////////////////////////////////| 26
          +---------------------------------------------+
       31 |//////////////////////////////////////////////| 30
          +---------------------------------------------+
       33 |//////////////////////////////////////////////| 32
          +----------------------+----------------------+
       35 |//////////////////////////////////////////////| 34
          +---------------------------------------------+
       37 |//////////////////////////////////////////////| 36
          +----------------------+----------------------+
```

FIRQB+FQNAM1	The attribute type code. Contains the type code of the attributes just read (same as passed except for lookup by index).
FIRQB+FQNAM1+1	The actual attribute data.

Errors (Read/Write/Delete Account Attributes)

DEVNFS	The specified device is not a disk.
EOF	The attribute type not found (read or delete only). The index specified for an indexed look-up call is greater than the number of attributes (read only).
NODEVC	The specified device does not exist.
NOROOM	The attribute block did not exist and it cannot be added because the directory is full (write only).
NOSUCH	The specified account does not exist.
NOTMNT	The specified disk is not mounted.
PRVIOL	You do not have sufficient privilege for the operation. The disk is write-locked (write and delete only). Attempt to delete in the DIGITAL-reserved type range (delete only).

Privileges Required (Read Pack Attributes)

DEVICE is required if the device specified is restricted.

Data Passed (Read Pack Attributes)

```
                                FIRQB
Mne-  Octal                                                  Octal   Mne-
monic Offset                                                 Offset  monic
           +---------------------+---------------------+
         1 |/////////////////////////////////////////////| 0
           +---------------------+---------------------+
FQFUN    3 |    UU.ATR (= -31)   |/////////////////////| 2
           +---------------------+---------------------+
         5 |/////////////////////|subfunction code (-4)| 4 FQFIL
           +---------------------+---------------------+
         7 |/////////////////////////////////////////////| 6 FQPPN
           +-------------------------------------------+
        11 |/////////////////////////////////////////////| 10 FQNAM1
           +-------------------------------------------+
        13 |/////////////////////////////////////////////| 12
           +-------------------------------------------+
        15 |/////////////////////////////////////////////| 14 FQEXT
           +-------------------------------------------+
        17 |/////////////////////////////////////////////| 16 FQSIZ
           +-------------------------------------------+
        21 |/////////////////////////////////////////////| 20 FQNAM2
           +-------------------------------------------+
        23 |/////////////////////////////////////////////| 22 FQMODE
           +-------------------------------------------+
        25 |/////////////////////////////////////////////| 24 FQFLAG
           +-------------------------------------------+
        27 |/////////////////////////////////////////////| 26 FQPFLG
           +-------------------------------------------+
        31 |     device name (2 ASCII characters)      | 30 FQDEV
           +-------------------------------------------+
        33 |  ≠0, unit no. real  | device unit number  | 32 FQDEVN
           +---------------------+---------------------+
        35 |/////////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |/////////////////////////////////////////////| 36
           +---------------------+---------------------+
```

Data Returned (Read Pack Attributes)

This call returns information about mounted disks. It is used to obtain the characteristics of a disk and the drive the disk is mounted in.

```
                              FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
          +----------------------+----------------------+
        1 |//////////////////////////////////////////////|  0
          +----------------------+----------------------+
        3 |//////////////////////////////////////////////|  2
          +----------------------+----------------------+
        5 |//////////////////////////////////////////////|  4
          +----------------------+----------------------+
        7 |//////////////////////////////////////////////|  6
          +---------------------------------------------+
       11 |              1st DCN of MFD                 | 10 FQNAM1
          +---------------------------------------------+
       13 |            Pack Revision Level              | 12
          +---------------------------------------------+
       15 |             Pack Cluster Size               | 14 FQEXT
          +---------------------------------------------+
       17 |             Pack Status/Flags               | 16 FQSIZ
          +---------------------------------------------+
       21 |          Pack ID in RAD50 format            | 20 FQNAM2
          +                                             +
       23 |               (in 2 words)                  | 22 FQMODE
          +---------------------------------------------+
       25 |            Size of disk in DCNs             | 24 FQFLAG
          +---------------------------------------------+
       27 |            Device Cluster Size              | 26 FQPFLG
          +---------------------------------------------+
       31 |//////////////////////////////////////////////| 30
          +---------------------------------------------+
       33 |//////////////////////////////////////////////| 32
          +----------------------+----------------------+
       35 |//////////////////////////////////////////////| 34
          +---------------------------------------------+
       37 |//////////////////////////////////////////////| 36
          +----------------------+----------------------+
```

FIRQB+FQNAM1	The starting device cluster number (DCN) of the MFD.
FIRQB+12	The pack revision level.
FIRQB+FQEXT	The pack cluster size.

FIRQB+FQSIZ	The pack status/flags. The defined bits are: o Bit 9 -- Pack is initialized "new files first" o Bit 11 -- Pack is initialized to maintain date of last write o Bit 12 -- Pack is initialized as a read-only pack o Bit 14 -- Pack is initialized as a private/system disk All other bits are reserved.
FIRQB+FQNAM2	The pack ID in RAD50 format. The first three characters are here and the last three characters are in the next word, FIRQB+FQMODE.
FIRQB+FQFLAG	The size of the disk in DCNs.
FIRQB+FQPFLG	The device cluster size.

Errors (Return Pack Attributes)

DEVNFS	Device specified is not a logically mounted device.
NODEVC	Device specified is not valid.

UU.ATT (Attach/Reattach Job/Swap Console)

Privileges Required

DEVICE is required to perform a REATTACH if the terminal is restricted. GACNT or WACNT is required to ATTACH to a job outside the caller's account.

Data Passed - Attach

```
                                FIRQB
Mne-   Octal                                               Octal   Mne-
monic  Offset                                              Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////|  0
          +--------------------+--------------------+
FQFUN   3 |    UU.ATT (= 6)    |////////////////////|  2
          +--------------------+--------------------+
FQSIZM  5 |      bit mask      | job number to attach|  4 FQFIL
          +--------------------+--------------------+
        7 |  project number    | programmer number  |  6 FQPPN
          +--------------------+--------------------+
       11 |        password in RAD50 format         | 10 FQNAM1
          +                                         +
       13 |                (2 words)                | 12
          +                                         +
       15 |                   or                    | 14
          +                                         +
       17 |        password in ASCII format         | 16
          +                                         +
       21 |                (7 words)                | 20
          +                                         +
       23 |                                         | 22
          +                                         +
       25 |                                         | 24
          +-----------------------------------------+
       27 |/////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |/////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |/////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |/////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |/////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQSIZM	Bit 0 must be clear for the attach subfunction. Bit 1 is clear for the old password format (RAD50) and set for the new format (ASCII). Bit 2 can be set on an attach to suppress the password check. If clear, the password is checked if you attach to a job in another account. All other bits are reserved.
FIRQB+FQPPN	Zero in both bytes means your PPN. You do not need to include a password. If you do not have GACNT or WACNT privilege, you can only specify your own PPN.
FIRQB+FQNAM1	You must include a password if you specify a PPN other than your own unless bit 2 is set in FIRQB+FQSIZM. Note that this use of the call requires GACNT or WACNT privilege.

Data Returned

Other than a possible error in byte 0 of the FIRQB, the attach function of the UU.ATT directive does not return any data.

Errors

BADFUO — The system returns this error for one of the following reasons:

- o Job executing the call has an open channel.
- o Job executing the call is detached.
- o Caller is a source (BAS) program rather than a compiled (BAC) program.
- o Job number at FIRQB+FQFIL is not detached.
- o The PPN specified does not match PPN of job to attach.
- o Caller does not have GACNT or WACNT privilege, and the job to attach to has a PPN different from that of the caller.
- o Password is not valid.

Data Passed - Reattach

```
                               FIRQB
Mne-   Octal                                               Octal   Mne-
monic Offset                                                Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.ATT (= 6)     |/////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 | KB no. to attach to | caller's job number |  4 FQFIL
          +---------------------+---------------------+
        7 |/////////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |/////////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |/////////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |/////////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |/////////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |/////////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |/////////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |/////////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Data Returned

Other than a possible error in byte 0 of the FIRQB, the reattach function of the UU.ATT directive does not return any data.

Errors

BADFUO	The system returns this error for one of the following reasons:
	o Job number at FIRQB+FQFIL is out of range or no such job exists.

- o Caller is not detached.
- o KB number is out of range.
- o KB is currently allocated, opened, or the console keyboard of some job other than the calling job.
- o The terminal is restricted and the caller does not have DEVICE privilege.

Data Passed -- Swap Console

```
                              FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.ATT (= 6)     |/////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 |      bit mask       |job no. to swap with |  4 FQFIL
          +---------------------+---------------------+
        7 |/////////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |/////////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |/////////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |/////////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |/////////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |/////////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |/////////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |/////////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQSIZM	Bit 0 must be set for the swap console subfunction. All other bits are reserved.
FIRQB+FQFIL	If the calling job is attached, this job must be detached. If the calling job is detached, this job must be attached. Both the calling job and this job must be running under the same PPN.

Data Returned

Other than a possible error in byte 0 of the FIRQB, the swap console function of the UU.ATT directive does not return any data.

Errors

BADFUO — The system returns this error for one of the following reasons:

- o Both the calling job and the job specified at FIRQB+FQFIL are detached, or neither job is detached.
- o The job specified at FIRQB+FQFIL has a PPN different from the caller's.
- o The value at FIRQB+FQSIZM is neither 0 nor 1.

UU.BCK (Change File Statistics)

Privileges Required

DATES is required to change the last access date.

Data Passed

```
                                 FIRQB
Mne-   Octal                                              Octal   Mne-
monic Offset                                             Offset   monic
            +---------------------+---------------------+
          1 |/////////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN     3 |    UU.BCK (= -13)   |/////////////////////|  2
            +---------------------+---------------------+
FQSIZM    5 | LSB of last access  |  channel no. (1-17) |  4 FQFIL
            +---------------------+---------------------+
          7 |LSB of creation date | MSB of last access  |  6 FQPPN
            +---------------------+---------------------+
         11 |LSB of creation time |MSB of creation date | 10 FQNAM1
            +---------------------+---------------------+
         13 |/////////////////////|MSB of creation time | 12
            +---------------------+---------------------+
         15 |/////////////////////////////////////////////| 14
            +---------------------+---------------------+
         17 |/////////////////////////////////////////////| 16
            +---------------------+---------------------+
         21 |/////////////////////////////////////////////| 20
            +---------------------+---------------------+
         23 |/////////////////////////////////////////////| 22
            +---------------------+---------------------+
         25 |/////////////////////////////////////////////| 24
            +---------------------+---------------------+
         27 |/////////////////////////////////////////////| 26
            +---------------------+---------------------+
         31 |/////////////////////////////////////////////| 30
            +---------------------+---------------------+
         33 |/////////////////////////////////////////////| 32
            +---------------------+---------------------+
         35 |/////////////////////////////////////////////| 34
            +---------------------+---------------------+
         37 |/////////////////////////////////////////////| 36
            +---------------------+---------------------+
```

FIRQB+5,6 FIRQB+7,10	The system internal format for dates has the form: [(year - 1970) * 1000] + day-within-year
FIRQB+11,12	Time is specified in minutes until midnight, where 1440 equals midnight.

Note

See the .DATE directive for a discussion of these formats.

Data Returned

Other than a possible error in byte 0 of the FIRQB, the UU.BCK subfunction does not return any meaningful data.

Errors

BADFUO	The file open on the channel specified is either not a disk file or is a user file directory (UFD).

UU.BYE (Logout)

Privileges Required

EXQTA is required to log out without a quota check. WACNT is required to log out without a self-kill.

Data Passed

```
                                  FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                                Offset  monic
         +---------------------+---------------------+
       1 |/////////////////////////////////////////////|  0
         +---------------------+---------------------+
FQFUN  3 |    UU.BYE (= 5)     |/////////////////////|  2
         +---------------------+---------------------+
       5 |/////////////////////|   type of logout    |  4 FQFIL
         +---------------------+---------------------+
       7 |/////////////////////////////////////////////|  6
         +---------------------+---------------------+
      11 |/////////////////////////////////////////////| 10
         +---------------------+---------------------+
      13 |/////////////////////////////////////////////| 12
         +---------------------+---------------------+
      15 |/////////////////////////////////////////////| 14
         +---------------------+---------------------+
      17 |/////////////////////////////////////////////| 16
         +---------------------+---------------------+
      21 |/////////////////////////////////////////////| 20
         +---------------------+---------------------+
      23 |/////////////////////////////////////////////| 22
         +---------------------+---------------------+
      25 |/////////////////////////////////////////////| 24
         +---------------------+---------------------+
      27 |/////////////////////////////////////////////| 26
         +---------------------+---------------------+
      31 |/////////////////////////////////////////////| 30
         +---------------------+---------------------+
      33 |/////////////////////////////////////////////| 32
         +---------------------+---------------------+
      35 |/////////////////////////////////////////////| 34
         +---------------------+---------------------+
      37 |/////////////////////////////////////////////| 36
         +---------------------+---------------------+
```

FIRQB+FQFIL Bits 0 and 1 specify:

Bit	Setting	Meaning
0	Clear	Close I/O channels, deassign devices, remove receivers, and dismount disks mounted /NOSHARE before performing logout.
	Set	Perform logout only; do not close channels, deassign devices, remove receivers, or dismount disks. This call requires WACNT privilege.
1	Clear	Check quotas on all mounted disks before performing logout.
	Set	Perform logout without checking disk quotas. This call requires EXQTA privilege.

If the caller does not have WACNT privilege, RSTS/E forces Bit 0 to zero. If the caller does not have EXQTA privilege, RSTS/E forces Bit 1 to zero.

Bits 2 through 7 are reserved.

Data Returned

```
                                FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                             Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |      logout status (0, -1, or -2)         |  4 FQFIL
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |/////////////////////|      quota code     | 16 FQSIZ
          +---------------------+---------------------+
       21 | MSB of current usage| MSB of current quota| 20 FQNAM2
          +---------------------+---------------------+
       23 |no. of det. jobs all.|   no. of det. jobs  | 22 FQMODE
          +---------------------+---------------------+
       25 |   LSB of current disk quota (in blocks)   | 24 FQFLAG
          +---------------------+---------------------+
       27 |   LSB of current disk usage (in blocks)   | 26 FQPFLG
          +---------------------+---------------------+
       31 |      disk name (2 ASCII characters)       | 30 FQDEV
          +---------------------+---------------------+
       33 |  ≠0, unit no. real  |   disk unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFIL	Zero means neither disk quota nor detached job quota is exceeded. If you have WACNT privilege, the system returns control to your program. Otherwise, the system kills your job after performing necessary clean-up functions.
	Minus one means a quota is exceeded; your job is still logged in.

	Minus two means a quota is exceeded, but your job is logged out. If you have WACNT privilege, the system returns control to your program. Otherwise, the system kills your job after performing necessary clean-up functions.
FIRQB+FQSIZ	Indicates which quota is exceeded. (This value is valid only if FIRQB+FQFIL equals -1 or -2.) Zero means disk quota. One means detached job quota.
FIRQB+FQMODE	Returned if detached job quota is exceeded. FIRQB+FQMODE contains the number of detached jobs in the current account. FIRQB+FQMODE+1 contains the number of detached jobs allowed.
FIRQB+FQDEV	Returned if disk quota is exceeded.

Errors

No errors are possible with the UU.BYE subfunction.

UU.CCL (Add/Delete CCL Command)

Privileges Required

INSTAL is required to add or delete a CCL command.

Data Passed -- Add

```
                                FIRQB
Mne-  Octal                                               Octal  Mne-
monic Offset                                              Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |   UU.CCL (= -30)    |/////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 | abbrev. point, chars|   must = 0 for add  |  4 FQFIL
          +---------------------+---------------------+
        7 |   project number    | programmer number   |  6 FQPPN
          +---------------------+---------------------+
       11 |         file name in RAD50 format         | 10 FQNAM1
          +             of program to run             +
       13 |                (2 words)                  | 12
          +---------------------+---------------------+
       15 |    file type in RAD50 format (1 word)     | 14 FQEXT
          +---------------------+---------------------+
       17 | CCL command  --  one to nine characters   | 16 FQSIZ
          +    padded with NULs (ASCII code 000)      +
       21 |           to nine characters              | 20
          +                                           +
       23 |                                           | 22
          +                                           +
       25 |                                           | 24
          +---------------------+                     +
FQPROT 27 |      (must be 0)    |                     | 26
          +---------------------+---------------------+
       31 | device name (2 ASCII chars) must be disk  | 30 FQDEV
          +---------------------+---------------------+
       33 | ≠0,unit number real |  device unit number | 32 FQDEVN
          +---------------------+---------------------+
       35 |      line number to start execution       | 34 FQCLUS
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Data Passed - Delete

```
                                FIRQB
Mne-   Octal                                             Octal   Mne-
monic Offset                                             Offset  monic
            +---------------------+---------------------+
          1 |/////////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN     3 |    UU.CCL (= -30)   |///////////////////////|  2
            +---------------------+---------------------+
FQSIZM    5 | abbrev. point, chars|   = -2 for delete   |  4 FQFIL
            +---------------------+---------------------+
          7 |/////////////////////////////////////////////|  6
            +---------------------+---------------------+
         11 |/////////////////////////////////////////////| 10
            +---------------------+---------------------+
         13 |/////////////////////////////////////////////| 12
            +---------------------+---------------------+
         15 |/////////////////////////////////////////////| 14
            +---------------------+---------------------+
         17 |  CCL command  --  one to nine characters  | 16 FQSIZ
            +     padded with NULs (ASCII code 000)     +
         21 |            to nine characters             | 20
            +                                           +
         23 |                                           | 22
            +                                           +
         25 |                                           | 24
            +---------------------+                     +
         27 |/////////////////////|                     | 26
            +---------------------+---------------------+
         31 |/////////////////////////////////////////////| 30
            +---------------------+---------------------+
         33 |/////////////////////////////////////////////| 32
            +---------------------+---------------------+
         35 |/////////////////////////////////////////////| 34
            +---------------------+---------------------+
         37 |/////////////////////////////////////////////| 36
            +---------------------+---------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.CCL directive does not return any meaningful data.

Errors

BADNAM	For add only. The CCL command either begins with a number or contains an otherwise unacceptable character.
INUSE	For add only. The CCL command is already defined.
NODEVC	For delete only. The CCL command does not exist.

.UUO
UU.CFG

FIRQB+FQPPN Parameter change flags:

Bit	Setting	Meaning
0	Clear	No change in enable/disable status
	Set	Use value in FIRQB+FQPPN+1 to change status
1	Clear	No change in device ownership
	Set	Use value in FIRQB+FQPPN+1 to change ownership
2	Clear	No change in modem control
	Set	Use value in FIRQB+FQPPN+1 to change modem control

All combinations of the bit values are legal.

FIRQB+FQPPN+1 If FIRQB+FQPPN is zero, then this byte should also be zero. If FIRQB+FQPPN is not zero, then the following values are used to alter device parameters:

Bit	Setting	Meaning
0	Clear	Enable device
	Set	Disable device
1	Clear	No privilege needed for device ownership
	Set	DEVICE privilege needed for device ownership
2	Clear	Put device under LOCAL control
	Set	Put device under MODEM control

Data Returned (Set DEVn: characteristics)

Except for a possible error in byte 0 of the FIRQB, the system returns the following:

```
                              FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                               Offset  monic
         +--------------------+--------------------+
       1 |/////////////////////////////////////////|  0
         +--------------------+--------------------+
       3 |/////////////////////////////////////////|  2
         +--------------------+--------------------+
       5 |/////////////////////////////////////////|  4
         +--------------------+--------------------+
       7 |  device parameters | device status flags |  6 FQPPN
         +--------------------+--------------------+
      11 |/////////////////////////////////////////| 10
         +--------------------+--------------------+
      13 |/////////////////////////////////////////| 12
         +--------------------+--------------------+
      15 |/////////////////////////////////////////| 14
         +--------------------+--------------------+
      17 |/////////////////////////////////////////| 16
         +--------------------+--------------------+
      21 |/////////////////////////////////////////| 20
         +--------------------+--------------------+
      23 |/////////////////////////////////////////| 22
         +--------------------+--------------------+
      25 |/////////////////////////////////////////| 24
         +--------------------+--------------------+
      27 |/////////////////////////////////////////| 26
         +--------------------+--------------------+
      31 |/////////////////////////////////////////| 30
         +--------------------+--------------------+
      33 |/////////////////////////////////////////| 32
         +--------------------+--------------------+
      35 |/////////////////////////////////////////| 34
         +--------------------+--------------------+
      37 |/////////////////////////////////////////| 36
         +--------------------+--------------------+
```

Errors (Set LPn: characteristics)

BDNERR	An attempt was made to set both bits 0 and 1 in the characteristics flag word.
NODEVC	An attempt was made to set characteristics of a printer which is not supported by the monitor, or the device name specified in FIRQB+FQDEV was not LP.

Data Passed (Set System Defaults)

```
                                FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
           +--------------------+--------------------+
        1  |/////////////////////////////////////////|  0
           +--------------------+--------------------+
FQFUN   3  |    UU.CFG (= 42)   |////////////////////|  2
           +--------------------+--------------------+
        5  |////////////////////| subfunction code =2 |  4 FQFIL
           +--------------------+--------------------+
        7  |         powerfail delay in seconds       |  6 FQPPN
           +--------------------+--------------------+
       11  | default time format | default date format | 10 FQNAM1
           +--------------------+--------------------+
       13  | default tape density| default tape label  | 12
           +--------------------+--------------------+
       15  |/////////////////////////////////////////| 14
           +--------------------+--------------------+
       17  |/////////////////////////////////////////| 16
           +--------------------+--------------------+
       21  |/////////////////////////////////////////| 20
           +--------------------+--------------------+
       23  |/////////////////////////////////////////| 22
           +--------------------+--------------------+
       25  |/////////////////////////////////////////| 24
           +--------------------+--------------------+
       27  |/////////////////////////////////////////| 26
           +--------------------+--------------------+
       31  |/////////////////////////////////////////| 30 FQDEV
           +--------------------+--------------------+
       33  |/////////////////////////////////////////| 32 FQDEVN
           +--------------------+--------------------+
       35  |/////////////////////////////////////////| 34
           +--------------------+--------------------+
       37  |/////////////////////////////////////////| 36
           +--------------------+--------------------+
```

FIRQB+FQPPN	Default powerfail delay in seconds. Zero indicates no change. A nonzero value represents the new default powerfail delay. The maximum is 454 (octal) seconds.

Data Passed -- Delete

```
                                FIRQB
Mne-   Octal                                                 Octal   Mne-
monic Offset                                                 Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.CCL (= -30)   |/////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 | abbrev. point, chars|   = -2 for delete   |  4 FQFIL
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |  CCL command  --  one to nine characters   | 16 FQSIZ
          +     padded with NULs (ASCII code 000)     +
       21 |            to nine characters             | 20
          +                                           +
       23 |                                           | 22
          +                                           +
       25 |                                           | 24
          +---------------------+                     +
       27 |/////////////////////|                     | 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.CCL directive does not return any meaningful data.

Errors

BADNAM	For add only. The CCL command either begins with a number or contains an otherwise unacceptable character.
INUSE	For add only. The CCL command is already defined.
NODEVC	For delete only. The CCL command does not exist.

UU.CFG (Set Device/System Default Characteristics)

Privileges Required

HWCFG is required to:

- Declare a device as restricted or unrestricted
- Set magnetic tape default density
- Set line printer characteristics

SWCFG is required to:

- Set magnetic tape label defaults
- Set power fail delay
- Set date/time presentation formats
- Set system flags

HWCTL is required to enable/disable a device.

Data Passed (Set DEVn: characteristics)

```
                                  FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
          +--------------------+--------------------+
       1  |/////////////////////////////////////////|  0
          +--------------------+--------------------+
FQFUN  3  |    UU.CFG (= 42)   |////////////////////|  2
          +--------------------+--------------------+
       5  |////////////////////| subfunction code =0 |  4 FQFIL
          +--------------------+--------------------+
       7  | parameter switches | par. change flags  |  6 FQPPN
          +--------------------+--------------------+
      11  |/////////////////////////////////////////| 10
          +--------------------+--------------------+
      13  |/////////////////////////////////////////| 12
          +--------------------+--------------------+
      15  |/////////////////////////////////////////| 14
          +--------------------+--------------------+
      17  |/////////////////////////////////////////| 16
          +--------------------+--------------------+
      21  |/////////////////////////////////////////| 20
          +--------------------+--------------------+
      23  |/////////////////////////////////////////| 22
          +--------------------+--------------------+
      25  |/////////////////////////////////////////| 24
          +--------------------+--------------------+
      27  |/////////////////////////////////////////| 26
          +--------------------+--------------------+
      31  |  device name (2 ASCII chars) not a disk |  30 FQDEV
          +--------------------+--------------------+
      33  |        377         | device unit number |  32 FQDEVN
          +--------------------+--------------------+
      35  |/////////////////////////////////////////| 34
          +--------------------+--------------------+
      37  |/////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQPPN — Parameter change flags:

Bit	Setting	Meaning
0	Clear	No change in enable/disable status
	Set	Use value in FIRQB+FQPPN+1 to change status
1	Clear	No change in device ownership
	Set	Use value in FIRQB+FQPPN+1 to change ownership
2	Clear	No change in modem control
	Set	Use value in FIRQB+FQPPN+1 to change modem control

All combinations of the bit values are legal.

FIRQB+FQPPN+1 — If FIRQB+FQPPN is zero, then this byte should also be zero. If FIRQB+FQPPN is not zero, then the following values are used to alter device parameters:

Bit	Setting	Meaning
0	Clear	Enable device
	Set	Disable device
1	Clear	No privilege needed for device ownership
	Set	DEVICE privilege needed for device ownership
2	Clear	Put device under LOCAL control
	Set	Put device under MODEM control

Data Returned (Set DEVn: characteristics)

Except for a possible error in byte 0 of the FIRQB, the system returns the following:

```
                                FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                              Offset  monic
         +--------------------+--------------------+
       1 |////////////////////////////////////////|  0
         +--------------------+--------------------+
       3 |////////////////////////////////////////|  2
         +--------------------+--------------------+
       5 |////////////////////////////////////////|  4
         +--------------------+--------------------+
       7 |  device parameters | device status flags|  6  FQPPN
         +--------------------+--------------------+
      11 |////////////////////////////////////////| 10
         +--------------------+--------------------+
      13 |////////////////////////////////////////| 12
         +--------------------+--------------------+
      15 |////////////////////////////////////////| 14
         +--------------------+--------------------+
      17 |////////////////////////////////////////| 16
         +--------------------+--------------------+
      21 |////////////////////////////////////////| 20
         +--------------------+--------------------+
      23 |////////////////////////////////////////| 22
         +--------------------+--------------------+
      25 |////////////////////////////////////////| 24
         +--------------------+--------------------+
      27 |////////////////////////////////////////| 26
         +--------------------+--------------------+
      31 |////////////////////////////////////////| 30
         +--------------------+--------------------+
      33 |////////////////////////////////////////| 32
         +--------------------+--------------------+
      35 |////////////////////////////////////////| 34
         +--------------------+--------------------+
      37 |////////////////////////////////////////| 36
         +--------------------+--------------------+
```

FIRQB+FQPPN	Device status flags indicating:

Bit	Setting	Meaning
0	Clear	Device is enabled and free
	Set	Device is disabled or in use
1	Clear	Device ownership does not require privilege
	Set	Device ownership requires HWCFG privilege
2	Clear	Device is in LOCAL mode
	Set	Device is under MODEM control

Note that Bit 2 is only valid if the device name passed in FIRQB+FQDEV is KB.

FIRQB+FQPPN+1 Valid values are:

0 -- If bit 0 in FIRQB+FQPPN is clear.

3 -- The device was disabled by this call and can be enabled by this call.

Any even integer -- Job number * 2 of the current device owner.

Any odd integer (other than 3) -- The device was disabled by the monitor and cannot be enabled.

Errors (Set DEVn: characteristics)

INUSE	An attempt was made to disable a device that was in use by another user or one that had been previously disable.
NOTAVL	An attempt was made to enable a device which was not disabled through use of this call.
PRVIOL	An attempt was made to alter device characteristic of a disk unit.

Data Passed (Set LPn: characteristics)

```
                                FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
           +----------------------+----------------------+
        1  |/////////////////////////////////////////////|  0
           +----------------------+----------------------+
FQFUN   3  |    UU.CFG (= 42)     |//////////////////////|  2
           +----------------------+----------------------+
        5  |//////////////////////| subfunction code =1  |  4 FQFIL
           +----------------------+----------------------+
        7  |      case flag       | default page width   |  6 FQPPN
           +----------------------+----------------------+
       11  |     set characteristics flag bits mask      | 10 FQNAM1
           +----------------------+----------------------+
       13  |    clear characteristics flag bits mask     | 12
           +----------------------+----------------------+
       15  | LPTSPC (ASCII value)|0=no chg/1=chg LPTSPC| 14 FQEXT
           +----------------------+----------------------+
       17  |   0=no change, n=new default form length    | 16 FQSIZ
           +----------------------+----------------------+
       21  |/////////////////////////////////////////////| 20
           +----------------------+----------------------+
       23  |/////////////////////////////////////////////| 22
           +----------------------+----------------------+
       25  |/////////////////////////////////////////////| 24
           +----------------------+----------------------+
       27  |/////////////////////////////////////////////| 26
           +----------------------+----------------------+
       31  | device name (2 ASCII chars) must be "LP"    | 30 FQDEV
           +----------------------+----------------------+
       33  |         377          |  device unit number  | 32 FQDEVN
           +----------------------+----------------------+
       35  |/////////////////////////////////////////////| 34
           +----------------------+----------------------+
       37  |/////////////////////////////////////////////| 36
           +----------------------+----------------------+
```

FIRQB+FQPPN	Default Page Width; zero indicates no change, nonzero indicates the new value for the default page width.
FIRQB+FQPPN+1	Case flag as follows:

0	No change
1	Printer can use mixed case
255	Printer can use uppercase only

FIRQB+FQNAM1	Set characteristics flag bits mask. Zero indicates no change. A nonzero value represents the flag bits to be set in the characteristics flag word. The bit positions currently in use are:

Bit 0	Allow <BS> for backspace (LA180)
Bit 1	Do not process <BS> as backspace
Bit 2	Allow 8-bit characters (LN01)
Bit 3	Allow nonprinting characters
Bit 4	No fill for <FF>
Bit 5	Allow <EOT>
Bit 6	No <CR> required before <LF>,<VT>,<FF> (LP11)
Bit 7	Ignore <CR> if next is <LF> (LP11)

Note that a value of 3 (set both bits 0 and 1) is illegal. All other combinations are permitted.

FIRQB+FQNAM1+2	Clear characteristics flag bits mask. Zero indicates no change. A nonzero value represents the combination of flag bits to be cleared. See the previous discussion for FIRQB+FQNAM1 for an explanation of the bit meanings.
FIRQB+FQEXT	Change special character flag. Zero indicates no change. One indicates a change to the Lineprinter Special Character (LPTSPC).
FIRQB+FQEXT+1	If FIRQB+FQEXT is equal to one, then this byte contains the new Lineprinter Special Character.
FIRQB+FQSIZ	The default printer form length. Zero indicates no change. A nonzero value is the new default printer form length. The maximum value is 377.
FIRQB+FQDEV	The device name in two ASCII characters; must be LP.
FIRQB+FQDEVN	The line printer unit number on which to perform the operation.

Data Returned (Set LPn: characteristics)

Except for a possible error in byte 0 of the FIRQB, the system returns the following:

```
                                FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
         +--------------------+--------------------+
       1 |////////////////////////////////////////|   0
         +--------------------+--------------------+
       3 |////////////////////////////////////////|   2
         +--------------------+--------------------+
       5 |////////////////////////////////////////|   4
         +--------------------+--------------------+
       7 |      case flag     | default page width |   6  FQPPN
         +--------------------+--------------------+
      11 |    current characteristics flag word    |  10  FQNAM1
         +--------------------+--------------------+
      13 |////////////////////////////////////////|  12
         +--------------------+--------------------+
      15 |current special char.|//////////////////|  14  FQEXT
         +--------------------+--------------------+
      17 |////////////////////| current form length |  16  FQSIZ
         +--------------------+--------------------+
      21 |////////////////////////////////////////|  20
         +--------------------+--------------------+
      23 |////////////////////////////////////////|  22
         +--------------------+--------------------+
      25 |////////////////////////////////////////|  24
         +--------------------+--------------------+
      27 |////////////////////////////////////////|  26
         +--------------------+--------------------+
      31 |////////////////////////////////////////|  30
         +--------------------+--------------------+
      33 |////////////////////////////////////////|  32
         +--------------------+--------------------+
      35 |////////////////////////////////////////|  34
         +--------------------+--------------------+
      37 |////////////////////////////////////////|  36
         +--------------------+--------------------+
```

FIRQB+FQPPN	Current width of the LP unit specified in FIRQB+FQDEV and FIRQB+FQDEVN.
FIRQB+FQPPN+1	Current value of the case flag.
FIRQB+FQNAM1	Current value of the characteristics flag word.
FIRQB+FQEXT+1	Current value of the special character.
FIRQB+FQSIZ	Current value of the default form length.

Errors (Set LPn: characteristics)

BDNERR	An attempt was made to set both bits 0 and 1 in the characteristics flag word.
NODEVC	An attempt was made to set characteristics of a printer which is not supported by the monitor, or the device name specified in FIRQB+FQDEV was not LP.

Data Passed (Set System Defaults)

```
                                FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.CFG (= 42)    |/////////////////////|  2
          +---------------------+---------------------+
        5 |/////////////////////| subfunction code =2 |  4  FQFIL
          +---------------------+---------------------+
        7 |         powerfail delay in seconds        |  6  FQPPN
          +---------------------+---------------------+
       11 | default time format | default date format | 10  FQNAM1
          +---------------------+---------------------+
       13 | default tape density| default tape label  | 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30  FQDEV
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32  FQDEVN
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQPPN	Default powerfail delay in seconds. Zero indicates no change. A nonzero value represents the new default powerfail delay. The maximum is 454 (octal) seconds.

FIRQB+FQNAM1	Default date format. Zero indicates no change. One indicates NUMERIC date format. The ALPHA date format is indicated by 377.
FIRQB+FQNAM1+1	Default time format. Zero indicates no change. One indicates 24 hour time format. The 12 hour (AM/PM) time format is indicated by 377.
FIRQB+FQNAM1+2	Default magnetic tape label. Zero indicates no change. One indicates DOS magnetic tape label. The ANSI magnetic tape label is indicated by 377.
FIRQB+FQNAM1+3	Default magnetic tape density. Zero indicates no change. One indicates 1600 bpi. A density of 800 bpi is indicated by 377.

Data Returned (Set System Defaults)

```
                                FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
           +---------------------+---------------------+
         1 |///////////////////////////////////////////|  0
           +---------------------+---------------------+
         3 |///////////////////////////////////////////|  2
           +---------------------+---------------------+
         5 |///////////////////////////////////////////|  4
           +---------------------+---------------------+
         7 |   seconds to delay on powerfail restarts  |  6 FQPPN
           +---------------------+---------------------+
        11 | default time format | default date format | 10 FQNAM1
           +---------------------+---------------------+
        13 |default tape density | default tape label  | 12
           +---------------------+---------------------+
        15 |///////////////////////////////////////////| 14
           +---------------------+---------------------+
        17 |///////////////////////////////////////////| 16
           +---------------------+---------------------+
        21 |///////////////////////////////////////////| 20
           +---------------------+---------------------+
        23 |///////////////////////////////////////////| 22
           +---------------------+---------------------+
        25 |///////////////////////////////////////////| 24
           +---------------------+---------------------+
        27 |///////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |///////////////////////////////////////////| 30
           +---------------------+---------------------+
        33 |///////////////////////////////////////////| 32
           +---------------------+---------------------+
        35 |///////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |///////////////////////////////////////////| 36
           +---------------------+---------------------+
```

FIRQB+FQPPN	The number of seconds to delay of powerfail restarts.
FIRQB+FQNAM1	Default date format. One indicates NUMERIC date format. The ALPHA date format is indicated by 377.
FIRQB+FQNAM1+1	Default time format. One indicates 24 hour time format. The 12 hour (AM/PM) time format is indicated by 377.
FIRQB+FQNAM1+2	Default magnetic tape label. One indicates DOS magnetic tape label. The ANSI magnetic tape label is indicated by 377.

FIRQB+FQNAM1+3 Default magnetic tape density. One indicates 1600 bpi. A density of 800 bpi is indicated by 377.

Errors (Set System Defaults)

FIXERR An attempt was made to specify a powerfail delay value greater than 454 (octal) seconds.

Data Passed (Load and Remove Monitor Overlay Code)

```
                                 FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
           +---------------------+---------------------+
         1 |///////////////////////////////////////////|  0
           +---------------------+---------------------+
FQFUN    3 |    UU.CFG (= 42)    |/////////////////////|  2
           +---------------------+---------------------+
         5 |/////////////////////| subfunction code =4 |  4 FQFIL
           +---------------------+---------------------+
         7 |/////////////////////| 0 = load, 1 = unload|  6 FQPPN
           +---------------------+---------------------+
        11 |   internal overlay name in RAD50 format   | 10 FQNAM1
           +---------------------+---------------------+
        13 |///////////////////////////////////////////| 12
           +---------------------+---------------------+
        15 |///////////////////////////////////////////| 14
           +---------------------+---------------------+
        17 |///////////////////////////////////////////| 16
           +---------------------+---------------------+
        21 |///////////////////////////////////////////| 20
           +---------------------+---------------------+
        23 |///////////////////////////////////////////| 22
           +---------------------+---------------------+
        25 |///////////////////////////////////////////| 24
           +---------------------+---------------------+
        27 |///////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |///////////////////////////////////////////| 30
           +---------------------+---------------------+
        33 |///////////////////////////////////////////| 32
           +---------------------+---------------------+
        35 |///////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |///////////////////////////////////////////| 36
           +---------------------+---------------------+
```

FIRQB+FQPPN	Operation flag. Zero indicates load the named overlay code. A one indicates unload the named code.
FIRQB+FQNAM1	Internal overlay name in RAD50 format. These names are subject to change in future releases of RSTS/E. Presently, the following names have been defined:

DIR	Handles directory lookups
DLN	Handles delete and rename
LIN	Handles the UU.ATR SYS() call
PFB	Handles several functions used by the DCL command file processor
TRM	Handles the set-terminal-characteristics (UU.TRM) SYS() call.
UUO	Handles UU.TB1, UU.TB2, UU.TB3, UU.FCB, UU.CNV, UU.SYS, and UU.ERR

Data Returned (Load and Remove Monitor Overlay Code)

Except for a possible error in byte 0 of the FIRQB, for a load operation, the system returns the amount of XBUF used (in bytes) in FIRQB+FQPPN. The remove monitor overlay code operation does not return any data.

Errors (Load and Remove Monitor Overlay Code)

INUSE	The overlay code specified is already loaded. (Load)
NOBUFS	There is not enough XBUF available to load the named overlay code. (Load)
NODEVC	The named overlay code is not loadable. (Load)
NOSUCH	The overlay name is invalid. (either)
NOTMNT	The overlay name is not loaded. (Remove)

UU.CHE (Enable/Disable Disk Caching)

Privileges Required

TUNE is required to enable or disable disk caching.

Data Passed

```
                                 FIRQB
Mne-   Octal                                              Octal   Mne-
monic Offset                                              Offset  monic
            +---------------------+---------------------+
          1 |///////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN     3 |    UU.CHE. (= 23)   |/////////////////////|  2
            +---------------------+---------------------+
FQSIZM    5 | cache cluster size  | subfunction code    |  4 FQFIL
            +---------------------+---------------------+
          7 | limit on total number of cache clusters   |  6 FQPPN
            +---------------------+---------------------+
         11 | limit on clusters for directory caching   | 10 FQNAM1
            +---------------------+---------------------+
         13 | limit on clusters for user data caching   | 12
            +---------------------+---------------------+
         15 | controls sm buffers | mode for enbl/disbl | 14 FQEXT
            +---------------------+---------------------+
         17 |///////////////////////////////////////////| 16
            +---------------------+---------------------+
         21 |///////////////////////////////////////////| 20
            +---------------------+---------------------+
         23 |///////////////////////////////////////////| 22
            +---------------------+---------------------+
         25 |///////////////////////////////////////////| 24
            +---------------------+---------------------+
         27 |///////////////////////////////////////////| 26
            +---------------------+---------------------+
         31 |///////////////////////////////////////////| 30
            +---------------------+---------------------+
         33 |///////////////////////////////////////////| 32
            +---------------------+---------------------+
         35 |///////////////////////////////////////////| 34
            +---------------------+---------------------+
         37 |///////////////////////////////////////////| 36
            +---------------------+---------------------+
```

```
FIRQB+FQFIL       0  Enable directory and data caching
                  1  Disable all caching
                200  Return current caching parameters
```

FIRQB+FQEXT	0	Use current setting
	1	Enable caching as specified in file open mode or UFD setting
	100	Cache all data transfers regardless of the file open mode or UFD setting
	200	Disable all caching
FIRQB+FQEXT+1	0	Use current setting
	1	Allow use of small buffer pool (nondata caching option only)
	200	Do not use small buffer pool

Data Returned

```
                                FIRQB
Mne- Octal                                                 Octal  Mne-
monic Offset                                               Offset  monic
           +----------------------+----------------------+
         1 |/////////////////////////////////////////////|  0
           +----------------------+----------------------+
         3 |/////////////////////////////////////////////|  2
           +----------------------+----------------------+
FQSIZM   5 | cache cluster size   |current cache setting|  4 FQFIL
           +----------------------+----------------------+
         7 |  limit on total number of cache clusters    |  6 FQPPN
           +----------------------+----------------------+
        11 |  limit on clusters for directory caching    | 10 FQNAM1
           +----------------------+----------------------+
        13 |  limit on clusters for user data caching    | 12
           +----------------------+----------------------+
        15 |controls sm. buffers  |mod. for en./disable  | 14 FQEXT
           +----------------------+----------------------+
        17 |                  cache age                  | 16 FQSIZ
           +----------------------+----------------------+
        21 |                monitor pool                 | 20 FQNAM2
           +----------------------+----------------------+
        23 |                  XBUF pool                  | 22 FQMODE
           +----------------------+----------------------+
        25 |          number of directory tags           | 24 FQFLAG
           +----------------------+----------------------+
        27 |             number of data tags             | 26 FQPFLG
           +----------------------+----------------------+
        31 |           number of invalid tags            | 30 FQDEV
           +----------------------+----------------------+
        33 |/////////////////////////////////////////////| 32
           +----------------------+----------------------+
        35 |/////////////////////////////////////////////| 34
           +----------------------+----------------------+
        37 |/////////////////////////////////////////////| 36
           +----------------------+----------------------+
```

FIRQB+FQFIL	Current cache settings and available options: 0 Cache disabled, user data caching is not available 1 Cache enabled, user data caching is not available 200 Cache disabled, user data caching is available 201 Cache enabled, user data caching is available
FIRQB+FQSIZ	The time in seconds that an unused cache can be kept in memory.
FIRQB+FQNAM2	The amount of small buffers used for caching (in units of cache clusters).
FIRQB+FQMODE	The amount of XBUF used for caching (in units of cache clusters).
FIRQB+FQFLAG	The number of cache clusters in XBUF that are used for directory caching. This value is valid only if data caching is included.
FIRQB+FQPFLG	The number of cache clusters in XBUF that are used for data caching. This value is valid only if data caching is included.
FIRQB+FQDEV	The number of invalid cache clusters in XBUF. This value is valid only if data caching is included.

Errors

ERRERR	Caching not enabled during system generation.
INUSE	All of the clusters allotted to the cache are in use.
NOROOM	Not enough XBUF space to enable data caching. System manager must allocate at least 2K words.
NOTAVL	Tried to change cluster size while cached file disk transfer in progress; retry.
PRVIOL	Current user does not have TUNE privilege.

.UUO
UU.CHK

Data Returned - Check File Access Rights

```
                                FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                              Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |               access flags                |  4 FQERNO
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

```
FIRQB+FQERNO   The bits are set to indicate access rights as follows:

               Bit 0  Owner rights are granted
               Bit 1  Read access is not allowed
               Bit 2  Write access is not allowed
               Bit 5  Execute access is not allowed
               Bit 6  Accounting rights are granted or the PPN is the
                      same as that of the caller.
               Bit 7  Accounting rights are granted
```

Errors

No errors are possible with this subfunction of UU.CHK.

Data Passed - Convert Privilege Name to Mask

```
                               FIRQB
Mne-   Octal                                               Octal   Mne-
monic Offset                                               Offset  monic
        +--------------------+--------------------+
      1 |/////////////////////////////////////////|   0
        +--------------------+--------------------+
FQFUN 3 |    UU.CHK (= 40)   |////////////////////|   2
        +--------------------+--------------------+
      5 |////////////////////| subfunction code =1 |   4 FQFIL
        +--------------------+--------------------+
      7 |/////////////////////////////////////////|   6
        +--------------------+--------------------+
     11 |     privilege name as a 6-character     |  10 FQNAM1
        +                                         +
     13 |     uppercase ASCII string or "ALL"     |  12
        +                                         +
     15 |             padded with NULLs           |  14
        +--------------------+--------------------+
     17 |/////////////////////////////////////////|  16
        +--------------------+--------------------+
     21 |/////////////////////////////////////////|  20
        +--------------------+--------------------+
     23 |/////////////////////////////////////////|  22
        +--------------------+--------------------+
     25 |/////////////////////////////////////////|  24
        +--------------------+--------------------+
     27 |/////////////////////////////////////////|  26
        +--------------------+--------------------+
     31 |/////////////////////////////////////////|  30
        +--------------------+--------------------+
     33 |/////////////////////////////////////////|  32
        +--------------------+--------------------+
     35 |/////////////////////////////////////////|  34
        +--------------------+--------------------+
     37 |/////////////////////////////////////////|  36
        +--------------------+--------------------+
```

Data Returned - Convert Privilege Name to Mask

```
                                 FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
          +--------------------+--------------------+
        1 |////////////////////////////////////////|  0
          +--------------------+--------------------+
        3 |////////////////////////////////////////|  2
          +--------------------+--------------------+
        5 |////////////////////|     error flag     |  4 FQERNO
          +--------------------+--------------------+
        7 |////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |             privilege mask              | 10 FQNAM1
          +                                         +
       13 |               (4 words)                 | 12
          +                                         +
       15 |                                         | 14
          +                                         +
       17 |                                         | 16
          +--------------------+--------------------+
       21 |////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQERNO   If the passed privilege name was not ALL:

Zero means the current job has the named privilege.

One means the current job does not.

If ALL was passed:

Zero means the current job has all privileges.

One means the current job does not.

FIRQB+FQNAM1	If the passed privilege name was not ALL, the bit corresponding to the named privilege is set. All other bits are clear. If ALL was passed, all privilege bits that currently have meaning are set. All other bits are clear.

Errors

NOSUCH	The name passed in FIRQB+FQNAM1 is not a valid privilege name.

Data Passed - Convert Privilege Mask to Name

```
                              FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
         +---------------------+---------------------+
       1 |/////////////////////////////////////////////| 0
         +---------------------+---------------------+
FQFUN  3 |    UU.CHK (= 40)    |/////////////////////| 2
         +---------------------+---------------------+
       5 |/////////////////////| subfunction code =2 | 4 FQFIL
         +---------------------+---------------------+
       7 |/////////////////////////////////////////////| 6
         +---------------------+---------------------+
      11 |                                           | 10 FQNAM1
         +                                           +
      13 |              privilege mask               | 12
         +                                           +
      15 |                 (4 words)                 | 14
         +                                           +
      17 |                                           | 16
         +---------------------+---------------------+
      21 |/////////////////////////////////////////////| 20
         +---------------------+---------------------+
      23 |/////////////////////////////////////////////| 22
         +---------------------+---------------------+
      25 |/////////////////////////////////////////////| 24
         +---------------------+---------------------+
      27 |/////////////////////////////////////////////| 26
         +---------------------+---------------------+
      31 |/////////////////////////////////////////////| 30
         +---------------------+---------------------+
      33 |/////////////////////////////////////////////| 32
         +---------------------+---------------------+
      35 |/////////////////////////////////////////////| 34
         +---------------------+---------------------+
      37 |/////////////////////////////////////////////| 36
         +---------------------+---------------------+
```

Data Returned - Convert Privilege Mask to Name

```
                                FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////|  0
          +--------------------+--------------------+
        3 |/////////////////////////////////////////|  2
          +--------------------+--------------------+
        5 |/////////////////////////////////////////|  4
          +--------------------+--------------------+
        7 |/////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |                                         | 10 FQNAM1
          +                                         +
       13 |             privilege mask              | 12
          +                                         +
       15 |                (4 words)                | 14
          +                                         +
       17 |                                         | 16
          +--------------------+--------------------+
       21 |                                         | 20 FQNAM2
          +      6-character uppercase ASCII string +
       23 |                                         | 22
          +             padded with NULLs           +
       25 |                                         | 24
          +--------------------+--------------------+
       27 |/////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |/////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |/////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |/////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |/////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQNAM1	The privilege mask is the same as that passed except that any undefined bits and the bit corresponding to the name in FQNAM2 is turned off. As a result, a subsequent call using the returned mask would return the name of the next privilege. This makes it easy to translate an entire mask by repeated calls to this function.
FIRQB+FQNAM2	The name of the privilege corresponding to the first set bit in the mask.

Errors

NOSUCH	The privilege mask in FIRQB+FQNAM1 is zero or specifies only unassigned privilege bits.

UU.CHU (Set Password, Change Password/Quota/Expiration Date, Disable Terminal,Kill Job)

This directive has five subfunctions:

- o Set Long Password
- o Change Quotas (V9.0 format), Expiration Date, and Short Password
- o Change Quotas (pre-V9.0 format), Expiration Date, and Short Password
- o Disable Terminal
- o Kill Job

Privileges Required (Set Long Password)

GACNT or WACNT are required to set passwords.

Data Passed (Set Long Password)

```
                                FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
            +--------------------+--------------------+
          1 |/////////////////////////////////////////|  0
            +--------------------+--------------------+
FQFUN     3 |     UU.CHU (=10)   |////////////////////|  2
            +--------------------+--------------------+
          5 |////////////////////|////////////////////|  4 FQFIL
            +--------------------+--------------------+
          7 |   project number   | programmer number  |  6 FQPPN
            +--------------------+--------------------+
         11 |                                         | 10 FQNAM1
            +                                         +
         13 |                                         | 12
            +                                         +
         15 |                                         | 14
            +                                         +
         17 |              long password              | 16
            +                                         +
         21 |      (14 bytes, padded with NULLs)      | 20
            +                                         +
         23 |                                         | 22
            +                                         +
         25 |                                         | 24
            +-----------------------------------------+
         27 |/////////////////////////////////////////| 26
            +--------------------+--------------------+
         31 |/////////////////////////////////////////| 30
            +--------------------+--------------------+
         33 |/////////////////////////////////////////| 32
            +--------------------+--------------------+
         35 |    (must be 0)     |    (must be 377)   | 34 FQCLUS
            +--------------------+--------------------+
         37 |/////////////////////////////////////////| 36
            +--------------------+--------------------+
```

FIRQB+FQNAM1 If the first byte is zero, the password information is deleted (this turns the account into a nonuser account).

If password lookup is enabled using CREATE/ACCOUNT or SET PASSWORD then the password must consist of six uppercase letters or numbers. Otherwise, it can contain any character except the question mark (?). The monitor also enforces a minimum password length of six characters.

Note

> In this subfunction, the PPN is specified in a different place than in the "Change Short Password/Quota" subfunctions where the PPN is specified in FQNAM1.

Privileges Required (Change Short Password, V9.0 Quotas, or Expiration Date)

GACNT or WACNT are required to set passwords, quotas, or expiration date.

Data Passed (Change Short Password, V9.0 Quotas, or Expiration Date)

```
                              FIRQB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
            +---------------------+---------------------+
         1  |///////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN    3  |     UU.CHU (=10)    |/////////////////////|  2
            +---------------------+---------------------+
         5  |      flag byte      | detached job quota  |  4 FQFIL
            +---------------------+---------------------+
         7  |///////////////////////////////////////////|  6
            +---------------------+---------------------+
        11  |    project number   |  programmer number  | 10 FQNAM1
            +---------------------+---------------------+
        13  |        new password in RAD50 format       | 12
            +                 (2 words)                 +
        15  |             (zero if no change)           | 14
            +---------------------+---------------------+
        17  |           logged-out quota (LSB)          | 16 FQSIZ
            +---------------------+---------------------+
        21  |          account expiration date          | 20 FQBUFL
            +---------------------+---------------------+
        23  |           logged-in quota (LSB)           | 22 FQMODE
            +---------------------+---------------------+
        25  |logged-out quota(MSB)| logged-in quota(MSB)| 24 FQFLAG
            +---------------------+---------------------+
        27  |/////////////////////| =377 to change quota| 26 FQPFLG
            +---------------------+---------------------+
        31  |      device name (2 ASCII characters)     | 30 FQDEV
            +---------------------+---------------------+
        33  |≠0, unit number real | device unit number  | 32 FQDEVN
            +---------------------+---------------------+
        35  |               (must be zero)              | 34 FQCLUS
            +---------------------+---------------------+
        37  |///////////////////////////////////////////| 36
            +---------------------+---------------------+
```

FIRQB+FQFIL	The number of detached jobs allowed under this account.
FIRQB+FQFIL+1	Flag byte. Zero means change the disk quotas using the pre-V9.0 format (see the next subfunction). Nonzero means the following: Bit 0 Change logged-out quota Bit 1 Change logged-in quota Bit 2 Reserved Bit 3 Change detached job quota Bit 4 Reserved Bit 5 Reserved Bit 6 Reserved Bit 7 Change disk quotas using V9.0 format
FIRQB+FQSIZ	The LSB of the logged-out quota.
FIRQB+FQBUFL	Zero means no change. Nonzero means this value is the new expiration date.
FIRQB+FQMODE	The LSB of the logged-in quota.
FIRQB+FQFLAG	The MSB of the logged-in quota.
FIRQB+FQFLAG+1	The MSB of the logged-out quota.

Privileges Required (Change Short Password, pre-V9.0 Quotas or Expiration Date)

GACNT or WACNT are required to set passwords, quotas, or expiration date.

Data Passed (Change Short Password, pre-V9.0 Quotas or Expiration Date)

```
                                FIRQB
Mne-   Octal                                               Octal   Mne-
monic  Offset                                              Offset  monic
            +---------------------+---------------------+
          1 |/////////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN     3 |     UU.CHU (=10)    |/////////////////////|  2
            +---------------------+---------------------+
          5 |/////////////////////////////////////////////|  4
            +---------------------+---------------------+
          7 |/////////////////////////////////////////////|  6
            +---------------------+---------------------+
         11 |    project number   |  programmer number  | 10 FQNAM1
            +---------------------+---------------------+
         13 |        new password in RAD50 format       | 12
            +                 (2 words)                 +
         15 |             (zero if no change)           | 14
            +---------------------+---------------------+
         17 | number of blocks for quota; 0 = unlimited | 16 FQSIZ
            +---------------------+---------------------+
         21 |          account expiration date          | 20 FQBUFL
            +---------------------+---------------------+
         23 |/////////////////////////////////////////////| 22
            +---------------------+---------------------+
         25 |/////////////////////////////////////////////| 24
            +---------------------+---------------------+
         27 |/////////////////////| =377 to change quota| 26 FQPFLG
            +---------------------+---------------------+
         31 |     device name (2 ASCII characters)      | 30 FQDEV
            +---------------------+---------------------+
         33 |≠0, unit number real |  device unit number | 32 FQDEVN
            +---------------------+---------------------+
         35 |                (must be zero)             | 34 FQCLUS
            +---------------------+---------------------+
         37 |/////////////////////////////////////////////| 36
            +---------------------+---------------------+
```

FIRQB+FQBUFL	Zero means no change. Nonzero means this value is the new expiration date.

Privileges Required (Disable Terminal)

HWCTL is required to disable a terminal.

Data Passed (Disable Terminal)

```
                                FIRQB
Mne-  Octal                                            Octal  Mne-
monic Offset                                           Offset monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////| 0
          +---------------------+---------------------+
FQFUN   3 |     UU.CHU (=10)    |/////////////////////| 2
          +---------------------+---------------------+
        5 |/////////////////////|   keyboard number   | 4 FQFIL
          +---------------------+---------------------+
        7 |///////////////////////////////////////////| 6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |     (must = 377)    |     (must = 377)    | 34 FQCLUS
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Privileges Required (Kill Job)

JOBCTL is required to kill a job.

Data Passed (Kill Job)

```
                                FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
            +---------------------+---------------------+
         1  |///////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN    3  |     UU.CHU (=10)    |/////////////////////|  2
            +---------------------+---------------------+
         5  |/////////////////////| job number to kill  |  4  FQFIL
            +---------------------+---------------------+
         7  |///////////////////////////////////////////|  6
            +---------------------+---------------------+
        11  |///////////////////////////////////////////| 10
            +---------------------+---------------------+
        13  |///////////////////////////////////////////| 12
            +---------------------+---------------------+
        15  |///////////////////////////////////////////| 14
            +---------------------+---------------------+
        17  |///////////////////////////////////////////| 16
            +---------------------+---------------------+
        21  |///////////////////////////////////////////| 20
            +---------------------+---------------------+
        23  |///////////////////////////////////////////| 22
            +---------------------+---------------------+
        25  |///////////////////////////////////////////| 24
            +---------------------+---------------------+
        27  |///////////////////////////////////////////| 26
            +---------------------+---------------------+
        31  |///////////////////////////////////////////| 30
            +---------------------+---------------------+
        33  |///////////////////////////////////////////| 32
            +---------------------+---------------------+
        35  |     (must = 377)    |   (must be zero)    | 34  FQCLUS
            +---------------------+---------------------+
        37  |///////////////////////////////////////////| 36
            +---------------------+---------------------+
```

FIRQB+FQFIL   Specify zero to kill current job.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.CHU subfunction does not return any meaningful data.

Errors

BADFUO	The system returns this error for one of the following: o Change password/quota. Device specified is not a disk. o Kill job. Job number is out of range or the job does not exist. o Disable terminal. The keyboard number is: - Greater than number of keyboards on system - A pseudo keyboard - Currently opened or allocated by a job
BADNAM	For change password/quota. Password is too short, or too long, or contains illegal characters.
NOSUCH	For change password/quota. Account not present on disk specified.
NODEVC	For change password/quota. Device specified is not on the system.

UU.CNV (Date and Time Conversion)

Privileges Required

None

Data Passed

```
                               FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.CNV (= 24)    |/////////////////////|  2
          +---------------------+---------------------+
        5 |internal form date (see .DATE);0 = current |  4 FQFIL
          +---------------------+---------------------+
        7 |                 date flag                 |  6 FQPPN
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |internal form time (see .DATE);0 = current | 22 FQMODE
          +---------------------+---------------------+
       25 |                 time flag                 | 24 FQFLAG
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFIL	Zero means use system default format. A value less than zero means use alphabetic date format. A value greater than zero means use ISO numeric date format.
FIRQB+FQFLAG	Zero means use system default format. A value less than zero means use 12-hour time format. A value greater than zero means use 24-hour time format.

Data Returned

```
                              FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                 Offset  monic
          +----------------------+----------------------+
        1 |//////////////////////////////////////////////| 0
          +----------------------+----------------------+
        3 |//////////////////////|   job number * 2     |  2 FQJOB
          +----------------------+----------------------+
        5 |            (same as data passed)            |  4 FQFIL
          +----------------------+----------------------+
        7 |            (same as data passed)            |  6 FQPPN
          +----------------------+----------------------+
       11 |                                             | 10 FQNAM1
          +                                             +
       13 |             date string (padded             | 12
          +                                             +
       15 |             at end with NULLs)              | 14
          +                                             +
       17 |            dd-mmm-yy or yy.mm.dd            | 16
          +                                             +
       21 |                                             | 20
          +----------------------+----------------------+
       23 |            (same as data passed)            | 22 FQMODE
          +----------------------+----------------------+
       25 |            (same as data passed)            | 24 FQFLAG
          +----------------------+----------------------+
       27 |                                             | 26 FQPFLG
          +             time string (padded             +
       31 |                                             | 30
          +             at end with NULLs)              +
       33 |                                             | 32
          +              hh:mm xm or hh:mm              +
       35 |                                             | 34
          +----------------------+----------------------+
       37 |//////////////////////////////////////////////| 36
          +----------------------+----------------------+
```

Errors

No errors are possible; however, if the date or time passed is illegal, random output is generated.

UU.DAL (Deallocate All Devices)

Privileges Required

None

Data Passed

```
                               FIRQB
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////|  0
          +--------------------+--------------------+
FQFUN   3 |    UU.DAL (= 14)   |////////////////////|  2
          +--------------------+--------------------+
        5 |/////////////////////////////////////////|  4
          +--------------------+--------------------+
        7 |/////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |/////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |/////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |/////////////////////////////////////////| 14
          +--------------------+--------------------+
       17 |/////////////////////////////////////////| 16
          +--------------------+--------------------+
       21 |/////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |/////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |/////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |/////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |/////////////////////////////////////////| 30
          +--------------------+--------------------+
       33 |/////////////////////////////////////////| 32
          +--------------------+--------------------+
       35 |/////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |/////////////////////////////////////////| 36
          +--------------------+--------------------+
```

Data Returned

The UU.DAL subfunction does not return any data.

Errors

No errors are possible with the UU.DAL subfunction.

UU.DAT (Change System Date/Time)

Privileges Required

DATES is required to change system date or time.

Data Passed

```
                                FIRQB
Mne- Octal                                              Octal   Mne-
monic Offset                                            Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.DAT (= -16)   |/////////////////////|  2
          +---------------------+---------------------+
        5 |              new current date             |  4 FQFIL
          +---------------------+---------------------+
        7 |              new current time             |  6 FQPPN
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFIL	The system internal format for date is: [(year - 1970) * 1000] + day-within-year
FIRQB+FQPPN	Time is expressed as minutes until midnight, where 1440 equals midnight.

Note

> A value of zero in either of these fields means no change is to be made. See the .DATE directive for a discussion of these formats.

Data Returned

The UU.DAT subfunction does not return any data.

Errors

PRVIOL	Dates prior to 1-Jan-85 cannot be set.

UU.DEA (Deallocate Device)

Privileges Required

None

Data Passed

```
                                FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
         +--------------------+--------------------+
       1 |/////////////////////////////////////////|  0
         +--------------------+--------------------+
FQFUN  3 |   UU.DEA (= 13)    |////////////////////|  2
         +--------------------+--------------------+
       5 |/////////////////////////////////////////|  4
         +--------------------+--------------------+
       7 |/////////////////////////////////////////|  6
         +--------------------+--------------------+
      11 |/////////////////////////////////////////| 10
         +--------------------+--------------------+
      13 |/////////////////////////////////////////| 12
         +--------------------+--------------------+
      15 |/////////////////////////////////////////| 14
         +--------------------+--------------------+
      17 |/////////////////////////////////////////| 16
         +--------------------+--------------------+
      21 |/////////////////////////////////////////| 20
         +--------------------+--------------------+
      23 |/////////////////////////////////////////| 22
         +--------------------+--------------------+
      25 |/////////////////////////////////////////| 24
         +--------------------+--------------------+
      27 |/////////////////////////////////////////| 26
         +--------------------+--------------------+
      31 |  device name (2 ASCII characters) or 0  | 30 FQDEV
         +--------------------+--------------------+
      33 |≠0, unit number real | device unit number | 32 FQDEVN
         +--------------------+--------------------+
      35 |/////////////////////////////////////////| 34
         +--------------------+--------------------+
      37 |/////////////////////////////////////////| 36
         +--------------------+--------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.DEA subfunction does not return any meaningful data.

.UUO
UU.DEA

Errors

NODEVC	The device or device type in the two words at FIRQB+FQDEV is not on the system.

UU.DET (Detach)

Privileges Required

None required to detach the caller's job (depending on quotas). JOBCTL is required to detach another job. EXQTA is required to suppress the detached job quota check.

Data Passed

```
                              FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////|  0
          +-------------------+-------------------+
FQFUN   3 |     UU.DET (= 7)  |///////////////////|  2
          +-------------------+-------------------+
        5 |///////////////////|close flag & job no.|  4 FQFIL
          +-------------------+-------------------+
        7 |///////////////////////////////////////|  6
          +-------------------+-------------------+
       11 |///////////////////////////////////////| 10
          +-------------------+-------------------+
       13 |///////////////////////////////////////| 12
          +-------------------+-------------------+
       15 |///////////////////////////////////////| 14
          +-------------------+-------------------+
       17 |///////////////////////////////////////| 16
          +-------------------+-------------------+
       21 |///////////////////////////////////////| 20
          +-------------------+-------------------+
       23 |///////////////////////////////////////| 22
          +-------------------+-------------------+
       25 |///////////////////////////////////////| 24
          +-------------------+-------------------+
       27 |///////////////////////////////////////| 26
          +-------------------+-------------------+
       31 |///////////////////////////////////////| 30
          +-------------------+-------------------+
       33 |///////////////////////////////////////| 32
          +-------------------+-------------------+
       35 |///////////////////////////////////////| 34
          +-------------------+-------------------+
       37 |///////////////////////////////////////| 36
          +-------------------+-------------------+
```

FIRQB+FQFIL	Flag byte as follows: Bits 0-6 equals the job number to detach. If zero, detach calling job. Bit 7 clear means do not close terminal channels or deallocate terminal. Bit 7 set means close all channels on which terminal is open and deallocate terminal, if allocated.

Note

If a job running under control of an indirect command file is detached using the UU.DET directive, a new job is created at that terminal, logged in to the same account, and execution of the command file continues with the new job. The detached job retains its job number, while the command file resumes under a different job number.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.DET subfunction does not return any meaningful data.

Errors

BADFUO	The current job is already detached.
QUOTA	The detached job quota for the account has been reached.

UU.DIE (System Shutdown)

Privileges Required

SHUTUP privilege is required to shut down the system.

Data Passed

```
                                FIRQB
Mne-   Octal                                                 Octal   Mne-
monic  Offset                                                Offset  monic
                +---------------------+--------------------+
          1     |//////////////////////////////////////////|  0
                +---------------------+--------------------+
FQFUN     3     |    UU.DIE (= -20)   |////////////////////|  2
                +---------------------+--------------------+
          5     |//////////////////////////////////////////|  4
                +---------------------+--------------------+
          7     |//////////////////////////////////////////|  6
                +---------------------+--------------------+
         11     |//////////////////////////////////////////| 10
                +---------------------+--------------------+
         13     |//////////////////////////////////////////| 12
                +---------------------+--------------------+
         15     |//////////////////////////////////////////| 14
                +---------------------+--------------------+
         17     |//////////////////////////////////////////| 16
                +---------------------+--------------------+
         21     |//////////////////////////////////////////| 20
                +---------------------+--------------------+
         23     |//////////////////////////////////////////| 22
                +---------------------+--------------------+
         25     |//////////////////////////////////////////| 24
                +---------------------+--------------------+
         27     |//////////////////////////////////////////| 26
                +---------------------+--------------------+
         31     |//////////////////////////////////////////| 30
                +---------------------+--------------------+
         33     |//////////////////////////////////////////| 32
                +---------------------+--------------------+
         35     |//////////////////////////////////////////| 34
                +---------------------+--------------------+
         37     |//////////////////////////////////////////| 36
                +---------------------+--------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.DIE subfunction does not return any meaningful data.

Errors

BADFUO	One of these conditions has not been met:

- o Only one job can be running
- o Logins must be disabled
- o No disks except the system disk can be mounted
- o No files can be open on the system disk

UU.DIR (Directory Lookup)

Privileges Required

If the device is restricted, DEVICE privilege is required to perform directory or disk lookup on the caller's account. File read or execute access (by protection code, GACNT or WACNT privileges) is required to perform disk directory lookup on another account.

Data Passed - Directory Lookup on Index

```
                               FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
           +----------------------+----------------------+
         1 |/////////////////////////////////////////////|  0
           +----------------------+----------------------+
FQFUN    3 |    UU.DIR (= 17)     |//////////////////////|  2
           +----------------------+----------------------+
         5 |            index of file to read            |  4 FQFIL
           +----------------------+----------------------+
         7 |    project number    |  programmer number   |  6 FQPPN
           +----------------------+----------------------+
        11 |/////////////////////////////////////////////| 10
           +----------------------+----------------------+
        13 |/////////////////////////////////////////////| 12
           +----------------------+----------------------+
        15 |/////////////////////////////////////////////| 14
           +----------------------+----------------------+
        17 |/////////////////////////////////////////////| 16
           +----------------------+----------------------+
        21 |/////////////////////////////////////////////| 20
           +----------------------+----------------------+
        23 |           marked-for-delete flag            | 22 FQMODE
           +----------------------+----------------------+
        25 |/////////////////////////////////////////////| 24
           +----------------------+----------------------+
        27 |/////////////////////////////////////////////| 26
           +----------------------+----------------------+
        31 |      device name (2 ASCII characters)       | 30 FQDEV
           +----------------------+----------------------+
        33 |≠0, unit number real | device unit number    | 32 FQDEVN
           +----------------------+----------------------+
        35 |/////////////////////////////////////////////| 34
           +----------------------+----------------------+
        37 |/////////////////////////////////////////////| 36
           +----------------------+----------------------+
```

FIRQB+FQPPN   Specifies the PPN to look up. For disks, the PPN defaults to the caller's PPN. For DOS format magnetic tape; a zero in FIRQB+FQPPN means look up files regardless of PPN, a nonzero means only files with that

PPN are returned. For ANSI magnetic tape and DECtape, RSTS/E ignores FIRQB+FQPPN.

FIRQB+FQMODE Marked-for-delete flag.

Bit 14 set causes UU.DIR to return information about marked-for-delete files.

Data Returned -- Directory Lookup on Index

```
                              FIRQB
Mne-  Octal                                             Octal  Mne-
monic Offset                                            Offset monic
         +---------------------+---------------------+
       1 |/////////////////////////////////////////////|  0
         +---------------------+---------------------+
       3 |/////////////////////| current job no. * 2 |  2 FQJOB
         +---------------------+---------------------+
       5 |          (same as data passed)            |  4 FQFIL
         +---------------------+---------------------+
       7 |  project number     | programmer number   |  6 FQPPN
         +---------------------+---------------------+
      11 |        file name in RAD50 format          | 10 FQNAM1
         +                (2 words)                  +
      13 |                                           | 12
         +---------------------+---------------------+
      15 |    file type in RAD50 format (1 word)     | 14 FQEXT
         +---------------------+---------------------+
      17 |            LSB of file size               | 16 FQSIZ
         +---------------------+---------------------+
      21 | MSB of file size    |   protection code   | 20 FQNAM2
         +---------------------+---------------------+
      23 |          date of last access              | 22 FQMODE
         +---------------------+---------------------+
      25 |            date of creation               | 24 FQFLAG
         +---------------------+---------------------+
      27 |            time of creation               | 26 FQPFLG
         +---------------------+---------------------+
      31 |          (same as data passed)            | 30 FQDEV
         +---------------------+---------------------+
      33 |          (same as data passed)            | 32 FQDEVN
         +---------------------+---------------------+
      35 |           file cluster size               | 34 FQCLUS
         +---------------------+---------------------+
      37 |     USTAT byte      |no. entries returned | 36 FQNENT
         +---------------------+---------------------+
```

FIRQB+FQMODE System internal format for date is:
FIRQB+FQFLAG

[(year - 1970) * 1000] + day-within-year

FIRQB+FQPFLG	Time is returned as minutes before midnight, where 1440 equals midnight. For look up on DOS format magnetic tape, this field contains the PPN of the file.

Note

See the .DATE directive for a discussion of the date and time formats.

FIRQB+FQNENT+1	The USTAT byte from the UFD Name entry. This byte contains the following internal flag information (for disk only):

Bit	Meaning When Set
1	File is placed
2	Some job has write access now
3	File is open in update mode
4	File is contiguous; no extend allowed
5	No delete or rename allowed
7	File is marked for deletion

Errors

BADDIR	The directory structure is invalid. For magnetic tape, this error usually indicates you are reading an ANSI format tape in DOS mode, or a DOS format tape in ANSI mode.
DEVNFS	The device specified is non-file-structured.
NOSUCH	The account specified does not exist on the device or no more files exist on the account.
xxxxxx	The call also returns device-dependent errors such as NOTMNT (disk pack not mounted).

Data Passed - Special Magnetic Tape Directory Lookup

```
                                FIRQB
Mne-    Octal                                              Octal    Mne-
monic Offset                                              Offset  monic
            +---------------------+---------------------+
          1 |/////////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN     3 |    UU.DIR (= 17)    |/////////////////////|  2
            +---------------------+---------------------+
          5 |            index of file to read            |  4 FQFIL
            +---------------------+---------------------+
          7 |    (= 177777 for magnetic tape lookup)      |  6 FQPPN
            +---------------------+---------------------+
         11 |/////////////////////////////////////////////| 10
            +---------------------+---------------------+
         13 |/////////////////////////////////////////////| 12
            +---------------------+---------------------+
         15 |/////////////////////////////////////////////| 14
            +---------------------+---------------------+
         17 |/////////////////////////////////////////////| 16
            +---------------------+---------------------+
         21 |/////////////////////////////////////////////| 20
            +---------------------+---------------------+
         23 |/////////////////////////////////////////////| 22
            +---------------------+---------------------+
         25 |/////////////////////////////////////////////| 24
            +---------------------+---------------------+
         27 |/////////////////////////////////////////////| 26
            +---------------------+---------------------+
         31 |     device name (2 ASCII characters)        | 30 FQDEV
            +---------------------+---------------------+
         33 | ≠0, unit number real|  device unit number   | 32 FQDEVN
            +---------------------+---------------------+
         35 |/////////////////////////////////////////////| 34
            +---------------------+---------------------+
         37 |/////////////////////////////////////////////| 36
            +---------------------+---------------------+
```

Data Returned - Special Magnetic Tape Directory Lookup

```
                                FIRQB
Mne-   Octal                                                Octal   Mne-
monic  Offset                                               Offset  monic
          +---------------------+---------------------+
       1  |///////////////////////////////////////////|  0
          +---------------------+---------------------+
       3  |/////////////////////| current job no. * 2 |  2 FQJOB
          +---------------------+---------------------+
       5  |           (same as data passed)           |  4 FQFIL
          +---------------------+---------------------+
       7  |           (same as data passed)           |  6 FQPPN
          +---------------------+---------------------+
      11  |        file name in RAD50 format          | 10 FQNAM1
          +                (2 words)                  +
      13  |                                           | 12
          +---------------------+---------------------+
      15  |    file type in RAD50 format (1 word)     | 14 FQEXT
          +---------------------+---------------------+
      17  |///////////////////////////////////////////| 16
          +---------------------+---------------------+
      21  |/////////////////////|   protection code   | 20 FQNAM2
          +---------------------+---------------------+
      23  |             date of creation              | 22 FQMODE
          +---------------------+---------------------+
      25  |///////////////////////////////////////////| 24
          +---------------------+---------------------+
FQPROT 27 |   project number    | programmer number   | 26 FQPFLG
          +---------------------+---------------------+
      31  |           (same as data passed)           | 30 FQDEV
          +---------------------+---------------------+
      33  |           (same as data passed)           | 32 FQDEVN
          +---------------------+---------------------+
      35  |///////////////////////////////////////////| 34
          +---------------------+---------------------+
      37  |/////////////////////|no. entries returned | 36 FQNENT
          +---------------------+---------------------+
```

FIRQB+FQMODE   System internal format for date is:

[(year - 1970) * 1000] + day-within-year

Errors

BADDIR	This error usually indicates you are reading an ANSI format tape in DOS mode, or a DOS format tape in ANSI mode.
NOSUCH	No more files exist on the tape.
DEVNFS	The device specified in the two words at FIRQB+FQDEVN is either non-file-structured or not a magnetic tape.

UU.DLU (Delete User Account)

Privileges Required

GACNT or WACNT is required to delete a user account.

Data Passed

```
                               FIRQB
Mne-   Octal                                            Octal   Mne-
monic  Offset                                           Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |     UU.DLU (= 1)    |/////////////////////|  2
          +---------------------+---------------------+
        5 |/////////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |/////////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |    project number   | programmer number   | 10 FQNAM1
          +---------------------+---------------------+
       13 |/////////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |/////////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |/////////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |/////////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |     device name (2 ASCII characters)      | 30 FQDEV
          +---------------------+---------------------+
       33 | ≠0, unit number real| device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.DLU subfunction does not return any meaningful data.

Errors

DEVNFS	Device is not disk or is a disk open in non-file-structured mode.
FIEXST	The account contains files.
INUSE	A user is currently logged in to the system under the account.
NOSUCH	The specified account does not exist.
PRVIOL	Account specified is either [0,0] or [0,1].

UU.DMP (Snap Shot Dump)

Privileges Required

SYSIO is required to perform a snap shot dump.

Data Passed

```
                                FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
            +-------------------+-------------------+
          1 |///////////////////////////////////////| 0
            +-------------------+-------------------+
FQFUN     3 |    UU.DMP (= -33)  |///////////////////| 2
            +-------------------+-------------------+
          5 |///////////////////////////////////////| 4
            +-------------------+-------------------+
          7 |///////////////////////////////////////| 6
            +-------------------+-------------------+
         11 |///////////////////////////////////////| 10
            +-------------------+-------------------+
         13 |///////////////////////////////////////| 12
            +-------------------+-------------------+
         15 |///////////////////////////////////////| 14
            +-------------------+-------------------+
         17 |///////////////////////////////////////| 16
            +-------------------+-------------------+
         21 |///////////////////////////////////////| 20
            +-------------------+-------------------+
         23 |///////////////////////////////////////| 22
            +-------------------+-------------------+
         25 |///////////////////////////////////////| 24
            +-------------------+-------------------+
         27 |///////////////////////////////////////| 26
            +-------------------+-------------------+
         31 |///////////////////////////////////////| 30
            +-------------------+-------------------+
         33 |///////////////////////////////////////| 32
            +-------------------+-------------------+
         35 |///////////////////////////////////////| 34
            +-------------------+-------------------+
         37 |///////////////////////////////////////| 36
            +-------------------+-------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.DMP subfunction of .UUO does not return any meaningful data.

Errors

BADFUO	A user without SYSIO privilege attempted to execute this call.
NOSUCH	The call tried to write data to the file CRASH.SYS, but crash dump was not enabled. This error occurs if RSTS/E tries to allocate a crash dump file during system start-up but could not due to lack of disk space.
xxxxxx	The call also returns device-dependent errors such as HNGDEV or NOTMNT.

UU.ERR (Return Error Messages)

Privileges Required

None

Data Passed

```
                               FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
           +---------------------+---------------------+
        1  |///////////////////////////////////////////|  0
           +---------------------+---------------------+
FQFUN   3  |     UU.ERR (=11)    |/////////////////////|  2
           +---------------------+---------------------+
        5  |/////////////////////|    error number     |  4 FQERNO
           +---------------------+---------------------+
        7  |///////////////////////////////////////////|  6
           +---------------------+---------------------+
       11  |///////////////////////////////////////////| 10
           +---------------------+---------------------+
       13  |///////////////////////////////////////////| 12
           +---------------------+---------------------+
       15  |///////////////////////////////////////////| 14
           +---------------------+---------------------+
       17  |///////////////////////////////////////////| 16
           +---------------------+---------------------+
       21  |///////////////////////////////////////////| 20
           +---------------------+---------------------+
       23  |///////////////////////////////////////////| 22
           +---------------------+---------------------+
       25  |///////////////////////////////////////////| 24
           +---------------------+---------------------+
       27  |///////////////////////////////////////////| 26
           +---------------------+---------------------+
       31  |///////////////////////////////////////////| 30
           +---------------------+---------------------+
       33  |///////////////////////////////////////////| 32
           +---------------------+---------------------+
       35  |///////////////////////////////////////////| 34
           +---------------------+---------------------+
       37  |///////////////////////////////////////////| 36
           +---------------------+---------------------+
```

Data Returned

```
                              FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
         +--------------------+--------------------+
       1 |/////////////////////////////////////////|   0
         +--------------------+--------------------+
FQFUN  3 |KB*2;1's comp=detach)| current job no. * 2 |   2 FQJOB
         +--------------------+--------------------+
       5 |                                         |   4 FQERNO
         +                                         +
       7 |                                         |   6
         +                                         +
      11 |                                         |  10
         +                                         +
      13 |   error message  --  padded with NULLs  |  12
         +                                         +
      15 |      to 28. characters (ASCII format)   |  14
         +                                         +
      17 |                                         |  16
         +                                         +
      21 |                                         |  20
         +                                         +
      23 |                                         |  22
         +                                         +
      25 |                                         |  24
         +                                         +
      27 |                                         |  26
         +                                         +
      31 |                                         |  30
         +                                         +
      33 |                                         |  32
         +                                         +
      35 |                                         |  34
         +                                         +
      37 |                                         |  36
         +--------------------+--------------------+
```

Errors

No errors are possible with the UU.ERR subfunction.

UU.FCB (Get Open Channel Statistics: WCB/DDB/FCB)

The WCB is the Window Control Block for the file system. The DDB is the Device Data Block for nondisk devices. The FCB is the File Control Block for the file system. Note that the WCB, DDB, and FCB are internal RSTS/E structures that are subject to change at any time.

Privileges Required

None

Data Passed

```
                              FIRQB
Mne-   Octal                                             Octal   Mne-
monic Offset                                             Offset  monic
            +--------------------+--------------------+
          1 |////////////////////////////////////////|  0
            +--------------------+--------------------+
FQFUN     3 |    UU.FCB (=10)    |////////////////////|  2
            +--------------------+--------------------+
FQSIZM    5 |  subcode = 0 or 1  |WCB,DDB,or FCB ch.no.|  4 FQFIL
            +--------------------+--------------------+
          7 |////////////////////////////////////////|  6
            +--------------------+--------------------+
         11 |////////////////////////////////////////| 10
            +--------------------+--------------------+
         13 |////////////////////////////////////////| 12
            +--------------------+--------------------+
         15 |////////////////////////////////////////| 14
            +--------------------+--------------------+
         17 |////////////////////////////////////////| 16
            +--------------------+--------------------+
         21 |////////////////////////////////////////| 20
            +--------------------+--------------------+
         23 |////////////////////////////////////////| 22
            +--------------------+--------------------+
         25 |////////////////////////////////////////| 24
            +--------------------+--------------------+
         27 |////////////////////////////////////////| 26
            +--------------------+--------------------+
         31 |////////////////////////////////////////| 30
            +--------------------+--------------------+
         33 |////////////////////////////////////////| 32
            +--------------------+--------------------+
         35 |////////////////////////////////////////| 34
            +--------------------+--------------------+
         37 |////////////////////////////////////////| 36
            +--------------------+--------------------+
```

FIRQB+FQSIZM	Zero returns WCB or DDB information. One returns FCB information.

Data Returned (for subcode = 0)

```
                               FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                             Offset  monic
          +---------------------+--------------------+
        1 |//////////////////////////////////////////|  0
          +---------------------+--------------------+
        3 |/////////////////////| current job no. * 2 |  2 FQJOB
          +---------------------+--------------------+
        5 |      word 1 of either the WCB or DDB      |  4 FQFIL
          +---------------------+--------------------+
        7 |      word 2 of either the WCB or DDB      |  6 FQPPN
          +---------------------+--------------------+
       11 |      word 3 of either the WCB or DDB      | 10 FQNAM1
          +---------------------+--------------------+
       13 |      word 4 of either the WCB or DDB      | 12
          +---------------------+--------------------+
       15 |      word 5 of either the WCB or DDB      | 14 FQEXT
          +---------------------+--------------------+
       17 |      word 6 of either the WCB or DDB      | 16 FQSIZ
          +---------------------+--------------------+
       21 |      word 7 of either the WCB or DDB      | 20 FQBUFL
          +---------------------+--------------------+
       23 |      word 8 of either the WCB or DDB      | 22 FQMODE
          +---------------------+--------------------+
       25 |      word 9 of either the WCB or DDB      | 24 FQFLAG
          +---------------------+--------------------+
       27 |      word 10 of either the WCB or DDB     | 26 FQPFLG
          +---------------------+--------------------+
       31 |      word 11 of either the WCB or DDB     | 30 FQDEV
          +---------------------+--------------------+
       33 |      word 12 of either the WCB or DDB     | 32 FQDEVN
          +---------------------+--------------------+
       35 |      word 13 of either the WCB or DDB     | 34 FQCLUS
          +---------------------+--------------------+
       37 |      word 14 of either the WCB or DDB     | 36 FQNENT
          +---------------------+--------------------+
```

Data Returned (for subcode = 1)

```
                                FIRQB
Mne-   Octal                                                   Octal   Mne-
monic Offset                                                   Offset  monic
              +--------------------+--------------------+
            1 |/////////////////////////////////////////|  0
              +--------------------+--------------------+
            3 |////////////////////| current job no. * 2 |  2 FQJOB
              +--------------------+--------------------+
FQSIZM      5 |users w/file open rr |users w/file open nrr|  4 FQFIL
              +--------------------+--------------------+
            7 |   MSB of file size  |     Status byte     |  6 FQPPN
              +--------------------+--------------------+
           11 |             LSB of file size             | 10 FQNAM1
              +--------------------+--------------------+
           13 |/////////////////////////////////////////| 12
              +--------------------+--------------------+
           15 |/////////////////////////////////////////| 14
              +--------------------+--------------------+
           17 |/////////////////////////////////////////| 16
              +--------------------+--------------------+
           21 |/////////////////////////////////////////| 20
              +--------------------+--------------------+
           23 |/////////////////////////////////////////| 22
              +--------------------+--------------------+
           25 |/////////////////////////////////////////| 24
              +--------------------+--------------------+
           27 |/////////////////////////////////////////| 26
              +--------------------+--------------------+
           31 |/////////////////////////////////////////| 30
              +--------------------+--------------------+
           33 |/////////////////////////////////////////| 32
              +--------------------+--------------------+
           35 |/////////////////////////////////////////| 34
              +--------------------+--------------------+
           37 |/////////////////////////////////////////| 36
              +--------------------+--------------------+
```

FIRQB+FQPPN This byte contains the following internal flag information:

Bit	Meaning When Set
1	Placed file
2	Some job has write access now
3	File is open in update mode
4	File is contiguous; no extend allowed
5	No delete or rename allowed
6	File is a UFD
7	File is marked for deletion

Errors

BADFUO	The subfunction code passed in FIRQB+FQSIZM is a value other than zero or one.
BSERR	The channel number specified at FIRQB+FQFIL is outside the range 0 through 17.
NODEVC	You requested FCB information (FIRQB+FQSIZM=1), but the file open on the I/O channel specified at FIRQB+FQFIL is not a disk file.
NOTOPN	The I/O channel specified at FIRQB+FQFIL is not open.

UU.FIL (File Placement and Modification)

Privileges Required

Read access (by protection code, GREAD or WREAD privilege) is required to read file flags. Write access (by protection code, GWRITE, WWRITE, and/or SYSIO privilege) is required to set file flags. DATES is required to change last access date. SYSIO is required to set/clear the no-delete flag (P-bit). TUNE is required to set/clear file caching bits.

Data Passed

```
                                 FIRQB
Mne-   Octal                                              Octal   Mne-
monic  Offset                                             Offset  monic
             +--------------------+--------------------+
         1   |/////////////////////////////////////////|   0
             +--------------------+--------------------+
FQFUN    3   |    UU.FIL (= -32)  |////////////////////|   2
             +--------------------+--------------------+
FQSIZM   5   |    function code   | channel no.(1-17)  |   4 FQFIL
             +--------------------+--------------------+
         7   |   project number   | programmer number  |   6 FQPPN
             +--------------------+--------------------+
        11   |        file name in RAD50 format        |  10 FQNAM1
             +               (2 words)                 +
        13   |                                         |  12
             +--------------------+--------------------+
        15   |   file type in RAD50 format (1 word)    |  14 FQEXT
             +--------------------+--------------------+
        17   | least significant bits of block number  |  16 FQSIZ
             +--------------------+--------------------+
        21   | MSB of block number|placed/cache/seq.flg|  20 FQNAM2
             +--------------------+--------------------+
        23   |         new date of last access         |  22 FQMODE
             +--------------------+--------------------+
        25   |           new date of creation          |  24 FQFLAG
             +--------------------+--------------------+
        27   |           new time of creation          |  26 FQPFLG
             +--------------------+--------------------+
        31   | 2-character(ASCII) device name (disk only)|  30 FQDEV
             +--------------------+--------------------+
        33   | ≠0, unit number real| device unit number |  32 FQDEVN
             +--------------------+--------------------+
        35   |////////////////////|  subfunction code  |  34 FQCLUS
             +--------------------+--------------------+
        37   |/////////////////////////////////////////|  36
             +--------------------+--------------------+
```

FIRQB+FQFIL   You can also place a zero in this byte and specify the file by file specification (device, PPN, file name, and type). The system ignores the values in bytes 6 through 15 and 30 through 33 if you use a nonzero channel number.

FIRQB+FQSIZM This byte contains the following internal flags:

Bit	Meaning When Set
0	Set or reset file's placed bit (cannot be used with bit 3).
1	Used with bit 4 to return zero as the device cluster number if file's placed bit is not set.
2	Change file's backup statistics.
3	Change file's run-time system name field.
4	Return file's retrieval information.
5	Reset file's contiguous bit.
6	Enable/disable sequential mode caching if file is cached (cannot be used with bit 3).
7	Enable/disable data caching on the file (cannot be used with bit 3).

FIRQB+FQSIZ If bit 3 of FIRQB+FQSIZM is set, the two words beginning at FIRQB+FQSIZ are the new run-time system name in RAD50.

FIRQB+FQNAM2 This byte contains the following flags:

Bit	Meaning When Set
1	New value for placed bit if bit 1 of FIRQB+FQSIZM is set.
2	New value for sequential bit if bit 6 of FIRQB+FQSIZM is set.
5	New value for P-bit (no delete/rename allowed).
7	New value for cached bit if bit 7 of FIRQB+FQSIZM is set.

FIRQB+FQCLUS Two subfunction codes currently in use are:

Bit	Meaning When Set
0	Change the value of the file's P-bit (see FIRQB+FQNAM2)
1	Do not return protection violation. If clear, and any of the requested functions cannot be executed, the monitor returns a protection violation error. If set, the monitor changes all valid requested functions and leaves the rest unchanged.

Data Returned

```
                                  FIRQB
Mne-   Octal                                           Octal   Mne-
monic Offset                                           Offset  monic
          +---------------------+--------------------+
        1 |//////////////////////////////////////////|  0
          +---------------------+--------------------+
FQFUN   3 |file characteristics | current job no. * 2 |  2 FQJOB
          +---------------------+--------------------+
        5 |          device cluster number           |  4 FQFIL
          +---------------------+--------------------+
        7 |                                          |  6 FQPPN
          +                                          +
       11 |                                          | 10
          +                                          +
       13 |                                          | 12
          +                                          +
       15 |           file attribute data            | 14
          +                                          +
       17 |    (unused words are padded with NULLs)   | 16
          +                                          +
       21 |                                          | 20
          +                                          +
       23 |                                          | 22
          +                                          +
       25 |                                          | 24
          +                                          +
       27 |                                          | 26
          +                                          +
       31 |                                          | 30
          +                                          +
       33 |                                          | 32
          +---------------------+--------------------+
       35 |   run-time system name in RAD50 format    | 34 FQCLUS
          +                 (2 words)                +
       37 |                                          | 36
          +---------------------+--------------------+
```

FIRQB+FQFUN File characteristics:

Bit	Meaning When Set
1	File is placed
2	File will be cached sequentially, if at all
4	File is contiguous
5	Current setting of file's P-bit
7	File will be cached when open

Errors

BADFUO	File open is not a disk file or is a user file directory.
NOSUCH	File or account is not present on disk.
NOTOPN	Channel is not open.
PRVIOL	File open is not a disk file or job lacks necessary privilege (see FIRQB+FQCLUS in data passed).

UU.HNG (Hang Up a Dataset)

Privileges Required

HWCTL is needed to hang up a dataset, except on a keyboard line owned by the calling job.

Data Passed

```
                               FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +----------------------+----------------------+
        1 |/////////////////////////////////////////////|  0
          +----------------------+----------------------+
FQFUN   3 |    UU.HNG (=-11)     |//////////////////////|  2
          +----------------------+----------------------+
FQSIZM  5 |   seconds to wait    |  KB number of line   |  4 FQFIL
          +----------------------+----------------------+
        7 |/////////////////////////////////////////////|  6
          +----------------------+----------------------+
       11 |/////////////////////////////////////////////| 10
          +----------------------+----------------------+
       13 |/////////////////////////////////////////////| 12
          +----------------------+----------------------+
       15 |/////////////////////////////////////////////| 14
          +----------------------+----------------------+
       17 |/////////////////////////////////////////////| 16
          +----------------------+----------------------+
       21 |/////////////////////////////////////////////| 20
          +----------------------+----------------------+
       23 |/////////////////////////////////////////////| 22
          +----------------------+----------------------+
       25 |/////////////////////////////////////////////| 24
          +----------------------+----------------------+
       27 |/////////////////////////////////////////////| 26
          +----------------------+----------------------+
       31 |/////////////////////////////////////////////| 30
          +----------------------+----------------------+
       33 |/////////////////////////////////////////////| 32
          +----------------------+----------------------+
       35 |/////////////////////////////////////////////| 34
          +----------------------+----------------------+
       37 |/////////////////////////////////////////////| 36
          +----------------------+----------------------+
```

FIRQB+FQSIZM	Use one of the following values:

377	Sets "Data Terminal Ready" to permit a modem connected to a RSTS/E system to dial out. If a connection is not made in 127. seconds, perform an automatic hang-up of the dataset.
0	Hang up in two seconds.
1-177	Hang up in 1 to 127. seconds.

Data Returned

The UU.HNG subfunction does not return any data.

Errors

BADFUO	Illegal KB number or insufficient privileges to hang up a line owned by another job.

UU.JOB (Create Job)

Privileges Required

Set account access is required (GACNT or WACNT privilege) to spawn to another account. EXQTA is required to ignore job quota on spawn. JOBCTL is required to spawn a job even if no logins is set. TUNE is required to specify a priority when creating a job. WACNT is required to spawn a logged-out job.

Data Passed - Logged-out job

```
                                FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
           +----------------------+----------------------+
         1 |/////////////////////////////////////////////|  0
           +----------------------+----------------------+
FQFUN    3 |    UU.JOB (= 30)     |//////////////////////|  2
           +----------------------+----------/-----------+
         5 |//////////////////////|      bit flags       |  4 FQFIL
           +----------------------+----------------------+
         7 |   project number     |  programmer number   |  6 FQPPN
           +----------------------+----------------------+
        11 |          file name in RAD50 format          | 10 FQNAM1
           +                  (2 words)                  +
        13 |                                             | 12
           +----------------------+----------------------+
        15 |    file type in RAD50 format (1 word)       | 14 FQEXT
           +----------------------+----------------------+
        17 |                                             | 16 FQSIZ
           +                                             +
        21 |                10 bytes of                  | 20
           +             information to be               +
        23 |          placed in created job's            | 22
           +             core common area                +
        25 |                                             | 24
           +                                             +
        27 |                                             | 26
           +----------------------+----------------------+
        31 |     device name (2 ASCII characters)        | 30 FQDEV
           +----------------------+----------------------+
        33 | ≠0,unit number real  | device unit number   | 32 FQDEVN
           +----------------------+----------------------+
        35 |        parameter word (see P.RUN)           | 34 FQCLUS
           +----------------------+----------------------+
        37 |/////////////////////////////////////////////| 36
           +----------------------+----------------------+
```

FIRQB+FQFIL	The following bit flag is valid for a logged-out job: **Bit** — **Meaning When Set** 7 — Create a job even if logins are disabled (JOBCTL privilege required). When this bit is not set, the job is created only if logins are enabled. If you do not have JOBCTL privilege, you must clear bit 7. All other bits must be 0.
FIRQB+FQPPN FIRQB+FQNAM1 FIRQB+FQEXT FIRQB+FQDEV FIRQB+FQDEVN	File specification of program to run.

Data Passed -- Logged-in job to run a program

FIRQB

Mnemonic	Octal Offset			Octal Offset	Mnemonic
	1	//////////////////////	//////////////////////	0	
FQFUN	3	UU.JOB (= 30)	//////////////////////	2	
FQSIZM	5	KB to attach to	bit flags	4	FQFIL
	7	project number	programmer number	6	FQPPN
	11	file name in RAD50 format (2 words)		10	FQNAM1
	13			12	
	15	file type in RAD50 format (1 word)		14	FQEXT
	17	project number	programmer number	16	FQSIZ
	21	run burst	priority	20	FQBUFL
	23	//////////////////////	maximum job size	22	FQMODE
	25	//////////////////////	//////////////////////	24	
	27	//////////////////////	//////////////////////	26	
	31	device name (2 ASCII characters)		30	FQDEV
	33	≠0,unit number real	device unit number	32	FQDEVN
	35	parameter word (see P.RUN)		34	FQCLUS
	37	//////////////////////	//////////////////////	36	

Note

When you create a logged-in job, the system passes your current job's core common and user logicals to the new job, regardless of privilege. The system temporarily stores this data in XBUF.

FIRQB+FQFIL	The following bit flags are valid when you create a logged-in job to run a program.

Bit	Meaning When Set
1	Do not pass the user logicals of the creating account to the new job. (The contents of core common are still passed.)
2	Create the new job with no privileges.
3	Create the new job with the authorized privileges of the new job equal to the authorized privileges of the job's account that are also current privileges of the calling job. If Bit 3 is clear, the new job is created with the authorized privileges of the job's account.
5	Create a job to run under account specified at FIRQB+FQSIZ (GACNT or WACNT privilege required).
6	Create a logged-in job. You must set this bit. The new job runs under the caller's account unless you also set bit 5.
7	Create a job even if logins are disabled (JOBCTL privilege required). When this bit is not set, the job is created only if logins are enabled. If you do not have JOBCTL privilege, you must clear bit 7.

Set all bits you do not use to zero.

FIRQB+FQSIZM	Bits 0-6 indicate keyboard number to attach to. To specify KB0:, set bit 7 to one. A zero byte indicates a detached job.
FIRQB+FQPPN FIRQB+FQNAM1 FIRQB+FQEXT FIRQB+FQDEV FIRQB+FQDEVN	File specification of program to run.
FIRQB+FQSIZ	Account under which job runs (GACNT or WACNT privilege required; bit 5 at FIRQB+FQFIL must be set). If you do not have the necessary privileges, set both bytes to 0.
FIRQB+FQBUFL FIRQB+FQMODE	If you specify zero, the system uses the caller's values. Use one to explicitly specify priority zero. Only users with GACNT or WACNT privileges can specify these values. If you do not have the necessary privileges, set all three bytes to zero.

Data Passed -- Logged-in Job to Enter a Keyboard Monitor at P.NEW

```
                              FIRQB
Mne-   Octal                                                  Octal   Mne-
monic  Offset                                                 Offset  monic
          +---------------------+---------------------+
       1  |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN  3  |    UU.JOB (= 30)    |//////////////////////|  2
          +---------------------+---------------------+
FQSIZM 5  |  KB to attach to    |      bit flags       |  4 FQFIL
          +---------------------+---------------------+
       7  |/////////////////////////////////////////////|  6 FQPPN
          +---------------------+---------------------+
      11  |   run-time system name in RAD50 format     | 10 FQNAM1
          +                (2 words)                  +
      13  |                                            | 12
          +---------------------+---------------------+
      15  |/////////////////////////////////////////////| 14 FQEXT
          +---------------------+---------------------+
      17  |   project number    | programmer number    | 16 FQSIZ
          +---------------------+---------------------+
      21  |      run burst      |       priority       | 20 FQBUFL
          +---------------------+---------------------+
      23  |/////////////////////|  maximum job size    | 22 FQMODE
          +---------------------+---------------------+
      25  |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
      27  |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
      31  |/////////////////////////////////////////////| 30
          +---------------------+---------------------+
      33  |/////////////////////////////////////////////| 32
          +---------------------+---------------------+
      35  |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
      37  |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Note

When you create a logged-in job, the system passes your current job's core common and user logicals to the new job, regardless of privilege. The system temporarily stores this data in XBUF.

FIRQB+FQFIL	The following bit flags are valid when you create a logged-in job to enter a keyboard monitor.

Bit	Meaning When Set
1	Do not pass the user logicals of the creating account to the new job. (The contents of core common are still passed.)
2	Create the new job with no privileges.
3	Create the new job with the authorized privileges of the new job equal to the authorized privileges of the job's account that are also current privileges of the calling job. If Bit 3 is clear, the new job is created with the authorized privileges of the job's account.
4	Enter keyboard monitor instead of running a program. (You must set this bit.)
5	Create a job to run under account specified at FIRQB+FQSIZ (GACNT or WACNT privilege required).
6	Create a logged-in job. (You must set this bit.) The new job runs under the caller's account unless you also set bit 5.
7	Create a job even if logins are disabled (JOBCTL privilege required). When this bit is not set, the job is created only if logins are enabled. If you do not have JOBCTL privilege, you must clear bit 7.

Set all bits you do not use to zero.

FIRQB+FQSIZM	Bits 0-6 indicate keyboard number to attach to. To specify KB0:, set Bit 7 equal to one. The value of zero (detached job) is not allowed.
FIRQB+FQNAM1	The run-time system you specify must be installed and must be a keyboard monitor. Specify zero for the system's default keyboard monitor (DCL).
FIRQB+FQSIZ	Account under which job runs (GACNT or WACNT privilege required; bit 5 at FIRQB+FQFIL must be set). If you do not have the necessary privileges, set both bytes to zero.
FIRQB+FQBUFL FIRQB+FQMODE	If you specify zero, the system uses the caller's values. Use one to explicitly specify priority zero. Only users with GACNT or WACNT privileges can specify these values. If you do not have the necessary privileges, set all three bytes to zero.

Data Returned (all types of jobs)

```
                                 FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                                Offset  monic
            +---------------------+---------------------+
          1 |///////////////////////////////////////////|  0
            +---------------------+---------------------+
          3 |/////////////////////| current job no. * 2 |  2 FQJOB
            +---------------------+---------------------+
          5 |/////////////////////| created job no. * 2 |  4 FQFIL
            +---------------------+---------------------+
          7 |///////////////////////////////////////////|  6
            +---------------------+---------------------+
         11 |///////////////////////////////////////////| 10
            +---------------------+---------------------+
         13 |///////////////////////////////////////////| 12
            +---------------------+---------------------+
         15 |///////////////////////////////////////////| 14
            +---------------------+---------------------+
         17 |///////////////////////////////////////////| 16
            +---------------------+---------------------+
         21 |///////////////////////////////////////////| 20
            +---------------------+---------------------+
         23 |///////////////////////////////////////////| 22
            +---------------------+---------------------+
         25 |///////////////////////////////////////////| 24
            +---------------------+---------------------+
         27 |///////////////////////////////////////////| 26
            +---------------------+---------------------+
         31 |///////////////////////////////////////////| 30
            +---------------------+---------------------+
         33 |///////////////////////////////////////////| 32
            +---------------------+---------------------+
         35 |///////////////////////////////////////////| 34
            +---------------------+---------------------+
         37 |///////////////////////////////////////////| 36
            +---------------------+---------------------+
```

Errors

BADFUO	You specified a keyboard monitor at FIRQB+FQNAM1, but did not supply a keyboard number at FIRQB+FQSIZM.
NOBUFS	You are trying to create a logged-in job, but not enough XBUF is available for temporary storage of your current job's core common and user logicals.
NODEVC	The keyboard number at FIRQB+FQSIZM is invalid.

NOROOM	Job cannot be created. Probable causes: o Logins are disabled and bit 7 (value = 200) at FIRQB+FQFIL is clear. o The system's job or swap slots are currently full.
NOSUCH	You have the necessary privilege and are trying to create a logged-in job, but the system cannot log the job in. Possible causes are that you specified a nonexistent account or an expired account.
NOTAVL	The keyboard specified at FIRQB+FQSIZM is open, is in use, or is not assigned to the calling job. A user without DEVICE privilege can also get this error if the system manager has restricted the device.
PRVIOL	You do not have the required privileges and tried to: o Create a job when logins were disabled. o Create a logged-out job. o Create a job to run under an account other than the current account. o Create a detached job that would cause you to exceed your detached job quota. (This quota is set by your system manager; its default value is zero.)

Note that when you create a logged-in job, you can also get any error that can occur for the UU.LIN (Login), .RUN, and .RTS directives, because the monitor executes these directives for the new job. If an error occurs, the monitor kills the new job and returns the error to your job instead.

UU.LIN (Login)

Privileges Required

Set account access (GACNT or WACNT) is required to check the password of any account. Also, DEVICE is required if the device is restricted.

Data Passed -- Login Function

```
                                  FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
            +---------------------+---------------------+
         1  |///////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN    3  |    UU.LIN (= 4)     |/////////////////////|  2
            +---------------------+---------------------+
FQSIZM   5  |    function flag    |/////////////////////|  4
            +---------------------+---------------------+
         7  |   project number    |  programmer number  |  6  FQPPN
            +---------------------+---------------------+
        11  |                                           | 10  FQNAM1
            +          password in RAD50 format         +
        13  |                 (2 words)                 | 12
            +                                           +
        15  |                    or                     | 14
            +                                           +
        17  |          password in ASCII format         | 16
            +                (14 bytes)                 +
        21  |                                           | 20
            +                                           +
        23  |                                           | 22
            +                                           +
        25  |                                           | 24
            +-------------------------------------------+
        27  |///////////////////////////////////////////| 26
            +-------------------------------------------+
        31  |///////////////////////////////////////////| 30
            +-------------------------------------------+
        33  |///////////////////////////////////////////| 32
            +-------------------------------------------+
        35  |///////////////////////////////////////////| 34
            +-------------------------------------------+
        37  |///////////////////////////////////////////| 36
            +-------------------------------------------+
```

FIRQB+FQSIZM The function flag bits are:

Bit	Setting	Meaning
0	Clear	Must be clear for the login function.
1	Clear	The monitor assumes the password is six characters in RAD50 format. The password is converted to ASCII before comparison.
	Set	The monitor assumes the password is 14 bytes of ASCII data.

In either case, the password must not contain a question mark (?).

Bit	Setting	Meaning
3	Clear	The monitor performs the password check. No privileges are required.
	Set	The monitor suppresses the password check. However, GACNT or WACNT is required.

Data Returned

```
                                FIRQB
Mne-   Octal                                              Octal   Mne-
monic  Offset                                             Offset  monic
          +--------------------+--------------------+
        1 |///////////////////////////////////////////| 0
          +--------------------+--------------------+
FQFUN   3 |     status flags    | current job no. * 2 | 2 FQJOB
          +--------------------+--------------------+
        5 | total number of jobs under this account  | 4 FQFIL
          +--------------------+--------------------+
        7 |detached job no. (2) |detached job no. (1) | 6 FQPPN
          +--------------------+--------------------+
       11 |                   .                      | 10 FQNAM1
          +                    .                     +
       13 |                   .                      | 12
          +                                          +
       15 |                                          | 14
          +        zero byte marks end of list       +
       17 |         of detached job numbers.         | 16
          +                                          +
       21 |        only the first 25. detached       | 20
          +         job numbers are returned.        +
       23 |                                          | 22
          +                                          +
       25 |                                          | 24
          +                                          +
       27 |                                          | 26
          +                                          +
       31 |                                          | 30
          +                                          +
       33 |                                          | 32
          +                                          +
       35 |                                          | 34
          +                                          +
       37 |                                          | 36
          +--------------------+--------------------+
```

FIRQB+FQFUN — On successful return, indicates where the log-in operation has been completed. Values are:

Value	Meaning
-1	Success -- the job is logged in.
-2	Detached job quota exceeded. Job is not logged in; attach instead to some job from the list starting at offset FQNAM1
≥0	Total job quota exceeded. Job is not logged in; the value returned is the actual job quota. If there are any detached jobs, you can attach to one.

Errors

BADDIR — The account does not have all the necessary directory structures.

NOBUFS — No buffers are available to create the necessary internal structures.

NOSUCH — The monitor returns this error code if:

- The password specified at FIRQB+FQNAM1 does not match the password of the account on the system or contains question marks.
- The PPN specified at FIRQB+FQPPN does not exist or the project number is zero.
- System password validation is required for this type of job, and the account is marked "no-lookup," but the "verify system password" function has not yet been successfully executed.

NOTAVL — The monitor returns this error code if:

- The PPN specified at FIRQB+FQPPN is expired or is marked as "no-interactive."
- The account is flagged as "no-dial-up" or "no-network" and the job is a dial-up or network job.

Data Passed - Verify Password Function

```
                                   FIRQB
Mne-   Octal                                               Octal   Mne-
monic  Offset                                              Offset  monic
           +---------------------+---------------------+
         1 |/////////////////////////////////////////////|  0
           +---------------------+---------------------+
FQFUN    3 |     UU.LIN (= 4)    |/////////////////////|  2
           +---------------------+---------------------+
FQSIZM   5 |    function flag    |/////////////////////|  4
           +---------------------+---------------------+
         7 |   project number    |  programmer number  |  6  FQPPN
           +---------------------+---------------------+
        11 |                                           | 10  FQNAM1
           +         password in RAD50 format          +
        13 |                (2 words)                  | 12
           +                                           +
        15 |                   or                      | 14
           +                                           +
        17 |         password in ASCII format          | 16
           +                (14 bytes)                 +
        21 |                                           | 20
           +                                           +
        23 |                                           | 22
           +                                           +
        25 |                                           | 24
           +-------------------------------------------+
        27 |/////////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |    device name (2 ASCII characters)       | 30  FQDEV
           +---------------------+---------------------+
        33 | ≠0,unit number real |  device unit number | 32  FQDEVN
           +---------------------+---------------------+
        35 |/////////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |/////////////////////////////////////////////| 36
           +---------------------+---------------------+
```

FIRQB+FQSIZM   The function flag bits:

Bit	Setting	Meaning
0	Set	The verify password function is performed. The caller supplies the password to verify in FIRQB+FQNAM1.
1	Clear	The monitor assumes the password is six characters in RAD50 format. The password is converted to ASCII data before verification.

Bit	Setting	Meaning
1	Set	The monitor assumes the password is 14 bytes of ASCII data.

In either case, the password must not contain any question marks.

FIRQB+FQPPN	A value of zero means use the caller's PPN.
FIRQB+FQDEV FIRQB+FQDEVN	Device name of the disk on which to check. You must have DEVICE privilege if the device is restricted.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.LIN subfunction does not return any meaningful data.

Errors

BADNAM	Passwords do not match.
DEVNFS	Device specified is not a disk.
NOSUCH	Account specified is a nonuser account (no password data present) or does not exist.
PRVIOL	Caller does not have required privilege.

Verify System Password Function

This function verifies the system access password, if any. If there is no system access password, or if the password check has already been performed, the function returns successfully. Otherwise, the monitor compares the password supplied with the system password. A match sets a job flag which is tested by the login function; a mismatch generates the NOSUCH error.

For compatibilty with old programs that perform a password lookup and then login, the login function does not require a system password on accounts where password lookup is allowed.

Data Passed - Verify System Password Function

```
                                 FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |      UU.LIN (= 4)   |/////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 |  function flag = 6  |/////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |                                           | 10 FQNAM1
          +                                           +
       13 |                                           | 12
          +                                           +
       15 |                                           | 14
          +                                           +
       17 |         password in ASCII format          | 16
          +                (14 bytes)                 +
       21 |                                           | 20
          +                                           +
       23 |                                           | 22
          +                                           +
       25 |                                           | 24
          +-------------------------------------------+
       27 |///////////////////////////////////////////| 26
          +-------------------------------------------+
       31 |///////////////////////////////////////////| 30
          +-------------------------------------------+
       33 |///////////////////////////////////////////| 32
          +-------------------------------------------+
       35 |///////////////////////////////////////////| 34
          +-------------------------------------------+
       37 |///////////////////////////////////////////| 36
          +-------------------------------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.LIN subfunction does not return any meaningful data.

Errors

NOSUCH	Password did not match the system password.

UU.LOG (Set Number of Logins)

Privileges Required

SWCTL is required to set the number of logins.

Data Passed

```
                                FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +------------------+------------------+
        1 |//////////////////////////////////////| 0
          +------------------+------------------+
FQFUN   3 |    UU.LOG (= -23)  |///////////////////| 2
          +------------------+------------------+
        5 |///////////////////| no. of allowed jobs | 4 FQFIL
          +------------------+------------------+
        7 |//////////////////////////////////////| 6
          +------------------+------------------+
       11 |//////////////////////////////////////| 10
          +------------------+------------------+
       13 |//////////////////////////////////////| 12
          +------------------+------------------+
       15 |//////////////////////////////////////| 14
          +------------------+------------------+
       17 |//////////////////////////////////////| 16
          +------------------+------------------+
       21 |//////////////////////////////////////| 20
          +------------------+------------------+
       23 |//////////////////////////////////////| 22
          +------------------+------------------+
       25 |//////////////////////////////////////| 24
          +------------------+------------------+
       27 |//////////////////////////////////////| 26
          +------------------+------------------+
       31 |//////////////////////////////////////| 30
          +------------------+------------------+
       33 |//////////////////////////////////////| 32
          +------------------+------------------+
       35 |//////////////////////////////////////| 34
          +------------------+------------------+
       37 |//////////////////////////////////////| 36
          +------------------+------------------+
```

FIRQB+FQFIL	A value of zero sets the number of allowed jobs to one. The upper limit for the number of logins is either the system JOB MAX or the number of jobs that can currently be swapped, whichever is lower. If you specify a larger value, the system sets the number of logins to the upper limit. In this case, no error is returned.

Data Returned

```
                                 FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|   0
          +---------------------+---------------------+
        3 |/////////////////////| current job no. * 2 |   2 FQJOB
          +---------------------+---------------------+
        5 |/////////////////////|  actual logins set  |   4 FQFIL
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|   6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////|  10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////|  12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////|  14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////|  16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////|  20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////|  22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////|  24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////|  26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////|  30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////|  32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////|  34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////|  36
          +---------------------+---------------------+
```

Errors

No errors are possible with the UU.LOG subfunction.

UU.LOK (Disk Directory Lookup by File Name/Wildcard Lookup)

Privileges Required

DEVICE is required to do a directory lookup on a restricted device. Read or execute access (by protection code, GREAD or WREAD privilege) is required to lookup another account.

Data Passed - Look up by File Name

```
                              FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.LOK (= 21)    |/////////////////////|  2
          +---------------------+---------------------+
        5 |              (must = 177777)              |  4 FQFIL
          +---------------------+---------------------+
        7 |   project number    | programmer number   |  6 FQPPN
          +---------------------+---------------------+
       11 |         file name in RAD50 format         | 10 FQNAM1
          +                  (2 words)                +
       13 |                                           | 12
          +---------------------+---------------------+
       15 |     file type in RAD50 format (1 word)    | 14 FQEXT
          +---------------------+---------------------+
       17 |/////////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |/////////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |  device name (disk) (2 ASCII characters)  | 30 FQDEV
          +---------------------+---------------------+
       33 | ≠0, unit number real| device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQPPN	Zero in both bytes means the current account.

Data Returned - Look up by File Name

```
                               FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
            +---------------------+---------------------+
        1   |///////////////////////////////////////////|  0
            +---------------------+---------------------+
        3   |/////////////////////| current job no. * 2 |  2 FQJOB
            +---------------------+---------------------+
        5   |               (= 177777)                  |  4 FQFIL
            +---------------------+---------------------+
        7   |   project number    | programmer number   |  6 FQPPN
            +---------------------+---------------------+
       11   |         file name in RAD50 format         | 10 FQNAM1
            +                 (2 words)                 +
       13   |                                           | 12
            +---------------------+---------------------+
       15   |     file type in RAD50 format (1 word)    | 14 FQEXT
            +---------------------+---------------------+
       17   |  LSB of file length in 512-byte blocks    | 16 FQSIZ
            +---------------------+---------------------+
       21   | MSB of file length  |   protection code   | 20 FQNAM2
            +---------------------+---------------------+
       23   |            date of last access            | 22 FQMODE
            +---------------------+---------------------+
       25   |             date of creation              | 24 FQFLAG
            +---------------------+---------------------+
       27   |             time of creation              | 26 FQPFLG
            +---------------------+---------------------+
       31   | device name (disk) (2 ASCII characters)   | 30 FQDEV
            +---------------------+---------------------+
       33   | ≠0, unit number real| device unit number  | 32 FQDEVN
            +---------------------+---------------------+
       35   |             file cluster size             | 34 FQCLUS
            +---------------------+---------------------+
       37   |         file identification index         | 36 FQNENT
            +---------------------+---------------------+
```

FIRQB+FQMODE
FIRQB+FQFLAG   System internal format for dates is:

[(year - 1970.) * 1000.)] + day-within-year

FIRQB+FQPFLG   Time is in minutes until midnight, where 1440. equals midnight.

See the .DATE directive for a discussion of these formats.

Errors

BADNAM	The file name at FIRQB+FQNAM1 is missing or invalid.
NOSUCH	Device specified at FIRBQ+FQDEV is not a disk, or the file specified does not exist on the disk. This error also occurs when a user does not have read or execute access to the specified file.
PAKLCK	The disk is restricted and the account under which the call is executed does not have DEVICE privilege.

Data Passed -- Disk Wildcard Directory Lookup

```
                               FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////|  0
          +-------------------+-------------------+
FQFUN   3 |    UU.LOK (= 21)  |///////////////////|  2
          +-------------------+-------------------+
        5 | index:n means search for n+1 occurrences |  4 FQFIL
          +-------------------+-------------------+
        7 |   project number  | programmer number |  6 FQPPN
          +-------------------+-------------------+
       11 |    wildcard file name in RAD50 format |  10 FQNAM1
          +              (2 words)                +
       13 |                                       |  12
          +-------------------+-------------------+
       15 |wildcard file type in RAD50 format (1 word)| 14 FQEXT
          +-------------------+-------------------+
       17 |///////////////////////////////////////|  16
          +-------------------+-------------------+
       21 |///////////////////////////////////////|  20
          +-------------------+-------------------+
       23 |///////////////////|marked-for-delete flg| 22 FQMODE
          +-------------------+-------------------+
       25 |///////////////////////////////////////|  24
          +-------------------+-------------------+
       27 |///////////////////////////////////////|  26
          +-------------------+-------------------+
       31 | device name (disk) (2 ASCII characters) | 30 FQDEV
          +-------------------+-------------------+
       33 | ≠0, unit number real| device unit number | 32 FQDEVN
          +-------------------+-------------------+
       35 |///////////////////////////////////////|  34
          +-------------------+-------------------+
       37 |///////////////////////////////////////|  36
          +-------------------+-------------------+
```

FIRQB+FQPPN   Zero in both bytes means the current account.

FIRQB+FQMODE   Bit 14 set causes the monitor to return information about marked-for-delete files.

Data Returned - Disk Wildcard Directory Lookup

```
                              FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +----------------------+----------------------+
        1 |//////////////////////////////////////////////|  0
          +----------------------+----------------------+
        3 |//////////////////////| current job no. * 2  |  2 FQJOB
          +----------------------+----------------------+
        5 |            (same as data passed)            |  4 FQFIL
          +----------------------+----------------------+
        7 |   project number     |  programmer number   |  6 FQPPN
          +----------------------+----------------------+
       11 |         file name in RAD50 format           | 10 FQNAM1
          +                  (2 words)                  +
       13 |                                             | 12
          +----------------------+----------------------+
       15 |     file type in RAD50 format (1 word)      | 14 FQEXT
          +----------------------+----------------------+
       17 |             LSB of file length              | 16 FQSIZ
          +----------------------+----------------------+
       21 | MSB of file length   |   protection code    | 20 FQNAM2
          +----------------------+----------------------+
       23 |      date of last access (disk only)        | 22 FQMODE
          +----------------------+----------------------+
       25 |              date of creation               | 24 FQFLAG
          +----------------------+----------------------+
       27 |              time of creation               | 26 FQPFLG
          +----------------------+----------------------+
       31 |            (same as data passed)            | 30 FQDEV
          +----------------------+----------------------+
       33 |            (same as data passed)            | 32 FQDEVN
          +----------------------+----------------------+
       35 |        file cluster size (disk only)        | 34 FQCLUS
          +----------------------+----------------------+
       37 |      USTAT byte      |//////////////////////| 36 FQNENT
          +----------------------+----------------------+
```

FIRQB+FQNENT+1	This byte contains the following internal flag information:

Bit	Meaning When Set
1	File is placed
2	Some job has write access now
3	File is open in update mode
4	File is contiguous; no extend allowed
5	No delete or rename allowed
7	File is marked for deletion

Errors

BADNAM	The file name at FIRQB+FQNAM1 is missing or invalid.
NOSUCH	Device specified is not a disk or no file corresponds to the index given.
PAKLCK	The disk is restricted and the account under which the call is executed does not have DEVICE privilege.

UU.MNT (Disk Pack Status)

Privileges Required

HWCFG is required to declare a mounted disk as restricted or unrestricted. MOUNT is required to:

- o Mount or dismount a shared disk
- o Dismount a /NOSHARE disk owned by another job
- o Mount a disk /NOSHARE for a job running under another PPN
- o Mount a dirty disk

Data Passed

```
                                FIRQB
Mne-   Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////|    0
          +-------------------+-------------------+
FQFUN   3 |    UU.MNT (= 3)   |///////////////////|    2
          +-------------------+-------------------+
FQSIZM  5 |     job number    |  subfunction code |    4  FQFIL
          +-------------------+-------------------+
        7 |///////////////////////////////////////|    6
          +-------------------+-------------------+
       11 | pack identification label in RAD50 format |  10  FQNAM1
          +               (2 words)               +
       13 |                                       |   12
          +-------------------+-------------------+
       15 |  =177777, use logical; =0, use pack ID |  14  FQEXT
          +-------------------+-------------------+
       17 |  logical name for disk in RAD50 format |  16  FQSIZ
          +            (first 2 words)            +
       21 |                                       |   20
          +-------------------+-------------------+
       23 |               mode word               |   22  FQMODE
          +-------------------+-------------------+
       25 | 3rd word of logical name for disk (RAD50) | 24  FQFLAG
          +-------------------+-------------------+
       27 |///////////////////////////////////////|   26
          +-------------------+-------------------+
       31 |    device name (2 ASCII characters)   |   30  FQDEV
          +-------------------+-------------------+
       33 | ≠0, unit number real| device unit number |  32  FQDEVN
          +-------------------+-------------------+
       35 |///////////////////////////////////////|   34
          +-------------------+-------------------+
       37 |///////////////////////////////////////|   36
          +-------------------+-------------------+
```

FIRQB+FQFIL	The subfunction codes are:

Code	Meaning
0	Mount a disk pack or cartridge
2	Dismount a disk pack or cartridge
4	Restrict a disk pack or cartridge
6	Unrestrict a disk pack or cartridge

FIRQB+FQSIZM	Job number of the intended owner. Applies only if bit 11 is set in FIRQB+FQMODE. Zero means the caller's job number. If bit 11 in FIRQB+FQMODE is clear, this byte should be zero.
FIRQB+FQNAM1	For mount function: pack ID of pack to mount. Must be supplied unless bit 10 is set in FQMODE.
FIRQB+FQMODE	The bits in the mode word are:

Bit	Meaning
8	Mount Level 1.2 disk with /NOQUOTA
10	Mount with lookup of pack ID
11	Mount disk for use by single user(/NOSHARE)
12	Mount disk read/write, even if disk was initialized as read-only
13	Mount disk read-only
14	Mount as a private disk (/PRIVATE)
15	Must be set if any of the above bits are set

You can set one bit or a combination of bits.

Data Returned

```
                              FIRQB
Mne-  Octal                                         Octal   Mne-
monic Offset                                         Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////|  0
          +-------------------+-------------------+
        3 |///////////////////| current job no. * 2 |  2 FQJOB
          +-------------------+-------------------+
        5 |///////////////////////////////////////|  4
          +-------------------+-------------------+
        7 |///////////////////////////////////////|  6
          +-------------------+-------------------+
       11 |///////////////////////////////////////| 10
          +-------------------+-------------------+
       13 |///////////////////////////////////////| 12
          +-------------------+-------------------+
       15 |///////////////////////////////////////| 14
          +-------------------+-------------------+
       17 |   logical name for disk in RAD50 format   | 16 FQSIZ
          +             (first 2 words)             +
       21 |                                       | 20
          +-------------------+-------------------+
       23 |///////////////////////////////////////| 22
          +-------------------+-------------------+
       25 | 3rd word of logical name for disk (RAD50) | 24 FQFLAG
          +-------------------+-------------------+
       27 |///////////////////////////////////////| 26
          +-------------------+-------------------+
       31 |///////////////////////////////////////| 30
          +-------------------+-------------------+
       33 |///////////////////////////////////////| 32
          +-------------------+-------------------+
       35 |///////////////////////////////////////| 34
          +-------------------+-------------------+
       37 |///////////////////////////////////////| 36
          +-------------------+-------------------+
```

Errors

BADFUO	You tried to mount a disk that is already mounted or that resides in a nondismounted drive, or drive specified is the system disk.
BADPAK	Disk directory structure is invalid. For example, the cluster size is larger than 16 or the storage allocation table is unreadable.
DEVNFS	You tried to restrict, unrestrict, or dismount a disk currently open in non-file-structured mode.

HNGDEV	You tried to mount a disk that is not write-enabled.
INTPAK	The storage allocation table on the disk needs to be rebuilt because the disk was not properly dismounted when it was last used. Before using the disk, use the ONLCLN program to rebuild the storage allocation table. **Note** The MOUNT command automatically performs the rebuild operation if it is needed. Note that when this error occurs, the mount succeeds, but the disk is always mounted read-only with the "dirty" bit set. (You can see if this bit is set by looking at the output of the SHOW DISK command which reports that the disk is "dirty.")
INUSE	You tried to dismount a disk with an open file.
NODEVC	Device type or unit at FIRQB+FQDEV is not on the system.
NOTMNT	You tried to lock, unlock, or dismount a disk that is not mounted.
PRVIOL	You tried to mount a disk that does not contain the RSTS/E file structure. Use the INITIALIZE command, the online DSKINT program, or the DSKINT initialization option to initialize the disk.
WRGPAK	You tried to mount a disk with an incorrect pack label.

UU.NAM (Associate a Run-Time System with a File)

Privileges Required

File write access (by protection code or GWRITE, WWRITE, and/or SYSIO privilege) is required to set the run-time system name for a file.

Data Passed

```
                                FIRQB
Mne-   Octal                                             Octal   Mne-
monic  Offset                                            Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////|   0
          +--------------------+--------------------+
FQFUN   3 |   UU.NAM (= -21)   |////////////////////|   2
          +--------------------+--------------------+
FQSIZM  5 |       run-time     |  channel number    |   4 FQFIL
          +                    +--------------------+
        7 |       system name in RAD50 format       |   6 FQPPN
          +--------------------+                    +
       11 |////////////////////|      (2 words)     |  10
          +--------------------+--------------------+
       13 |/////////////////////////////////////////|  12
          +--------------------+--------------------+
       15 |/////////////////////////////////////////|  14
          +--------------------+--------------------+
       17 |/////////////////////////////////////////|  16
          +--------------------+--------------------+
       21 |/////////////////////////////////////////|  20
          +--------------------+--------------------+
       23 |/////////////////////////////////////////|  22
          +--------------------+--------------------+
       25 |/////////////////////////////////////////|  24
          +--------------------+--------------------+
       27 |/////////////////////////////////////////|  26
          +--------------------+--------------------+
       31 |/////////////////////////////////////////|  30
          +--------------------+--------------------+
       33 |/////////////////////////////////////////|  32
          +--------------------+--------------------+
       35 |/////////////////////////////////////////|  34
          +--------------------+--------------------+
       37 |/////////////////////////////////////////|  36
          +--------------------+--------------------+
```

Data Returned

Other than a possible error in byte 0 of the FIRQB, the UU.NAM subfunction does not return any meaningful data.

Errors

NOTOPN	Channel is not open.
PRVIOL	File open on channel is not a disk file or the job executing the call does not have write access.

UU.NLG (Disable Further Logins)

Privileges Required

SWCTL is required to disables logins.

Data Passed

```
                                 FIRQB
Mne-   Octal                                             Octal   Mne-
monic  Offset                                            Offset  monic
             +-------------------+-------------------+
          1  |///////////////////////////////////////|  0
             +-------------------+-------------------+
FQFUN     3  |    UU.NLG (= -2)   |//////////////////|  2
             +-------------------+-------------------+
          5  |///////////////////////////////////////|  4
             +-------------------+-------------------+
          7  |///////////////////////////////////////|  6
             +-------------------+-------------------+
         11  |///////////////////////////////////////| 10
             +-------------------+-------------------+
         13  |///////////////////////////////////////| 12
             +-------------------+-------------------+
         15  |///////////////////////////////////////| 14
             +-------------------+-------------------+
         17  |///////////////////////////////////////| 16
             +-------------------+-------------------+
         21  |///////////////////////////////////////| 20
             +-------------------+-------------------+
         23  |///////////////////////////////////////| 22
             +-------------------+-------------------+
         25  |///////////////////////////////////////| 24
             +-------------------+-------------------+
         27  |///////////////////////////////////////| 26
             +-------------------+-------------------+
         31  |///////////////////////////////////////| 30
             +-------------------+-------------------+
         33  |///////////////////////////////////////| 32
             +-------------------+-------------------+
         35  |///////////////////////////////////////| 34
             +-------------------+-------------------+
         37  |///////////////////////////////////////| 36
             +-------------------+-------------------+
```

Data Returned

The UU.NLG subfunction does not return any meaningful data.

Errors

No errors are possible with the UU.NLG subfunction.

UU.ONX (Open Next Disk File)

This call opens a disk file or a series of disk files matching a wildcard specification. UU.ONX requires an I/O channel to use and grants access to the file.

When you attempt an open next on a closed channel, UU.ONX finds the first file that matches the specification. When you attempt an open next on an open channel, UU.ONX finds the next file that matches the specification. If there are no more files that match the specification, UU.ONX closes the channel.

Privileges Required

DEVICE is required to perform an open next on a restricted disk. File read access (by protection code, or GREAD or WREAD privilege) is required to perform an open next for read access. File write access (by protection code, or GWRITE, WWRITE, and/or SYSIO privilege) is required to perform an open next for write access.

Data Passed

```
                              FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.ONX (= 41)    |/////////////////////|  2
          +---------------------+---------------------+
        5 |            channel number * 2             |  4 FQFIL
          +---------------------+---------------------+
        7 |    project number   | programmer number   |  6 FQPPN
          +---------------------+---------------------+
       11 |        file name in RAD50 format          | 10 FQNAM1
          +                (2 words)                  +
       13 |                                           | 12
          +---------------------+---------------------+
       15 |   file type in RAD50 format (1 word)      | 14 FQEXT
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |                 OPEN mode                 | 22 FQMODE
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 | device name (2 ASCII chars) must be disk  | 30 FQDEV
          +---------------------+---------------------+
       33 |  ≠0, unit no. real  | device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQPPN	The file owner's PPN. Zero indicates the current account. Wildcards are not permitted in the specification.

.UUO
UU.ONX

FIRQB+FQSIZ	The LSB of the file's size in 512-byte blocks. (See FIRQB+FQSIZM.)
FIRQB+FQBUFL	The date of last access to the file.
FIRQB+FQMODE	The date of creation of the file.
FIRQB+FQFLAG	The time of creation of the file.
FIRQB+FQPFLG	The file cluster size, modulo 256. That is, a file cluster size of 256 is indicated by a value of zero.
FIRQB+FQPROT	The protection code of the file.
FIRQB+FQDEV	The device name, as two ASCII characters.
FIRQB+FQDEVN	The device unit number.
FIRQB+FQCLUS	The file identification index of the file. This word can be used instead of the file name in subsequent opens of the file. The file identification index is provided for compatibility with RSX.
FIRQB+FQNENT	The description of the device just opened. The low byte contains the device's handler index. The high byte contains a set of status flags to allow for device-independent I/O routines. (See the OPNFQ subfunction of CALFIP.)

Errors

BADFUO	The parameters passed in the call are inconsistent with the currently open channel.
DEVNFS	You tried to open a device that is not a disk.
HNGDEV	Device is hung or write-locked.
NOSUCH	No more files match the specification. The channel is closed.
NOTMNT	The device is not mounted.
PAKLCK	The disk pack is restricted. You need DEVICE privilege to open a restricted disk.

UU.PAS (Create User Account)

Privileges Required

Set account access (GACNT or WACNT) is required to create an account.

Data Passed (V9.0 Format)

```
                                 FIRQB
Mne- Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |     UU.PAS (=0)     |/////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 |        flags        | preextnd UFD clstrs |  4 FQFIL
          +---------------------+---------------------+
        7 |    device cluster number for UFD or -1    |  6 FQPPN
          +---------------------+---------------------+
       11 |    project number   |  programmer number  | 10 FQNAM1
          +---------------------+---------------------+
       13 |                                           | 12
          +   password (2 words in RAD50 format)      +
       15 |                                           | 14
          +---------------------+---------------------+
       17 |      LSB of logged-out quota (blocks)     | 16 FQSIZ
          +---------------------+---------------------+
       21 |              expiration date              | 20 FQBUFL
          +---------------------+---------------------+
       23 |      LSB of logged-in quota (blocks)      | 22 FQMODE
          +---------------------+---------------------+
       25 | MSB logged-out quota| MSB logged-in quota | 24 FQFLAG
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26 FQPFLG
          +---------------------+---------------------+
       31 |      device name (2 ASCII characters)     | 30 FQDEV
          +---------------------+---------------------+
       33 |  ≠0, unit no. real  |  device unit number | 32 FQDEVN
          +---------------------+---------------------+
       35 | UFD cluster size (0=use pack cluster size)| 34 FQCLUS
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36 FQNENT
          +---------------------+---------------------+
```

FIRQB+FQFIL	Zero means preextend one cluster. 1-7 means preextend specified number of clusters.
FIRQB+FQSIZM	Flag byte. Bit 7 set means data passed in the FIRQB is in the new format. If bit 7 is clear, the data is in the V8.0 format. Other bits are reserved.

FIRQB+FQPPN	Minus one means place UFD in the middle of the disk.
FIRQB+12	Specify zero to create an account without setting a password. (Used to set long passwords; the password is set with a subsequent call to UU.CHU.)
FIRQB+FQSIZ FIRQB+FQFLAG+1	The logged-out quota in blocks. The LSB are in FIRQB+FQSIZ and the MSB are in FIRQB+FQFLAG+1. Use 377 in all three bytes for an unlimited quota. Zero in all three bytes means no blocks can be allocated.
FIRQB+FQBUFL	Minus one means the account does not expire. This is the default.
FIRQB+FQMODE FIRQB+FQFLAG	The logged-in quota in blocks. The LSB are in FIRQB+FQMODE and the MSB are in FIRQB+FQFLAG. Use 377 in all three bytes for an unlimited quota. Zero in all three bytes means no blocks can be allocated.

Data Returned

```
                               FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
          +-------------------+-------------------+
        1 |///////////////////////////////////////|   0
          +-------------------+-------------------+
        3 |///////////////////|  job number * 2   |   2  FQJOB
          +-------------------+-------------------+
        5 |///////////////////////////////////////|   4
          +-------------------+-------------------+
        7 |///////////////////////////////////////|   6
          +-------------------+-------------------+
       11 |///////////////////////////////////////|  10
          +-------------------+-------------------+
       13 |///////////////////////////////////////|  12
          +-------------------+-------------------+
       15 |///////////////////////////////////////|  14
          +-------------------+-------------------+
       17 |///////////////////////////////////////|  16
          +-------------------+-------------------+
       21 |///////////////////////////////////////|  20
          +-------------------+-------------------+
       23 |device cluster number for cluster 0 of UFD |  22  FQMODE
          +-------------------+-------------------+
       25 |device cluster number for cluster 1 of UFD |  24  FQFLAG
          +-------------------+-------------------+
       27 |device cluster number for cluster 2 of UFD |  26  FQPFLG
          +-------------------+-------------------+
       31 |device cluster number for cluster 3 of UFD |  30  FQDEV
          +-------------------+-------------------+
       33 |device cluster number for cluster 4 of UFD |  32  FQDEVN
          +-------------------+-------------------+
       35 |device cluster number for cluster 5 of UFD |  34  FQCLUS
          +-------------------+-------------------+
       37 |device cluster number for cluster 6 of UFD |  36  FQNENT
          +-------------------+-------------------+
```

Data Passed (Pre-V9.0 Format)

```
                                FIRQB
Mne-   Octal                                              Octal   Mne-
monic Offset                                              Offset  monic
              +--------------------+--------------------+
        1     |//////////////////////////////////////////|  0
              +--------------------+--------------------+
        3     |     UU.PAS (=0)    |/////////////////////|  2
              +--------------------+--------------------+
FQSIZM  5     |        flags       | preextnd UFD clstrs |  4 FQFIL
              +--------------------+--------------------+
        7     |    device cluster number for UFD or -1   |  6 FQPPN
              +--------------------+--------------------+
       11     |    project number  | programmer number   | 10 FQNAM1
              +--------------------+--------------------+
       13     |                                          | 12
              +   password (2 words in RAD50 format)     +
       15     |                                          | 14
              +--------------------+--------------------+
       17     | number of blocks for quota; 0 = unlimited| 16 FQSIZ
              +--------------------+--------------------+
       21     |              expiration date             | 20 FQBUFL
              +--------------------+--------------------+
       23     |//////////////////////////////////////////| 22
              +--------------------+--------------------+
       25     |//////////////////////////////////////////| 24
              +--------------------+--------------------+
       27     |//////////////////////////////////////////| 26
              +--------------------+--------------------+
       31     |      device name (2 ASCII characters)    | 30 FQDEV
              +--------------------+--------------------+
       33     |  ≠0, unit no. real | device unit number  | 32 FQDEVN
              +--------------------+--------------------+
       35     | UFD cluster size (0=use pack cluster size)| 34 FQCLUS
              +--------------------+--------------------+
       37     |//////////////////////////////////////////| 36
              +--------------------+--------------------+
```

FIRQB+FQFIL	Zero means preextend one cluster. 1-7 means preextend specified number of clusters.
FIRQB+FQSIZM	Flag byte. Bit 7 set means data passed in the FIRQB is in the new format. If bit 7 is clear, the data is in the V8.0 format. Other bits are reserved.
FIRQB+FQPPN	Minus one means place UFD in the middle of the disk.

Data Returned

Same as for the V9.0 format.

Errors

ABORT	The account has been entered, and the directory has been preextended. However, the account has not been given a password or quota because an internal consistency check has failed. Submit an SPR if you get this error, and include: o A copy of the disk. o A snap shot dump. (Use the DUMP/SYSTEM command to copy the contents of memory to CRASH.SYS, then run the ANALYS program to save the information. Submit a listing that contains the ANALYS program's output.) See the *RSTS/E System Manager's Guide* for more information on ANALYS.
BADCLU	Cluster size is either greater than 16 or is nonzero and less than the pack cluster size.
BADCNT	The number of clusters specified at FIRQB+FQFIL: o Is out of range. o When added to the device cluster number specified at FIRQB+FQPPN yields an effective address that is greater than the disk size. (See Appendix B for disk size information.)
DEVNFS	Device is not a disk or the disk is open in non-file-structured mode.
FIEXST	Account specified already exists on device.
NOROOM	The monitor cannot allocate one cluster for the UFD you are creating because the disk is full.
PRVIOL	PPN is [0,0], or either the project number or programmer number is 255.

UU.POK (Poke Memory)

Privileges Required

SYSMOD is required to write into kernel memory.

Data Passed

```
                                FIRQB
Mne-  Octal                                                    Octal   Mne-
monic Offset                                                   Offset  monic
           +---------------------+---------------------+
         1 |///////////////////////////////////////////|  0
           +---------------------+---------------------+
FQFUN    3 |    UU.POK (= -6)    |/////////////////////|  2
           +---------------------+---------------------+
         5 |       address of word to be changed       |  4 FQFIL
           +---------------------+---------------------+
         7 |           new contents of word            |  6 FQPPN
           +---------------------+---------------------+
        11 |///////////////////////////////////////////| 10
           +---------------------+---------------------+
        13 |///////////////////////////////////////////| 12
           +---------------------+---------------------+
        15 |///////////////////////////////////////////| 14
           +---------------------+---------------------+
        17 |///////////////////////////////////////////| 16
           +---------------------+---------------------+
        21 |///////////////////////////////////////////| 20
           +---------------------+---------------------+
        23 |///////////////////////////////////////////| 22
           +---------------------+---------------------+
        25 |///////////////////////////////////////////| 24
           +---------------------+---------------------+
        27 |///////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |///////////////////////////////////////////| 30
           +---------------------+---------------------+
        33 |///////////////////////////////////////////| 32
           +---------------------+---------------------+
        35 |///////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |///////////////////////////////////////////| 36
           +---------------------+---------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.POK subfunction does not return any meaningful data.

Errors

PRVIOL	The job executing the call does not have SYSMOD privilege or the address specified is odd.

UU.PPN (Wildcard PPN Lookup)

Privileges Required

DEVICE is required to perform a lookup on a restricted device.

Data Passed

```
                                   FIRQB
Mne-  Octal                                                   Octal   Mne-
monic Offset                                                   Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////| 0
          +---------------------+---------------------+
FQFUN   3 |    UU.PPN (=31)     |/////////////////////| 2
          +---------------------+---------------------+
        5 | index: n means search for n+1 occurrence  | 4 FQFIL
          +---------------------+---------------------+
        7 |  377 or proj. no.   |  377 or prog. no.   | 6 FQPPN
          +---------------------+---------------------+
       11 |/////////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |/////////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |/////////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |/////////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |/////////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 | device name (2 ASCII chars)- must be disk | 30 FQDEV
          +---------------------+---------------------+
       33 |  ≠0, unit no. real  | device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQPPN   377 represents a wildcard. Use zero in both bytes to indicate the current account.

Data Returned

```
                                 FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
          +----------------------+----------------------+
        1 |/////////////////////////////////////////////|  0
          +----------------------+----------------------+
        3 |//////////////////////| current job no. * 2  |  2 FQJOB
          +----------------------+----------------------+
        5 |/////////////////////////////////////////////|  4
          +----------------------+----------------------+
        7 |   project number     |  programmer number   |  6 FQPPN
          +----------------------+----------------------+
       11 |/////////////////////////////////////////////| 10
          +----------------------+----------------------+
       13 |/////////////////////////////////////////////| 12
          +----------------------+----------------------+
       15 |/////////////////////////////////////////////| 14
          +----------------------+----------------------+
       17 |/////////////////////////////////////////////| 16
          +----------------------+----------------------+
       21 |/////////////////////////////////////////////| 20
          +----------------------+----------------------+
       23 |/////////////////////////////////////////////| 22
          +----------------------+----------------------+
       25 |/////////////////////////////////////////////| 24
          +----------------------+----------------------+
       27 |/////////////////////////////////////////////| 26
          +----------------------+----------------------+
       31 |           (same as data passed)             | 30 FQDEV
          +----------------------+----------------------+
       33 |           (same as data passed)             | 32 FQDEVN
          +----------------------+----------------------+
       35 |/////////////////////////////////////////////| 34
          +----------------------+----------------------+
       37 |/////////////////////////////////////////////| 36
          +----------------------+----------------------+
```

Errors

NOSUCH	Device is not a disk, or no match exists for the specified index.
PAKLCK	The disk is restricted and the caller does not have DEVICE privilege.

UU.PRI (Change Priority/Run Burst/Job Size)

Privileges Required

TUNE is required to change any job's priority, run burst, or size.

Data Passed

```
                              FIRQB
Mne-   Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |   UU.PRI (= -15)    |/////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 | ≠0, change priority | job no.;377 = caller|  4 FQFIL
          +---------------------+---------------------+
        7 | ≠0, change burst    |  new run priority   |  6 FQPPN
          +---------------------+---------------------+
       11 |    ≠0, change size  |    new run burst    | 10 FQNAM1
          +---------------------+---------------------+
       13 |/////////////////////|    maximum size     | 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQPPN	Run priority values range from -200 to 170 in steps of 10. The value -200 suspends the job.
FIRQB+FQNAM1	The run burst should be a value from 1 to 177. If you use a value outside this range, the monitor sets the run burst to 6.

FIRQB+FQNAM1+2	The maximum size can range from 1 to 377 (1K-word units). The system uses SWAP MAX if you exceed that value.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.PRI subfunction does not return any meaningful data.

Errors

BADFUO	The job number specified at FIRQB+FQFIL does not exist.

UU.PRV (Set/Clear/Read Current Privileges)

The UU.PRV directive lets you read the current privilege mask or selectively set and/or clear bits in it. This function is distinct from the .SET or .CLEAR directives, which apply only when running a privileged program and then update the entire current mask at once.

Privileges Required

None

Data Passed

```
                                  FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.PRV (= 34)    |/////////////////////|  2
          +---------------------+---------------------+
        5 |                                           |  4 FQFIL
          +            bits to be set in the          +
        7 |            current privilege mask         |  6
          +                 (4 words)                 +
       11 |                                           | 10
          +                                           +
       13 |                                           | 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |                                           | 20 FQNAM2
          +           bits to be cleared in the       +
       23 |            current privilege mask         | 22
          +                 (4 words)                 +
       25 |                                           | 24
          +                                           +
       27 |                                           | 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFIL	The bits to be set in the current privilege mask. If you do not want to set any privileges, use four words of zero. Before setting the specified bits, the system performs an AND operation between the bits the caller passes and the caller's authorized privilege mask to prevent the caller from setting unauthorized bits. Note that this applies even for temporary privileges. Thus, a temporarily privileged program can use UU.PRV to drop any single privilege, but it can regain only those that the caller is authorized to have.
FIRQB+FQNAM2	The bits to be cleared in the current privilege mask. If you do not want to clear any privileges, use four words of zero.

Note

If all eight words are zero, no bits are set or cleared. Thus, the job's privileges remain unchanged. This is the way to read your current privileges.

Data Returned

```
                            FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +--------------------+--------------------+
       1  |//////////////////////////////////////////|  0
          +--------------------+--------------------+
       3  |//////////////////////////////////////////|  2
          +--------------------+--------------------+
       5  |                                          |  4 FQFIL
          +                                          +
       7  |          current privilege mask          |  6
          +                (4 words)                 +
      11  |                                          | 10
          +                                          +
      13  |                                          | 12
          +--------------------+--------------------+
      15  |//////////////////////////////////////////| 14
          +--------------------+--------------------+
      17  |//////////////////////////////////////////| 16
          +--------------------+--------------------+
      21  |//////////////////////////////////////////| 20
          +--------------------+--------------------+
      23  |//////////////////////////////////////////| 22
          +--------------------+--------------------+
      25  |//////////////////////////////////////////| 24
          +--------------------+--------------------+
      27  |//////////////////////////////////////////| 26
          +--------------------+--------------------+
      31  |//////////////////////////////////////////| 30
          +--------------------+--------------------+
      33  |//////////////////////////////////////////| 32
          +--------------------+--------------------+
      35  |//////////////////////////////////////////| 34
          +--------------------+--------------------+
      37  |//////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQFIL   The bits set represent the privileges now in effect.

Note

> A privileged program that wants to find out what privileges the user has needs to do a drop temporary privileges function followed by a read privileges function.
>
> Conversely, a privileged program that wants to check whether the program has the right privileges should issue the read privileges function while temporary privileges are in effect.

UU.RAD (Read or Read-and-Reset Accounting Data)

Privileges Required

None required to read your own account. Set account access (GACNT or WACNT) is required for all other accounts, or to reset accounting data.

Data Passed

```
                                  FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
           +---------------------+---------------------+
         1 |///////////////////////////////////////////|  0
           +---------------------+---------------------+
FQFUN    3 |   UU.RAD (= 16)     |/////////////////////|  2
           +---------------------+---------------------+
         5 |          index number of account          |  4 FQFIL
           +---------------------+---------------------+
         7 |       0 = read; ≠0 = read and reset       |  6 FQPPN
           +---------------------+---------------------+
        11 |  project number     |  programmer number  | 10 FQNAM1
           +---------------------+---------------------+
        13 |/////////////////////|        flag         | 12
           +---------------------+---------------------+
        15 |///////////////////////////////////////////| 14
           +---------------------+---------------------+
        17 |///////////////////////////////////////////| 16
           +---------------------+---------------------+
        21 |///////////////////////////////////////////| 20
           +---------------------+---------------------+
        23 |///////////////////////////////////////////| 22
           +---------------------+---------------------+
        25 |///////////////////////////////////////////| 24
           +---------------------+---------------------+
        27 |///////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |  device name (disk) (2 ASCII characters)  | 30 FQDEV
           +---------------------+---------------------+
        33 | ≠0, unit no. real   |  device unit number | 32 FQDEVN
           +---------------------+---------------------+
        35 |///////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |///////////////////////////////////////////| 36
           +---------------------+---------------------+
```

FIRQB+FQPPN	If the caller does have account access privilege, the system ignores this word and performs a read operation only.

FIRQB+FQNAM1+2 Flag byte.

Bit	Setting	Meaning
0	Clear	Implies the call returns the number of blocks owned by the account.
	Set	Implies do not return this data. Note that setting this bit to one can save considerable execution time (for Level 1.1 or older disk structures) if you do not need the disk usage information otherwise returned to the word at FIRQB+FQPPN.
1	Clear	Means the PPN at FIRQB+FQNAM1 corresponds to a real account, if index value at FIRQB+FQFIL is zero. If index value is not zero, the system uses the index value and ignores any PPN at FIRQB+FQNAM1.
	Set	Means the PPN at FIRQB+FQNAM1 contains a wildcard (377).
2	Clear	Means the call returns general accounting information in the pre-V9.0 format.
	Set	Means return disk quota information in the V9.0 format.
3	Clear	Means the call is a read-only lookup on the caller's own PPN if the caller does not have GACNT or WACNT privileges.
	Set	Means UU.RAD returns a protection violation error if a caller tries to perform a restricted function without having the required privileges.

Data Returned (Bit 2 set in FIRQB+FQNAM1+2)

```
                                  FIRQB
Mne-  Octal                                                   Octal   Mne-
monic Offset                                                  Offset  monic
             +---------------------+---------------------+
           1 |/////////////////////////////////////////////|  0
             +---------------------+---------------------+
           3 |//////////////////////| current job no. * 2 |  2 FQJOB
             +---------------------+---------------------+
           5 |            (same as data passed)            |  4 FQFIL
             +---------------------+---------------------+
           7 |/////////////////////////////////////////////|  6
             +---------------------+---------------------+
          11 |              PPN of account read            | 10 FQNAM1
             +---------------------+---------------------+
          13 |          LSB of logged-out quota            | 12
             +---------------------+---------------------+
          15 |          LSB of logged-in quota             | 14 FQEXT
             +---------------------+---------------------+
          17 | MSB logged-out quota| MSB logged-in quota |  16 FQSIZ
             +---------------------+---------------------+
          21 |  MSB current usage  |//////////////////////| 20 FQBUFL
             +---------------------+---------------------+
          23 |/////////////////////////////////////////////| 22
             +---------------------+---------------------+
          25 |            LSB of current usage             | 24 FQFLAG
             +---------------------+---------------------+
FQPROT    27 |   count of open files and logged in jobs   | 26 FQPFLG
             +---------------------+---------------------+
          31 |            (same as data passed)            | 30 FQDEV
             +---------------------+---------------------+
          33 |            (same as data passed)            | 32 FQDEVN
             +---------------------+---------------------+
          35 |/////////////////////////////////////////////| 34
             +---------------------+---------------------+
          37 |/////////////////////////////////////////////| 36
             +---------------------+---------------------+
```

FIRQB+FQPFLG   Count of open files is in bits 0-9. The number of jobs is in bits 10-15.

Data Returned (Bit 2 clear in FIRQB+FQNAM1+2)

```
                                FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
          +--------------------+--------------------+
        1 |//////////////////////////////////////////|  0
          +--------------------+--------------------+
        3 |////////////////////| current job no. * 2 |  2 FQJOB
          +--------------------+--------------------+
        5 |         (same as data passed)           |  4 FQFIL
          +--------------------+--------------------+
        7 |  number of blocks owned by account read |  6 FQPPN
          +--------------------+--------------------+
       11 |           PPN of account read           | 10 FQNAM1
          +--------------------+--------------------+
       13 |  password of account read (2 words in   | 12
          +         RAD50 format) or zeros          +
       15 |                                         | 14 FQEXT
          +--------------------+--------------------+
       17 |      LSB of CPU time (.1 seconds)       | 16 FQSIZ
          +--------------------+--------------------+
       21 |         connect time (minutes)          | 20 FQBUFL
          +--------------------+--------------------+
       23 |              LSB of KCTs                | 22
          +--------------------+--------------------+
       25 |          device time (minutes)          | 24 FQFLAG
          +--------------------+--------------------+
FQPROT 27 |  MSB of CPU time   |    MSB of KCTs     | 26 FQPFLG
          +--------------------+--------------------+
       31 |         (same as data passed)           | 30 FQDEV
          +--------------------+--------------------+
       33 |         (same as data passed)           | 32 FQDEVN
          +--------------------+--------------------+
       35 |   disk quota in blocks; 0 = unlimited   | 34
          +--------------------+--------------------+
       37 |            UFD cluster size             | 36
          +--------------------+--------------------+
```

FIRQB+FQPPN	If the byte at FIRQB+FQNAM1+2 in the data passed is 1, this word is zero.
FIRQB+FQNAM1+2	If the caller has GACNT or WACNT privilege and the account allows password look-up, the password is returned as 2 words of RAD50. Otherwise, zeros are returned.
FIRQB+FQSIZ	The LSB of the CPU time in tenths of a second. See the byte at FIRQB+FQPROT for the MSB.
FIRQB+FQMODE	The LSB of the Kilo-Core-Ticks (KCTs). See the byte at FIRQB+FQPFLG for the MSB.

Errors

BADDIR	The account does not have all the necessary directory structures.
BADFUO	Device specified is not a disk.
NOSUCH	The PPN specified does not exist on the disk, or the index specified is greater than the number of accounts on the disk.
PRVIOL	The caller does not have sufficient privilege to perform the indicated function.

UU.RTS (Add/Remove/Run-Time System or Resident Library, or Create Dynamic Region)

Privileges Required

Read access (by protection code, GREAD, WREAD, and/or SYSIO privilege) is required to install a run-time system or resident library. INSTALL is required to install, remove, load, or unload a run-time system or resident library, or create a dynamic region.

Data Passed - Add a Run-Time System

```
                                 FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                              Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |   UU.RTS (= -22)     |//////////////////////|  2
          +---------------------+---------------------+
        5 |//////////////////////|         flag         |  4 FQFIL
          +---------------------+---------------------+
        7 |   project number     |  programmer number   |  6 FQPPN
          +---------------------+---------------------+
       11 |              run-time system name           | 10 FQNAM1
          +                                             +
       13 |              (2 words of RAD50)             | 12
          +---------------------+---------------------+
       15 |          start load address or zero         | 14 FQEXT
          +---------------------+---------------------+
       17 | max. user job image size, K words (P.SIZE)| 16 FQSIZ
          +---------------------+---------------------+
       21 | min. user job image size, K words (P.MSIZ)| 20 FQBUFL
          +---------------------+---------------------+
       23 |      stay flag       | linked-list position| 22 FQMODE
          +---------------------+---------------------+
       25 |              flag word (P.FLAG)             | 24 FQFLAG
          +---------------------+---------------------+
FQPROT 27 |  dflt. exec. file type (P.DEXT), RAD50     | 26 FQPFLG
          +---------------------+---------------------+
       31 |        device name (disk) where stored      | 30 FQDEV
          +---------------------+---------------------+
       33 |  ≠0, unit no. real   |  device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFIL	All bits clear implies the values for P.SIZE, P.MSIZ, P.FLAG, and P.DEXT symbols are in the call. Bit 7 set means the system should obtain the values for these symbols from the .RTS file.
FIRQB+FQEXT	If zero, the monitor calculates the start load address each time the run-time system is loaded. Otherwise, the start load address is K times the value specified.
FIRQB+FQMODE+1	All bits clear means the memory occupied by this run-time system can be released when usage count goes to zero. Bit 7 set means the run-time system is permanently resident.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.RTS subfunction does not return any meaningful data.

Errors

BADCNT	The range of memory starting at the load address at FIRQB+FQEXT is not available. See the output of the SHOW MEMORY command to select an available range of memory.
FIEXST	A run-time system with the same name already exists.
NOBUFS	Adding a run-time system description block requires a small buffer and none is currently available.
NOROOM	If loaded at address specified, memory is fragmented and a swapping violation occurs. See the discussion of assigning and allocating memory in the *RSTS/E System Installation and Update Guide* for guidelines on how to avoid fragmenting memory.
NOSUCH	No file with the name given and a type of .RTS can be found in the account at FIRQB+FQPPN on the device specified.
PRVIOL	The file to be added has a bad format. For example, it is not contiguous or has illegal entries in the SIL index.

Data Passed - Remove a Run-Time System

```
                             FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
           +---------------------+---------------------+
         1 |///////////////////////////////////////////|  0
           +---------------------+---------------------+
FQFUN    3 |   UU.RTS (= -22)    |/////////////////////|  2
           +---------------------+---------------------+
         5 |/////////////////////| subfunction code =4 |  4 FQFIL
           +---------------------+---------------------+
         7 |///////////////////////////////////////////|  6
           +---------------------+---------------------+
        11 |           run-time system name            | 10 FQNAM1
           +                                           +
        13 |           (2 words of RAD50)              | 12
           +---------------------+---------------------+
        15 |///////////////////////////////////////////| 14
           +---------------------+---------------------+
        17 |///////////////////////////////////////////| 16
           +---------------------+---------------------+
        21 |///////////////////////////////////////////| 20
           +---------------------+---------------------+
        23 |///////////////////////////////////////////| 22
           +---------------------+---------------------+
        25 |///////////////////////////////////////////| 24
           +---------------------+---------------------+
        27 |///////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |///////////////////////////////////////////| 30
           +---------------------+---------------------+
        33 |///////////////////////////////////////////| 32
           +---------------------+---------------------+
        35 |///////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |///////////////////////////////////////////| 36
           +---------------------+---------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.RTS subfunction does not return any meaningful data.

Errors

INUSE	This run-time system is currently being loaded into memory or is resident and in use. It cannot be removed until the usage count is zero.
NOSUCH	The run-time system is not currently defined.

PRVIOL The run-time system is the primary run-time system or the system default keyboard monitor and cannot be removed with this call. Use the DEFAULT initialization option to change the primary run-time system before starting timesharing. The system default keyboard monitor is always DCL.

Data Passed - Unload a Run-Time System

```
                               FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                           Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////| 0
          +---------------------+---------------------+
FQFUN   3 |   UU.RTS (= -22)    |/////////////////////| 2
          +---------------------+---------------------+
        5 |/////////////////////| subfunction code =6 | 4 FQFIL
          +---------------------+---------------------+
        7 |///////////////////////////////////////////| 6
          +---------------------+---------------------+
       11 |              run-time system name         | 10 FQNAM1
          +                                           +
       13 |              (2 words of RAD50)           | 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.RTS subfunction does not return any meaningful data.

Errors

INUSE	The run-time system is currently being loaded or is resident and in use. It cannot be unloaded until usage count is zero.
NOSUCH	The run-time system is not currently defined.

Data Passed -- Add a Resident Library

```
                               FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                          Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////| 0
          +---------------------+---------------------+
FQFUN   3 |   UU.RTS (= -22)    |/////////////////////| 2
          +---------------------+---------------------+
        5 |/////////////////////| subfunction code=20 | 4 FQFIL
          +---------------------+---------------------+
        7 |   project number    | programmer number   | 6 FQPPN
          +---------------------+---------------------+
       11 |           run-time system name            | 10 FQNAM1
          +                                           +
       13 |             (2 words of RAD50)            | 12
          +---------------------+---------------------+
       15 |        start load address or zero         | 14 FQEXT
          +---------------------+---------------------+
       17 |/////////////////////////////////////////// | 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |     stay flag       |/////////////////////| 22 FQMODE
          +---------------------+---------------------+
       25 |                flag word                  | 24 FQFLAG
          +---------------------+---------------------+
FQPROT 27 |   protection code   | ≠0, prot.code real  | 26 FQPFLG
          +---------------------+---------------------+
       31 |      device name (disk) where stored      | 30 FQDEV
          +---------------------+---------------------+
       33 | ≠0, unit no. real   | device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQEXT	If zero and the library requires a fixed address, the monitor finds the address progressing from high to low memory. Otherwise, the monitor calculates the start load address when the library is actually needed.

If -1, the monitor finds the first free space large enough to hold the library, starting from the top of memory.

Otherwise, the start load address is K times the value specified. The value can be in the range 1 to n-1; 1 means the first available 1K word section and n is the total number of 1K word sections available on the system.

FIRQB+FQMODE+1	All bits clear means the resident library can be removed from memory when usage count goes to zero.
	Bit 7 set means the resident library is permanently resident.
FIRQB+FQFLAG	Bit settings indicate resident library characteristics:

Bit	Setting	Meaning
9	Set	The resident library is to be accessed by only one user. That is, the library is not to be shared.
	Clear	The resident library can be shared by more than one user.
10	Set	Read/write access is allowed.
	Clear	Read-only access.
12	Set	Errors occurring within code in the resident library are not to be recorded in the system error log.
	Clear	Errors may be recorded in the system error log.
13	Set	The resident library is immediately removed from memory when its usage count goes to zero.
	Clear	When the usage count for this library goes to zero, the monitor will remove it from memory when the space it takes is needed for something else.

All other bits are reserved and should be zero.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.RTS subfunction does not return any meaningful data.

Errors

BADCNT	You did not specify a load address, or the one specified is not available. See the output of the SHOW MEMORY command to find an available range of memory.
FIEXST	A resident library with the specified name already exists.
NOBUFS	Adding a resident library description block requires a small buffer and none is currently available.
NOROOM	If loaded at address specified, memory is fragmented and a swapping violation occurs. See the discussion of assigning and allocating memory in the *RSTS/E System Installation and Update Guide* for guidelines on how to avoid fragmenting memory.
NOSUCH	No file with the name given and a type of .LIB can be found for the given account and device.
PRVIOL	The file to be added has a bad format. For example, it is not contiguous or has illegal entries in the SIL index.

Data Passed - Remove a Resident Library

```
                              FIRQB
Mne-   Octal                                             Octal   Mne-
monic Offset                                             Offset  monic
            +---------------------+---------------------+
          1 |/////////////////////////////////////////////| 0
            +---------------------+---------------------+
FQFUN     3 |   UU.RTS (= -22)    |/////////////////////| 2
            +---------------------+---------------------+
          5 |/////////////////////| subfunction code=24 | 4 FQFIL
            +---------------------+---------------------+
          7 |/////////////////////////////////////////////| 6
            +---------------------+---------------------+
         11 |             resident library name          | 10 FQNAM1
            +                                           +
         13 |              (2 words of RAD50)            | 12
            +---------------------+---------------------+
         15 |/////////////////////////////////////////////| 14
            +---------------------+---------------------+
         17 |/////////////////////////////////////////////| 16
            +---------------------+---------------------+
         21 |/////////////////////////////////////////////| 20
            +---------------------+---------------------+
         23 |/////////////////////////////////////////////| 22
            +---------------------+---------------------+
         25 |/////////////////////////////////////////////| 24
            +---------------------+---------------------+
         27 |/////////////////////////////////////////////| 26
            +---------------------+---------------------+
         31 |/////////////////////////////////////////////| 30
            +---------------------+---------------------+
         33 |/////////////////////////////////////////////| 32
            +---------------------+---------------------+
         35 |/////////////////////////////////////////////| 34
            +---------------------+---------------------+
         37 |/////////////////////////////////////////////| 36
            +---------------------+---------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.RTS subfunction does not return any meaningful data.

Errors

INUSE	You tried to remove a library that is being loaded into memory or is in use. A resident library cannot be removed while a job is attached to it.
NOSUCH	You specified a resident library that is not currently defined.

Data Passed -- Unload a Resident Library

```
                                FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |   UU.RTS (= -22)    |///////////////////////|  2
          +---------------------+---------------------+
        5 |/////////////////////| subfunction code=26 |  4 FQFIL
          +---------------------+---------------------+
        7 |/////////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |            resident library name            | 10 FQNAM1
          +                                             +
       13 |             (2 words of RAD50)              | 12
          +---------------------+---------------------+
       15 |/////////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |/////////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |/////////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |/////////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |/////////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.RTS subfunction does not return any meaningful data.

Errors

INUSE	You tried to unload a resident library that is now being loaded or is in use. A library cannot be unloaded while a job is still attached to it.
NOSUCH	You specified a resident library that is not currently defined.

Data Passed - Create Dynamic Region

```
                                  FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
            +---------------------+---------------------+
         1  |///////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN    3  |   UU.RTS (= -22)    |/////////////////////|  2
            +---------------------+---------------------+
         5  |/////////////////////| subfunction code=30 |  4 FQFIL
            +---------------------+---------------------+
         7  |///////////////////////////////////////////|  6
            +---------------------+---------------------+
        11  |            dynamic region name            | 10 FQNAM1
            +                                           +
        13  |             (2 words of RAD50)            | 12
            +---------------------+---------------------+
        15  |       start dynamic region address        | 14 FQEXT
            +---------------------+---------------------+
        17  |/////////////////////| dynamic region size | 16 FQSIZ
            +---------------------+---------------------+
        21  |///////////////////////////////////////////| 20
            +---------------------+---------------------+
        23  |              creation flags               | 22 FQMODE
            +---------------------+---------------------+
        25  |               region flags                | 24 FQFLAG
            +---------------------+---------------------+
FQPROT  27  |   protection code   | protection code flag| 26 FQPFLG
            +---------------------+---------------------+
        31  |///////////////////////////////////////////| 30
            +---------------------+---------------------+
        33  |///////////////////////////////////////////| 32
            +---------------------+---------------------+
        35  |///////////////////////////////////////////| 34
            +---------------------+---------------------+
        37  |///////////////////////////////////////////| 36
            +---------------------+---------------------+
```

FIRQB+FQNAM1	If zero, the monitor creates an unnamed dynamic region.
FIRQB+FQEXT	The starting address of the dynamic region is K times the value of this word. If zero, the monitor computes the starting address for the dynamic region.
FIRQB+FQSIZ	The size of the region in K words. The value can be in the range 1 to 177.
FIRQB+FQMODE	Creation flags. The following bits are currently defined: Bit 7, when set, means attach this job to the named region. Bit 15, when set, means do not delete this region even if all users detach from it. All other bits are reserved and should be zero.
FIRQB+FQFLAG	Region flags. The following bit is currently defined: Bit 9, when set, means this region cannot be shared. All other bits are reserved and should be zero.
FIRQB+FQPFLG	If nonzero, then the value in FIRQB+FQPROT is real.
FIRQB+FQPROT	The protection code of the region.

Data Returned - Create Dynamic Region

```
                                FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
              +---------------------+---------------------+
          1   |///////////////////////////////////////////|   0
              +---------------------+---------------------+
          3   |///////////////////////////////////////////|   2
              +---------------------+---------------------+
          5   |///////////////////////////////////////////|   4
              +---------------------+---------------------+
          7   |            dynamic region ID              |   6  FQPPN
              +---------------------+---------------------+
         11   |///////////////////////////////////////////|  10
              +---------------------+---------------------+
         13   |///////////////////////////////////////////|  12
              +---------------------+---------------------+
         15   |///////////////////////////////////////////|  14
              +---------------------+---------------------+
         17   |/////////////////////| dynamic region size |  16  FQSIZ
              +---------------------+---------------------+
         21   |///////////////////////////////////////////|  20
              +---------------------+---------------------+
         23   |///////////////////////////////////////////|  22
              +---------------------+---------------------+
         25   |///////////////////////////////////////////|  24
              +---------------------+---------------------+
         27   |///////////////////////////////////////////|  26
              +---------------------+---------------------+
         31   |///////////////////////////////////////////|  30
              +---------------------+---------------------+
         33   |///////////////////////////////////////////|  32
              +---------------------+---------------------+
         35   |///////////////////////////////////////////|  34
              +---------------------+---------------------+
         37   |///////////////////////////////////////////|  36
              +---------------------+---------------------+
```

FIRQB+FQPPN	The ID of the created dynamic region. The .PLAS directives use this ID.
FIRQB+FQSIZ	The size of the created dynamic region in K words. The value can be in the range 1 to 177.

Errors

BADCNT	You tried to create a region with an illegal size specification or the load address was not valid.
FIEXST	You specified the name of a dynamic region or resident library that already exists.
NOBUFS	A small buffer was not available for the region description block. Also, if attachment was specified, a small buffer was not available for the window descriptor block.
NOROOM	If the dynamic region was loaded at the specified address, memory is fragmented and a swapping violation occurs. If the monitor was choosing the address, there is not enough free memory to create the region.

UU.SLN (System Logical Names)

Privileges Required

INSTAL is required to add, remove, or change system logical names.

Data Passed - Add New Name

```
                                FIRQB
Mne-  Octal                                           Octal   Mne-
monic Offset                                           Offset  monic
           +-------------------+-------------------+
        1  |///////////////////////////////////////|  0
           +-------------------+-------------------+
FQFUN   3  |    UU.SLN (= 25)  |///////////////////|  2
           +-------------------+-------------------+
        5  |///////////////////| subfunction code= 1 |  4 FQFIL
           +-------------------+-------------------+
        7  | PPN for this name (if 0, none associated) |  6 FQPPN
           +-------------------+-------------------+
       11  |      logical name in RAD50 format      | 10 FQNAM1
           +              (3 words)                  +
       13  |                                         | 12
           +                                         +
       15  |                                         | 14
           +-------------------+-------------------+
       17  |///////////////////////////////////////| 16
           +-------------------+-------------------+
       21  |///////////////////////////////////////| 20
           +-------------------+-------------------+
       23  |///////////////////////////////////////| 22
           +-------------------+-------------------+
       25  |///////////////////////////////////////| 24
           +-------------------+-------------------+
       27  |///////////////////////////////////////| 26
           +-------------------+-------------------+
       31  |     device name (2 ASCII characters)    | 30 FQDEV
           +-------------------+-------------------+
       33  |      must ≠0      | device unit number | 32 FQDEVN
           +-------------------+-------------------+
       35  |///////////////////////////////////////| 34
           +-------------------+-------------------+
       37  |///////////////////////////////////////| 36
           +-------------------+-------------------+
```

FIRQB+FQFIL	This code adds a new entry to the list of system logical names.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.SLN subfunction does not return any meaningful data.

Errors

BADNAM	No logical name was given or the name contains invalid characters.
INUSE	The logical name given duplicates one already in use.
NOBUFS	An attempt was made to extend the logical name table and there was no room in XBUF.
NODEVC	The device specification is illegal or is not on the system.

Data Passed - Remove Logical Name

```
                                   FIRQB
Mne-  Octal                                                   Octal   Mne-
monic Offset                                                  Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////| 0
          +---------------------+---------------------+
FQFUN   3 |    UU.SLN (= 25)    |/////////////////////| 2
          +---------------------+---------------------+
        5 |/////////////////////| subfunction code= 0 | 4 FQFIL
          +---------------------+---------------------+
        7 |/////////////////////////////////////////////| 6
          +---------------------+---------------------+
       11 |        logical name in RAD50 format        | 10 FQNAM1
          +                (3 words)                   +
       13 |                                            | 12
          +                                            +
       15 |                                            | 14
          +---------------------+---------------------+
       17 |/////////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |/////////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |/////////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |/////////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.SLN subfunction does not return any meaningful data.

Errors

BADNAM	No logical name was given or the name contains invalid characters.
NOSUCH	The name given is not currently defined as a logical name.

Data Passed - Change Disk Logical Name

```
                                  FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
           +--------------------+--------------------+
         1 |/////////////////////////////////////////|  0
           +--------------------+--------------------+
FQFUN    3 |    UU.SLN (= 25)   |////////////////////|  2
           +--------------------+--------------------+
         5 |////////////////////| subfunction code=-1 |  4 FQFIL
           +--------------------+--------------------+
         7 |/////////////////////////////////////////|  6
           +--------------------+--------------------+
        11 |       logical name in RAD50 format      | 10 FQNAM1
           +                (3 words)                +
        13 |                                         | 12
           +                                         +
        15 |                                         | 14
           +--------------------+--------------------+
        17 |/////////////////////////////////////////| 16
           +--------------------+--------------------+
        21 |/////////////////////////////////////////| 20
           +--------------------+--------------------+
        23 |/////////////////////////////////////////| 22
           +--------------------+--------------------+
        25 |/////////////////////////////////////////| 24
           +--------------------+--------------------+
        27 |/////////////////////////////////////////| 26
           +--------------------+--------------------+
        31 |    device name (2 ASCII characters)     | 30 FQDEV
           +--------------------+--------------------+
        33 |      must ≠0       | device unit number | 32 FQDEVN
           +--------------------+--------------------+
        35 |/////////////////////////////////////////| 34
           +--------------------+--------------------+
        37 |/////////////////////////////////////////| 36
           +--------------------+--------------------+
```

FIRQB+FQFIL	This code changes the logical name associated with a disk in the list of system logical names.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.SLN subfunction does not return any meaningful data.

Errors

BADNAM	No logical name was given, or the name contains invalid characters.
INUSE	The logical name given duplicates one already in use.
NOSUCH	The disk specified is not on the system.
NODEVC	The device specified is illegally formatted or is not a disk.

Data Passed - List Logical Name

```
                               FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                              Offset  monic
            +---------------------+---------------------+
          1 |///////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN     3 |    UU.SLN (= 25)    |/////////////////////|  2
            +---------------------+---------------------+
          5 |/////////////////////| subfunction code= 2 |  4 FQFIL
            +---------------------+---------------------+
          7 |                   index                   |  6 FQPPN
            +---------------------+---------------------+
         11 |///////////////////////////////////////////| 10
            +---------------------+---------------------+
         13 |///////////////////////////////////////////| 12
            +---------------------+---------------------+
         15 |///////////////////////////////////////////| 14
            +---------------------+---------------------+
         17 |///////////////////////////////////////////| 16
            +---------------------+---------------------+
         21 |///////////////////////////////////////////| 20
            +---------------------+---------------------+
         23 |///////////////////////////////////////////| 22
            +---------------------+---------------------+
         25 |///////////////////////////////////////////| 24
            +---------------------+---------------------+
         27 |///////////////////////////////////////////| 26
            +---------------------+---------------------+
         31 |///////////////////////////////////////////| 30
            +---------------------+---------------------+
         33 |///////////////////////////////////////////| 32
            +---------------------+---------------------+
         35 |///////////////////////////////////////////| 34
            +---------------------+---------------------+
         37 |///////////////////////////////////////////| 36
            +---------------------+---------------------+
```

FIRQB+FQPPN	Index. An integer representing the relative position of the specified logical name in the list of system logical names. You can list all logicals by repeated calls with an index value starting at zero and incremented by one each time.

Data Returned

```
                              FIRQB
Mne-  Octal                                         Octal   Mne-
monic Offset                                        Offset  monic
         +---------------------+---------------------+
       1 |/////////////////////////////////////////////|  0
         +---------------------+---------------------+
       3 |/////////////////////////////////////////////|  2
         +---------------------+---------------------+
       5 |/////////////////////////////////////////////|  4
         +---------------------+---------------------+
       7 |   project number    |  programmer number  |  6 FQPPN
         +---------------------+---------------------+
      11 |        logical name in RAD50 format        | 10 FQNAM1
         +                                           +
      13 |                 (3 words)                 | 12
         +                                           +
      15 |                                           | 14
         +---------------------+---------------------+
      17 |/////////////////////////////////////////////| 16
         +---------------------+---------------------+
      21 |/////////////////////////////////////////////| 20
         +---------------------+---------------------+
      23 |/////////////////////////////////////////////| 22
         +---------------------+---------------------+
      25 |/////////////////////////////////////////////| 24
         +---------------------+---------------------+
      27 |/////////////////////////////////////////////| 26
         +---------------------+---------------------+
      31 |    device name (2 ASCII characters)        | 30 FQDEV
         +---------------------+---------------------+
      33 |       must ≠0        |  device unit number | 32 FQDEVN
         +---------------------+---------------------+
      35 |//////////////////////|      mount flag      | 34 FQCLUS
         +---------------------+---------------------+
      37 |/////////////////////////////////////////////| 36
         +---------------------+---------------------+
```

FIRQB+FQCLUS	The value of this byte is 377 if the indexed logical is a disk and the disk is not mounted.

Errors

NOSUCH	Index out of range.

UU.SPL (Spooling)

RSTS/E currently has two spooling packages: the Print/Batch Services (PBS) package and the OPSER-based package. When you execute the UU.SPL directive, the system first looks at the spooled device name field (FIRQB+FQSIZ). The monitor, by default, routes the request to PBS if it is running and the field is null or LP, or it is BA and the file type is .COM. Otherwise, the monitor routes the request to the OPSER-based package.

Note that by setting a bit in FQFLAG, you can force the request to be sent to whichever spooling package you want.

For requests routed to PBS, any request parameter that cannot be passed by UU.SPL is assigned a default value.

Privileges Required

File read access (by protection code, or GREAD or WREAD privilege) is required to queue a file. File write access (by protection code, or GWRITE or WWRITE privilege) is also required to queue a file with the delete option flag. DEVICE privilege is required if the file is on a restricted device.

Data Passed

```
                                FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
            +--------------------+---------------------+
          1 |///////////////////////////////////////////|  0
            +--------------------+---------------------+
FQFUN     3 |    UU.SPL (= -34)  |//////////////////////|  2
            +--------------------+---------------------+
          5 |///////////////////////////////////////////|  4
            +--------------------+---------------------+
          7 |    project number  | programmer number   |  6 FQPPN
            +--------------------+---------------------+
         11 |    wildcard file name in RAD50 format    | 10 FQNAM1
            +                 (2 words)                +
         13 |                                          | 12
            +--------------------+---------------------+
         15 |wildcard file type in RAD50 format (1 word)| 14 FQEXT
            +--------------------+---------------------+
         17 |device name to queue to (ASCII); 0 = "LP0" | 16 FQSIZ
            +--------------------+---------------------+
         21 |device unit real flag| device unit number | 20 FQBUFL
            +--------------------+---------------------+
         23 |///////////////////////////////////////////| 22 FQMODE
            +--------------------+---------------------+
         25 |                 flag word                | 24 FQFLAG
            +--------------------+---------------------+
         27 |///////////////////////////////////////////| 26
            +--------------------+---------------------+
         31 |      device name where file is (disk)    | 30 FQDEV
            +--------------------+---------------------+
         33 |≠0, unit number real | device unit number | 32 FQDEVN
            +--------------------+---------------------+
         35 |///////////////////////////////////////////| 34
            +--------------------+---------------------+
         37 |///////////////////////////////////////////| 36
            +--------------------+---------------------+
```

FIRQB+FQSIZ	The requested device name. Permitted values are LP, BA, and zero which translates to "LP0:."
FIRQB+FQBUFL	The device unit real flag. If zero, it means use the default print or batch queue. If nonzero, it means use the queue with the same name as the device name.

FIRQB+FQFLAG You can set the following bits for PBS:

Bit	Meaning When Set
2	Delete after spooling; same as DCL command PRINT/DELETE.
5	No header; same as DCL command PRINT/NOFLAG_PAGES. (Ignore for batch requests.)
13	Route request to OPSER spooler
14	Route request to PBS

If both bits 13 and 14 are set, then the request is routed to PBS. If both bits are clear, then the following rules apply:

- o If the spooler device name is "LP" or null, or the device name is "BA" and the file type is .COM, then the request is routed to PBS if it is running; otherwise, it is routed to OPSER.
- o If the spooler device name is not "LP" or null, then the request is routed to OPSER.

You can set the following bits for OPSER:

Bit	Meaning When Set
0	The file is spooled with FORTRAN carriage control; same effect as QUE/TYP:FTN.
1	Restart file; same effect as QUE/RE.
2	Delete file after spooling; same effect as QUE/DE.
3	Binary file; same effect as QUE/BI.
4	The end; same effect as QUE/END.
5	No header; same effect as QUE/NH.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.SPL subfunction does not return any meaningful data.

Errors

DEVNFS	The device specified at FIRQB+FQDEV of the call is not a file-structured device.
HNGDEV	This error is caused by a hardware condition. For example, the specified disk could not be accessed.
NOBUFS	System buffers are not currently available to store this message. This may be a transient condition; retry the operation.
NODEVC	An attempt was made to queue a file to a device that had a unit number greater than seven, or the file to be queued is on an invalid device.
NOROOM	The number of messages pending for the queue manager program is at its declared maximum. This may be a transient condition; retry the operation.
NOSUCH	The account specified at FIRQB+FQPPN does not exist on the device specified, the file name or type specified at FIRQB+FQNAM1 cannot be found.
NOTMNT	The specified disk device is not mounted.
PAKLCK	The specified disk is restricted, and the caller does not have DEVICE privilege.
PRVIOL	An attempt was made to queue a file to which the user did not have read access or to queue an executable file.

UU.STL (Stall/Unstall System)

Privileges Required

HWCTL is required to stall or unstall a system.

Data Passed

```
                              FIRQB
Mne-  Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
        +--------------------+--------------------+
      1 |////////////////////////////////////////|  0
        +--------------------+--------------------+
FQFUN 3 |    UU.STL (= 35)   |////////////////////|  2
        +--------------------+--------------------+
      5 |////////////////////|        0 or 1      |  4 FQFIL
        +--------------------+--------------------+
      7 |////////////////////////////////////////|  6
        +--------------------+--------------------+
     11 |////////////////////////////////////////| 10
        +--------------------+--------------------+
     13 |////////////////////////////////////////| 12
        +--------------------+--------------------+
     15 |////////////////////////////////////////| 14
        +--------------------+--------------------+
     17 |////////////////////////////////////////| 16
        +--------------------+--------------------+
     21 |////////////////////////////////////////| 20
        +--------------------+--------------------+
     23 |////////////////////////////////////////| 22
        +--------------------+--------------------+
     25 |////////////////////////////////////////| 24
        +--------------------+--------------------+
     27 |////////////////////////////////////////| 26
        +--------------------+--------------------+
     31 |////////////////////////////////////////| 30
        +--------------------+--------------------+
     33 |////////////////////////////////////////| 32
        +--------------------+--------------------+
     35 |////////////////////////////////////////| 34
        +--------------------+--------------------+
     37 |////////////////////////////////////////| 36
        +--------------------+--------------------+
```

FIRQB+FQFIL	Zero means return the system to normal (unstalled) state. One means Stall; suspend all currently active jobs except for the calling job.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.STL subfunction does not return any meaningful data.

Errors

BADFUO	You specified unstall, but the system is not stalled.
INUSE	You tried to stall the system, but it is already stalled.

UU.SWP (Add, Remove, and List System Files)

Privileges Required

INSTAL is required to add or remove system files. File read access (by protection code, or GREAD or WREAD privilege) and file write access (by protection code, or GWRITE, WWRITE, and/or SYSIO privilege) is required to add system files. WRTNFS is required to add non-file-structured swap space.

Data Passed -- Add System File

```
                                  FIRQB
Mne-  Octal                                               Octal   Mne-
monic Offset                                               Offset  monic
          +--------------------+--------------------+
        1 |/////////////////////////////////////////| 0
          +--------------------+--------------------+
FQFUN   3 |    UU.SWP (= 27)   |////////////////////| 2
          +--------------------+--------------------+
FQSIZM  5 |     = 1 for add    |     file to add    | 4 FQFIL
          +--------------------+--------------------+
        7 |/////////////////////////////////////////| 6
          +--------------------+--------------------+
       11 |    for [0,1] file with type of .SYS,    | 10 FQNAM1
          +    file name as 2 words RAD50. If both  +
       13 |   words = 0, non-file-structured disk.  | 12
          +--------------------+--------------------+
       15 |/////////////////////////////////////////| 14
          +--------------------+--------------------+
       17 |/////////////////////////////////////////| 16
          +--------------------+--------------------+
       21 |/////////////////////////////////////////| 20
          +--------------------+--------------------+
       23 |/////////////////////////////////////////| 22
          +--------------------+--------------------+
       25 |/////////////////////////////////////////| 24
          +--------------------+--------------------+
       27 |/////////////////////////////////////////| 26
          +--------------------+--------------------+
       31 |  device name (disk) (2 ASCII characters) | 30 FQDEV
          +--------------------+--------------------+
       33 |≠0, unit number real | device unit number | 32 FQDEVN
          +--------------------+--------------------+
       35 |/////////////////////////////////////////| 34
          +--------------------+--------------------+
       37 |/////////////////////////////////////////| 36
          +--------------------+--------------------+
```

FIRQB+FQFIL The possible bit settings are:

Bit	Meaning
0	Swap file 0
1	Swap file 1
2	Illegal (generates an error; slot 2 is always added)
3	Swap file 3
4	Overlay file
5	Error message file
6	DECnet/E system file

Data Returned

Except for a possible error in byte 0 of the FIRQB, the add option of the UU.SWP subfunction does not return any meaningful data.

Errors

BADFUO	The number specified at FIRQB+FQFIL is either two or is greater than six. The swap file 2 must exist on the system disk and cannot be added during timesharing. System files to be added are defined only by the values 0, 1, 3, 4, 5, and 6.
BADNAM	No file name is specified when an overlay, an error message, or a DECnet/E system file is being added, or the name specified contains nonalphanumeric characters.
DEVNFS	The device specified is not a disk device.
FIEXST	The system file being added is already installed on the system.
INUSE	A swap file is being added to a non-file-structured disk, but the disk is currently mounted (that is, it is being used as a file-structured device).
NODEVC	The device specified is a disk but is not on this system.

NOROOM	If an overlay or error file is being added, this error indicates that the file is not long enough. (The overlay file should be at least 128. blocks and the error file at least 16. blocks.) If a swap file is being added to a file-structured device, this error means that the file is not long enough to store even one job.
NOSUCH	A system file is being added to a file-structured disk, but the file with the name specified and with a .SYS file type does not exist in account [0,1].
NOTAVL	A swap file is being added to a non-file-structured disk, but either the disk unit or its controller has been disabled. The system manager must use an initialization option to enable the unit or its controller.
NOTMNT	A system file is being added to a file-structured disk but that disk is not currently mounted. Use the MOUNT command to logically mount the disk before the file is added.
PRVIOL	A system file is being added to a file-structured disk. Either the unit is logically write locked, or the file specified is bad; that is, it is not contiguous or is currently open.

Data Passed - Remove System File

```
                              FIRQB
Mne-  Octal                                              Octal   Mne-
monic Offset                                              Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.SWP (= 27)    |/////////////////////|  2
          +---------------------+---------------------+
FQSIZM  5 |   = 0 for remove    |   file to remove    |  4 FQFIL
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFIL The possible bit settings are:

Bit	Meaning
0	Swap file 0
1	Swap file 1
2	Illegal (generates an error; slot 2 is never removed)
3	Swap file 3
4	Overlay file
5	Error message file
6	DECnet/E system file

Data Returned

Except for a possible error in byte 0 of the FIRQB, the remove option of the UU.SWP subfunction does not return any meaningful data.

Errors

BADFUO	The number specified at FIRQB+FQFIL is either two or greater than six. The swap file 2 must exist on the system disk and cannot be removed during time sharing. System files to be removed are defined only by the values 0, 1, 3, 4, 5 and 6.
INUSE	The swap file to be removed can be properly removed but currently contains one or more swapped out jobs. The system locks the file and begins swapping jobs to other files. Retry the call at a later time when the swapped out jobs are no longer in this file.
PRVIOL	A swap file is to be removed but its removal decreases the swap file space below the limit required to store the maximum number of jobs on the system. To remove the swap file, decrease the number of logins currently allowed (by either the SET LOGINS command or SYS call), wait until the number of logged in jobs falls to the maximum, and try the removal operation again. This error also occurs if you attempt to remove the DECnet/E system file when DECnet/E is still on.

Data Passed - List System File

```
                                 FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
            +---------------------+---------------------+
          1 |///////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN     3 |    UU.SWP (= 27)    |/////////////////////|  2
            +---------------------+---------------------+
FQSIZM    5 |     = -1 to list    |    file to list     |  4 FQFIL
            +---------------------+---------------------+
          7 |///////////////////////////////////////////|  6
            +---------------------+---------------------+
         11 |///////////////////////////////////////////| 10
            +---------------------+---------------------+
         13 |///////////////////////////////////////////| 12
            +---------------------+---------------------+
         15 |///////////////////////////////////////////| 14
            +---------------------+---------------------+
         17 |///////////////////////////////////////////| 16
            +---------------------+---------------------+
         21 |///////////////////////////////////////////| 20
            +---------------------+---------------------+
         23 |///////////////////////////////////////////| 22
            +---------------------+---------------------+
         25 |///////////////////////////////////////////| 24
            +---------------------+---------------------+
         27 |///////////////////////////////////////////| 26
            +---------------------+---------------------+
         31 |///////////////////////////////////////////| 30
            +---------------------+---------------------+
         33 |///////////////////////////////////////////| 32
            +---------------------+---------------------+
         35 |///////////////////////////////////////////| 34
            +---------------------+---------------------+
         37 |///////////////////////////////////////////| 36
            +---------------------+---------------------+
```

FIRQB+FQFIL The possible bit settings are:

Bit	Meaning
0	Swap file 0
1	Swap file 1
2	Swap file 2
3	Swap file 3
4	Overlay file
5	Error message file
6	DECnet/E system file

Data Returned - List System File

```
                                FIRQB
Mne-  Octal                                          Octal  Mne-
monic Offset                                         Offset  monic
         +--------------------+--------------------+
       1 |/////////////////////////////////////////| 0
         +--------------------+--------------------+
       3 |/////////////////////////////////////////| 2
         +--------------------+--------------------+
       5 |/////////////////////////////////////////| 4
         +--------------------+--------------------+
       7 |   project number   | programmer number  | 6 FQPPN
         +--------------------+--------------------+
      11 |        file name in RAD50 format        | 10 FQNAM1
         +                (2 words)                +
      13 |                                         | 12
         +--------------------+--------------------+
      15 |   file type in RAD50 format (1 word)    | 14 FQEXT
         +--------------------+--------------------+
      17 |/////////////////////////////////////////| 16
         +--------------------+--------------------+
      21 |/////////////////////////////////////////| 20
         +--------------------+--------------------+
      23 |/////////////////////////////////////////| 22
         +--------------------+--------------------+
      25 |/////////////////////////////////////////| 24
         +--------------------+--------------------+
      27 |/////////////////////////////////////////| 26
         +--------------------+--------------------+
      31 |     device name (2 ASCII characters)    | 30 FQDEV
         +--------------------+--------------------+
      33 |≠0, unit number real | device unit number | 32 FQDEVN
         +--------------------+--------------------+
      35 |/////////////////////////////////////////| 34
         +--------------------+--------------------+
      37 |/////////////////////////////////////////| 36
         +--------------------+--------------------+
```

Errors

BADFUO	The number specified at FIRQB+FQFIL is less than zero or greater than six.
NOSUCH	The number specified at FIRQB+FQFIL refers to a file that is not currently installed.

UU.SYS (Return Job Status Information)

Privileges Required

JOBCTL is required to read the status of another job, except a job running on a pseudo keyboard controlled by the caller. TUNE is required to obtain priority or run burst status.

Data Passed

```
                                FIRQB
Mne-   Octal                                                Octal   Mne-
monic  Offset                                               Offset  monic
             +---------------------+---------------------+
          1  |///////////////////////////////////////////|  0
             +---------------------+---------------------+
FQFUN     3  |    UU.SYS (= 32)    |/////////////////////|  2
             +---------------------+---------------------+
FQSIZM    5  | subcode= 0, 1, or 2 |   job number or 0   |  4 FQFIL
             +---------------------+---------------------+
          7  |///////////////////////////////////////////|  6
             +---------------------+---------------------+
         11  |///////////////////////////////////////////| 10
             +---------------------+---------------------+
         13  |///////////////////////////////////////////| 12
             +---------------------+---------------------+
         15  |///////////////////////////////////////////| 14
             +---------------------+---------------------+
         17  |///////////////////////////////////////////| 16
             +---------------------+---------------------+
         21  |///////////////////////////////////////////| 20
             +---------------------+---------------------+
         23  |///////////////////////////////////////////| 22
             +---------------------+---------------------+
         25  |///////////////////////////////////////////| 24
             +---------------------+---------------------+
         27  |///////////////////////////////////////////| 26
             +---------------------+---------------------+
         31  |///////////////////////////////////////////| 30
             +---------------------+---------------------+
         33  |///////////////////////////////////////////| 32
             +---------------------+---------------------+
         35  |///////////////////////////////////////////| 34
             +---------------------+---------------------+
         37  |///////////////////////////////////////////| 36
             +---------------------+---------------------+
```

Data Returned (for subcode = 0)

```
                              FIRQB
Mne-   Octal                                         Octal   Mne-
monic Offset                                         Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////| 0
          +---------------------+---------------------+
        3 |/////////////////////| calling job no. * 2 | 2 FQJOB
          +---------------------+---------------------+
FQSIZM  5 | job's console KB no.|  status job no. * 2 | 4 FQFIL
          +---------------------+---------------------+
        7 |  swap slot location | (ctrlg job no.*2)+1 | 6 FQPPN
          +---------------------+---------------------+
       11 |        LSB of CPU time, in .1 seconds     | 10 FQNAM1
          +---------------------+---------------------+
       13 |  job's current connect time, in minutes   | 12
          +---------------------+---------------------+
       15 |                LSB of KCTs                | 14 FQEXT
          +---------------------+---------------------+
       17 |   job's accumulated device time, minutes  | 16 FQSIZ
          +---------------------+---------------------+
       21 |   MSB of CPU time   |     MSB of KCTs     | 20 FQBUFL
          +---------------------+---------------------+
       23 |          job name in RAD50 format         | 22 FQMODE
          +                 (2 words)                 +
       25 |                                           | 24
          +---------------------+---------------------+
FQPROT 27 |    project number   |  programmer number  | 26 FQPFLG
          +---------------------+---------------------+
       31 |    job keyboard monitor in RAD50 format   | 30 FQDEV
          +                 (2 words)                 +
       33 |                                           | 32
          +---------------------+---------------------+
       35 |  current run-time system in RAD50 format  | 34 FQCLUS
          +                 (2 words)                 +
       37 |                                           | 36
          +---------------------+---------------------+
```

FIRQB+FQSIZM	If this value is negative, the job is detached, and the value is the one's complement of the keyboard number to which the job was previously attached.
FIRQB+FQPPN	Returned only if job is attached to a pseudo keyboard.

Data Returned (for subcode = 1)

```
                                  FIRQB
Mne-   Octal                                                  Octal   Mne-
monic Offset                                                  Offset  monic
              +---------------------+---------------------+
           1  |///////////////////////////////////////////|   0
              +---------------------+---------------------+
           3  |/////////////////////| calling job no. * 2 |   2 FQJOB
              +---------------------+---------------------+
FQSIZM     5  | job's console KB no.| status job no. * 2  |   4 FQFIL
              +---------------------+---------------------+
           7  |             current flag word             |   6 FQPPN
              +---------------------+---------------------+
          11  | curr. info. posting |    current IOSTS    |  10 FQNAM1
              +---------------------+---------------------+
          13  |            current JBSTAT word            |  12
              +---------------------+---------------------+
          15  |            current JBWAIT word            |  14 FQEXT
              +---------------------+---------------------+
          17  | mem. ctrl sub-block |  size, in K words   |  16 FQSIZ
              +---------------------+---------------------+
          21  |current physical addr., 32-word increments |  20 FQBUFL
              +---------------------+---------------------+
          23  | run burst,in .1 sec |       priority      |  22 FQMODE
              +---------------------+---------------------+
          25  |  value at offset 6  |  max. memory size   |  24 FQFLAG
              +---------------------+---------------------+
          27  |        (depends on JBWAIT and JBSTAT)     |  26 FQPFLG
              +---------------------+---------------------+
          31  |            read or write state            |  30 FQDEV
              +---------------------+---------------------+
          33  |         pointer to Job Data Block         |  32 FQDEVN
              +---------------------+---------------------+
          35  |     pointer to second Job Data Block      |  34 FQCLUS
              +---------------------+---------------------+
          37  |    pointer to first Receiver ID Block     |  36 FQNENT
              +---------------------+---------------------+
```

FIRQB+FQSIZM	If this value is negative, the job is detached, and the value is the one's complement of the keyboard number to which the job was previously attached.
FIRQB+FQMODE	Returned only if the caller has TUNE privilege.

Data Returned (for subcode = 2)

```
                                 FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
              +--------------------+--------------------+
            1 |/////////////////////////////////////////|   0
              +--------------------+--------------------+
            3 |////////////////////| calling job no. * 2 |   2 FQJOB
              +--------------------+--------------------+
FQSIZM      5 | job's console KB no.| status job no. * 2 |   4 FQFIL
              +--------------------+--------------------+
            7 |job's current D-space|job's current I-space|   6 FQPPN
              +--------------------+--------------------+
           11 |                                         |  10 FQNAM1
              +      job's current privilege mask       +
           13 |                                         |  12
              +               (4 words)                 +
           15 |                                         |  14 FQEXT
              +                                         +
           17 |                                         |  16 FQSIZ
              +--------------------+--------------------+
           21 |////////////////////|  job's access type  |  20 FQBUFL
              +--------------------+--------------------+
           23 |/////////////////////////////////////////|  22
              +--------------------+--------------------+
           25 |/////////////////////////////////////////|  24
              +--------------------+--------------------+
           27 |/////////////////////////////////////////|  26
              +--------------------+--------------------+
           31 |/////////////////////////////////////////|  30
              +--------------------+--------------------+
           33 |/////////////////////////////////////////|  32
              +--------------------+--------------------+
           35 |/////////////////////////////////////////|  34
              +--------------------+--------------------+
           37 |/////////////////////////////////////////|  36
              +--------------------+--------------------+
```

FIRQB+FQSIZM	If this value is negative, the job is detached, and the value is the one's complement of the keyboard number to which the job was previously attached.
FIRQB+FQPPN	The monitor returns the job's current I-space size in FIRQB+FQPPN and the job's current D-space size in FIRQB+FQPPN+1.
FIRQB+FQNAM1	The monitor returns the job's current privilege mask in four words beginning at FIRQB+FQNAM1.

FIRQB+FQBUFL The monitor returns the job's access type in this byte. The following values are defined:

Value	Meaning
0	Local
1	Dial-up
2	Batch
4	Network (SET HOST)
6	Network server

All other values are reserved.

Errors

BADFUO	The job number specified at FIRQB+FQFIL is less than zero or greater than JOB MAX.
PRVIOL	The job specified at FIRQB+FQFIL does not exist.

UU.TB1 (Get Monitor Tables, Part I)

Privileges Required

None

Data Passed

```
                                FIRQB
Mne-   Octal                                               Octal   Mne-
monic  Offset                                              Offset  monic
           +-----------------------+----------------------+
        1  |//////////////////////////////////////////////|  0
           +-----------------------+----------------------+
FQFUN   3  |     UU.TB1 (= -3)     |//////////////////////|  2
           +-----------------------+----------------------+
        5  |//////////////////////////////////////////////|  4
           +-----------------------+----------------------+
        7  |//////////////////////////////////////////////|  6
           +-----------------------+----------------------+
       11  |//////////////////////////////////////////////| 10
           +-----------------------+----------------------+
       13  |//////////////////////////////////////////////| 12
           +-----------------------+----------------------+
       15  |//////////////////////////////////////////////| 14
           +-----------------------+----------------------+
       17  |//////////////////////////////////////////////| 16
           +-----------------------+----------------------+
       21  |//////////////////////////////////////////////| 20
           +-----------------------+----------------------+
       23  |//////////////////////////////////////////////| 22
           +-----------------------+----------------------+
       25  |//////////////////////////////////////////////| 24
           +-----------------------+----------------------+
       27  |//////////////////////////////////////////////| 26
           +-----------------------+----------------------+
       31  |//////////////////////////////////////////////| 30
           +-----------------------+----------------------+
       33  |//////////////////////////////////////////////| 32
           +-----------------------+----------------------+
       35  |//////////////////////////////////////////////| 34
           +-----------------------+----------------------+
       37  |//////////////////////////////////////////////| 36
           +-----------------------+----------------------+
```

Data Returned

```
                              FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                            Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |/////////////////////| current job no. * 2 |  2 FQJOB
          +---------------------+---------------------+
FQSIZM  5 | MAXCNT(max. job no.)|CNT.KB-1(max. KB no.)|  4 FQFIL
          +---------------------+---------------------+
        7 |  (DEVCNT) address of max. unit no. table  |  6 FQPPN
          +---------------------+---------------------+
       11 | (DEVPTR) address of pointers to dev. DDBs | 10 FQNAM1
          +---------------------+---------------------+
       13 |(MEMLST) root link word in 1st mem ctl sblk| 12
          +---------------------+---------------------+
       15 |     (JOBTBL) address of the job table     | 14 FQEXT
          +---------------------+---------------------+
       17 |   (JBSTAT) address of job status table    | 16 FQSIZ
          +---------------------+---------------------+
       21 |  (JBWAIT) address of job wait flag table  | 20 FQBUFL
          +---------------------+---------------------+
       23 | (UNTCLU) addr of unit owners/pk cl sz tbl | 22 FQMODE
          +---------------------+---------------------+
       25 |   (UNTCNT) address of disk status table   | 24 FQFLAG
          +---------------------+---------------------+
       27 |   (SATCTL) address of free-block table    | 26 FQPFLG
          +---------------------+---------------------+
       31 |(JSBTBL) adr of job stat ordered by drv idx| 30 FQDEV
          +---------------------+---------------------+
       33 |  (SATCTM) addr of free block count table  | 32 FQDEVN
          +---------------------+---------------------+
       35 |  current date in system internal format   | 34 FQCLUS
          +---------------------+---------------------+
       37 |    (UNTOPT) addr of unit option table     | 36 FQNENT
          +---------------------+---------------------+
```

FIRQB+FQCLUS   System internal format for date is:

[(year - 1970) * 1000.] + day-within-year

Errors

No errors are possible with UU.TB1.

UU.TB2 (Get Monitor Tables, Part II)

Privileges Required

None

Data Passed

```
                                    FIRQB
Mne-   Octal                                                          Octal   Mne-
monic  Offset                                                         Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.TB2 (= -14)   |///////////////////////|  2
          +---------------------+---------------------+
        5 |/////////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |/////////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |/////////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |/////////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |/////////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |/////////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |/////////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |/////////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |/////////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Data Returned

```
                              FIRQB
Mne- Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |/////////////////////| current job no. * 2 |  2 FQJOB
          +---------------------+---------------------+
        5 |   (FREES) addr of free buffer info table  |  4 FQFIL
          +---------------------+---------------------+
        7 |     (DEVNAM) addr of device name table    |  6 FQPPN
          +---------------------+---------------------+
       11 |        (CSRTBL) addr of CSR table         | 10 FQNAM1
          +---------------------+---------------------+
       13 |       (DEVOKB) no. disks in DEVNAM * 2    | 12
          +---------------------+---------------------+
       15 | (TTYHCT) no. hung term errs since startup | 14 FQEXT
          +---------------------+---------------------+
       17 | (JOBCNT) no. jobs now/no. logins allowed  | 16 FQSIZ
          +---------------------+---------------------+
       21 | (RTSLST) rt. lnk.word in RTS dsc.blk.list | 20 FQBUFL
          +---------------------+---------------------+
       23 |    (ERLCTL) error logging control data    | 22 FQMODE
          +---------------------+---------------------+
       25 | (SNDLST) list of eligible msg. rec. jobs  | 24 FQFLAG
          +---------------------+---------------------+
       27 |        (DSKLOG) disk logical table        | 26 FQPFLG
          +---------------------+---------------------+
       31 | (DEVSYN) start of synonym names in DEVNAM | 30 FQDEV
          +---------------------+---------------------+
       33 | (MEMSIZ) physical memory size, K-words*32 | 32 FQDEVN
          +---------------------+---------------------+
       35 | (CCLLST)root link word of CCL desc. blks. | 34 FQCLUS
          +---------------------+---------------------+
       37 |  (FCBLST) addr of table of roots of FCBs  | 36 FQNENT
          +---------------------+---------------------+
```

Errors

No errors are possible with UU.TB2.

UU.TB3 (Get Monitor Tables, Part III)

Privileges Required

None

Data Passed

```
                                 FIRQB
Mne-   Octal                                                Octal   Mne-
monic Offset                                                Offset  monic
            +---------------------+---------------------+
         1  |///////////////////////////////////////////|  0
            +---------------------+---------------------+
FQFUN    3  |    UU.TB3 (= -29)   |/////////////////////|  2
            +---------------------+---------------------+
         5  |///////////////////////////////////////////|  4
            +---------------------+---------------------+
         7  |///////////////////////////////////////////|  6
            +---------------------+---------------------+
        11  |///////////////////////////////////////////| 10
            +---------------------+---------------------+
        13  |///////////////////////////////////////////| 12
            +---------------------+---------------------+
        15  |///////////////////////////////////////////| 14
            +---------------------+---------------------+
        17  |///////////////////////////////////////////| 16
            +---------------------+---------------------+
        21  |///////////////////////////////////////////| 20
            +---------------------+---------------------+
        23  |///////////////////////////////////////////| 22
            +---------------------+---------------------+
        25  |///////////////////////////////////////////| 24
            +---------------------+---------------------+
        27  |///////////////////////////////////////////| 26
            +---------------------+---------------------+
        31  |///////////////////////////////////////////| 30
            +---------------------+---------------------+
        33  |///////////////////////////////////////////| 32
            +---------------------+---------------------+
        35  |///////////////////////////////////////////| 34
            +---------------------+---------------------+
        37  |///////////////////////////////////////////| 36
            +---------------------+---------------------+
```

Data Returned

```
                            FIRQB
Mne-  Octal                                         Octal   Mne-
monic Offset                                        Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |//////////////////////| current job no. * 2 |  2 FQJOB
          +---------------------+---------------------+
        5 | (DDCTBL) addr of controller/device table   |  4 FQFIL
          +---------------------+---------------------+
        7 |  (UCTTBL) addr of unit/controller table    |  6 FQPPN
          +---------------------+---------------------+
       11 |      (SATEND) addr of disk size table      | 10 FQNAM1
          +---------------------+---------------------+
       13 |(UNTLVL) addr of disk structure level table| 12
          +---------------------+---------------------+
       15 |      (MFDPTR) addr of MFD point table      | 14 FQEXT
          +---------------------+---------------------+
       17 |(MAGLBL) addr of mag tape labeling default | 16 FQSIZ
          +---------------------+---------------------+
       21 |//////////////////////| no. jobs on system  | 20 FQBUFL
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 | (DSTPTR) address of disk statistics table | 24 FQFLAG
          +---------------------+---------------------+
       27 |        hardware configuration word         | 26 FQPFLG
          +---------------------+---------------------+
       31 |(UNTERR) address of unit error count table | 30 FQDEV
          +---------------------+---------------------+
       33 |(DEVCLU) addr of unit dev cl sz/cl fac tbl | 32 FQDEVN
          +---------------------+---------------------+
       35 |    (NULRTS) address of NULL RTS block      | 34 FQCLUS
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFLAG	The Memory Management Unit (MMU) address of the disk statistics table, if this monitor was generated with the unsupported disk statistics feature. Otherwise, FIRQB+FQFLAG is zero.

FIRQB+FQPFLG Flag word bit settings indicate hardware configuration attributes:

Bit	Setting	Meaning
5	Clear	Unibus system
	Set	QBUS system
9	Clear	FPP not available
	Set	FPP available
10	Clear	CIS not available
	Set	CIS available
13	Clear	System does not have I/D space
	Set	System has I/D space

Errors

No errors are possible with UU.TB3.

UU.TRM (Set Terminal Characteristics)

Privileges Required

HWCFG is required to set a terminal other than the caller's own, or to set permanent characteristics.

Data Passed (subfunction= 0)

FIRQB

Mne-monic	Octal Offset			Octal Offset	Mne-monic
	1	//		0	
FQFUN	3	UU.TRM (= 20)	//////////////////////	2	
FQSIZM	5	377(current) or KB no	subfunction code =0	4	FQFIL
	7	200(tab)/377(no tab)	width: 0 or width+1	6	FQPPN
	11	200(lwc out)/377(no)	200(no frm)/377(form)	10	FQNAM1
	13	200(fdplx)/377(local)	200(no XON)/377(XON)	12	
	15	200(no)/377(lwc in)	200(no scp)/377(scp)	14	FQEXT
	17	rcv baud rate	FILL fctr+1;1(noFILL)	16	FQSIZ
	21	xmt baud rate	PARITY;1(no)2(e)3(od)	20	FQBUFL
	23	200(^)/377(no ^)	200(no stl)/377(stl)	22	FQMODE
	25	parameters	//////////////////////	24	FQFLAG
FQPROT	27	200(no)/377(esc seq)	PERMANENT flag	26	FQPFLG
	31	200(no esc)/377(esc)	private delimiter	30	FQDEV
	33	RSUM;200(^C)/377(any)	200(no ^R/377(^R)	32	FQDEVN
	35	200(no GAG)/377(GAG)	BRAK;200(nul)/377(^C)	34	FQCLUS
	37	//		36	

FIRQB+FQEXT	Setting No Scope automatically sets No Stall. Setting Scope automatically sets Stall.

FIRQB+FQSIZ+1 Receiver baud rate is set as follows:

0	No change
n	The internal code to determine the baud rate at which the terminal receives characters. If FIRQB+FQBUFL+1 is zero, this code also determines the transmit (output) baud rate. (Requires HWCTL privilege.) Note that you can set internal codes only for DH11, DZ11/DZV11, and DHU11/DHV11 interface lines as follows:

DH11		DZ11/DZV11		DHU11/DHV11	
Code	Speed	Code	Speed	Code	Speed
1	0	1	0	1	0
2	50	2	50	2	75
3	75	3	75	3	110
4	110	4	110	4	134.5
5	134.5	5	134.5	5	150
6	150	6	150	6	300
7	200	7	300	7	600
8	300	8	600	8	1200
9	600	9	1200	9	1800
10	1200	10	1800	10	2000
11	1800	11	2000	11	2400
12	2400	12	2400	12	4800
13	4800	13	3600	13	Res.
14	9600	14	4800	14	9600
		15	7200	15	19200
		16	9600		

FIRQB+FQBUFL Parity and 8-bit mode values:

1	No parity
2	Even parity
3	Odd parity
4	Enable parity characteristic
20	8-bit characteristic off
30	8-bit characteristic on

FIRQB+FQBUFL+1 Transmitter baud rate is set as follows:

0	Both the receive (input) and transmit (output) speeds are determined by the value n in FIRQB+FQSIZ+1.
n	The internal code to determine the baud rate at which the terminal transmits characters when a split speed setting is used. (HWCTL privilege required.) You can use split speed settings only with the DH11 interface line.

FIRQB+FQFLAG+1 Value=10+DATA+STOP+PARITY

where:

DATA is	0	for 5-bit characters
	1	for 6-bit characters
	2	for 7-bit characters
	3	for 8-bit characters
STOP is	0	for 1 stop bit per character
	4	for 2 stop bits per character or 1.5 bits if DATA = 0
PARITY is	0	for no parity bit
	20	for even parity format
	60	for odd parity format

This byte applies only to interfaces that support DATA/STOP/PARITY features. When this byte is used with these interfaces, it overrides the setting of FIRQB+FQBUFL. In addition, when this byte is used, FIRQB+FQSIZ+1 (receiver baud rate) must be nonzero.

FIRQB+FQPFLG The PERMANENT flag is as follows:

0	No change
200	Return permanent characteristics
377	Set permanent characteristics

FIRQB+FQDEV Determines the use of private delimiters as follows:

0	No change.
200	Disable (clear) the private delimiter.
200+n	Set the private delimiter to ASCII code n (in the range 1 to 127). If the character has a special meaning (for example, horizontal tab or the CTRL/Z combination), the private delimiter usage has higher precedence.

Data Returned (Current Keyboard Characteristics)

FIRQB

Mnemonic	Octal Offset			Octal Offset	Mnemonic
	1	//////////////////////	//////////////////////	0	
	3	//////////////////////	current job no. * 2	2	FQJOB
FQSIZM	5	255(current) or KB no	KB status	4	FQFIL
	7	tab/no tab	width: 0 or width+1	6	FQPPN
	11	lc outpt/no lc outpt	no form/form	10	FQNAM1
	13	full dplx/local echo	no XON/XON	12	
	15	no lc input/lc input	no scope/scope	14	FQEXT
	17	receive baud rate	no fill/fill	16	FQSIZ
	21	transmit baud rate	no parity/even/odd	20	FQBUFL
	23	up arrow/no up arrow	no stall/stall	22	FQMODE
	25	//////////////////////	interface for line	24	FQFLAG
FQPROT	27	no esc seq/esc seq	PERMANENT flag	26	FQPFLG
	31	no esc/esc	delimiter/delim x	30	FQDEV
	33	resume CTRL/C or any	CTRL/R,CTRL/T flag	32	FQDEVN
	35	enable/dis broadcast	BREAK=null or CTRL/C	34	FQCLUS
	37	//////////////////////	8-bit characteristics	36	FQNENT

FIRQB+FQFIL Bit 0 when set means the keyboard is disabled or is a pseudo keyboard that is not in use.

When bits 1-6 are clear, it means no job owns the keyboard. Otherwise, the value in bits 1-6 is equal to the job number times two of the job that owns keyboard.

Bit 7 when set means the modem line is hung up or is a pseudo keyboard that is not in use.

Errors

BADFUO	The system returns this error for one of two possibilities: o Keyboard number is out of range of valid numbers. o Current keyboard is specified but is detached.
PRVIOL	You do not have HWCFG privilege and tried to either: o Read the characteristics of a terminal not opened or assigned to your job. o Change the characteristics of a terminal other than your job's console terminal (KB:). o Change the speed setting for your terminal. (FIRQB+FQSIZ+1 or FIRQB+FQBUFL+1 is nonzero.) o Set the permanent characteristics for a dial-up terminal, pseudo keyboard, or local terminal. (FIRQB+FQPFLG is nonzero.)

Data Passed (Subfunction = 1)

```
                              FIRQB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
             +---------------------+---------------------+
          1  |///////////////////////////////////////////|  0
             +---------------------+---------------------+
FQFUN     3  |    UU.TRM (= 20)    |/////////////////////|  2
             +---------------------+---------------------+
FQSIZM    5  |377(current) or KB no| subfunction code=1  |  4 FQFIL
             +---------------------+---------------------+
          7  |/////////////////////| terminal type code  |  6 FQPPN
             +---------------------+---------------------+
         11  |///////////////////////////////////////////| 10
             +---------------------+---------------------+
         13  |///////////////////////////////////////////| 12
             +---------------------+---------------------+
         15  |///////////////////////////////////////////| 14
             +---------------------+---------------------+
         17  |///////////////////////////////////////////| 16
             +---------------------+---------------------+
         21  |///////////////////////////////////////////| 20
             +---------------------+---------------------+
         23  |///////////////////////////////////////////| 22
             +---------------------+---------------------+
         25  |/////////////////////|/////////////////////| 24
             +---------------------+---------------------+
FQPROT   27  | input buffer quota  |   PERMANENT flag    | 26 FQPFLG
             +---------------------+---------------------+
         31  | clear control chars |  set control chars  | 30 FQDEV
             +---------------------+---------------------+
         33  |       set terminal capability flags       | 32 FQDEVN
             +---------------------+---------------------+
         35  |      clear terminal capability flags      | 34 FQCLUS
             +---------------------+---------------------+
         37  |///////////////////////////////////////////| 36
             +---------------------+---------------------+
```

FIRQB+FQPPN   Terminal type codes. The following decimal values are valid; zero causes no change to occur:

code	type	code	type	code	type
0	no chg	12	LA50	24	LA30
1	Unknown	13	VT101	25	VT50
2	LA36	14	VT102	26	VT50H
3	VT52	15	VT125	27	VT05
4	VT55	16	VT131	28	VT05B
5	LA180S	17	VT132	29	LA30S
6	VT100	18	VT220	30	2741
7	LA120	19	VT240	31	ASR33
8	LA12	20	VT241	32	KSR33

code	type	code	type	code	type
9	LA100	21	VT105	33	ASR35
10	LA34	22	VK100	34	KSR35
11	LA38	23	RT02		
35 to 128	Reserved for DIGITAL use				
129 to 255	Customer definable				

FIRQB+FQPFLG The PERMANENT flag is as follows:

0	No change
200	Return permanent characteristics
377	Set permanent characteristics

FIRQB+FQPROT The terminal's input buffer quota is as follows:

0	No change
6 to 255	The maximum number of input buffers. The default value is 6. There are 30 characters per buffer. The /BUFFER_QUOTA qualifier of the SET TERMINAL command sets this characteristic. Users need HWCFG privilege to use this feature.

Note

There is no guarantee that a terminal can allocate its full buffer quota.

Excessive use of this feature to allocate large amounts of buffers to several terminals could create a shortage of small buffers.

FIRQB+FQDEV Setting control character functions or auto-baud. The bits currently defined are:

0	Set CTRL/C
1	Set CTRL/T
2	Set CTRL/R
3	Set CTRL/X
4	Set autobaud

Bits can be combined; that is one or more bits can be set at any time. If no bits are set, then the current setting is unchanged. All other bits are reserved and should be zero.

FIRQB+FQDEV+1	Clearing control character functions or auto-baud. The bits currently defined are:

0	Clear CTRL/C
1	Clear CTRL/T
2	Clear CTRL/R
3	Clear CTRL/X
4	Clear autobaud

Bits can be combined; that is one or more bits can be cleared at any time. If no bits are cleared, then the current setting is unchanged. All other bits are reserved and should be zero.

FIRQB+FQDEVN	Set terminal capability flags. The following bits are currently defined:

0	Set ANSI escape sequences
1	Set advanced video
2	Set width to 132 columns
3	Set printer port
4	Set ReGIS graphics
5	Set SIXEL graphics
6	Set Katakana character set
7	Set selectively erasable characters
8	Set dynamically redefinable character sets (DRCS)
9	Set user defined keys (UDKs)
10	Set local copy

If no bits are set, the current capability status is unchanged. All other bits are reserved and should be zero.

FIRQB+FQCLUS	Clear terminal capability flags. The following bits are currently defined:

0	Clear ANSI escape sequences
1	Clear advanced video
2	Clear 132 column flag
3	Clear printer port
4	Clear ReGIS graphics
5	Clear SIXEL graphics
6	Clear Katakana character set
7	Clear selectively erasable characters
8	Clear dynamically redefinable character set (DRCS)
9	Clear user defined keys (UDKs)
10	Clear local copy

If no bits are set, the current capability status is unchanged. All other bits are reserved and should be zero.

Data Returned (Current Terminal Characteristics)

```
                                  FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                 Offset  monic
         +--------------------+--------------------+
      1  |/////////////////////////////////////////|  0
         +--------------------+--------------------+
      3  |/////////////////////////////////////////|  2
         +--------------------+--------------------+
      5  |/////////////////////////////////////////|  4
         +--------------------+--------------------+
      7  |////////////////////|current terminal type|  6 FQPPN
         +--------------------+--------------------+
     11  |/////////////////////////////////////////| 10
         +--------------------+--------------------+
     13  |/////////////////////////////////////////| 12
         +--------------------+--------------------+
     15  |/////////////////////////////////////////| 14
         +--------------------+--------------------+
     17  |/////////////////////////////////////////| 16
         +--------------------+--------------------+
     21  |/////////////////////////////////////////| 20
         +--------------------+--------------------+
     23  |/////////////////////////////////////////| 22
         +--------------------+--------------------+
     25  |/////////////////////////////////////////| 24
         +--------------------+--------------------+
     27  |////////////////////|   PERMANENT flag    | 26 FQPFLG
         +--------------------+--------------------+
     31  |////////////////////|current ctrl settings| 30 FQDEV
         +--------------------+--------------------+
     33  | current terminal capability flag settings | 32 FQDEVN
         +--------------------+--------------------+
     35  |/////////////////////////////////////////| 34
         +--------------------+--------------------+
     37  |/////////////////////////////////////////| 36
         +--------------------+--------------------+
```

Errors

BADFUO	The system returns this error for one of three possibilities:

- o Keyboard number is out of range of valid numbers.
- o Current keyboard is specified but is detached.
- o The terminal's input buffer quota is out of range.

PRVIOL	You do not have the required privileges and you tried to either:

- o Read the characteristics of a terminal not opened or assigned to your job.
- o Change the characteristics of a terminal other than your job's console terminal (KB:).
- o Change the speed setting for your terminal. (FIRQB+FQSIZ+1 or FIRQB+FQBUFL+1 is nonzero.)
- o Set the permanent characteristics for a dial-up terminal, pseudo keyboard, or local terminal. (FIRQB+FQPFLG is nonzero.)
- o Change the terminal's input buffer quota.

UU.YLG (Enable Logins)

Privileges Required

SWCTL is required to enable logins.

Data Passed

```
                               FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.YLG (= -1)    |/////////////////////|  2
          +---------------------+---------------------+
        5 |/////////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |/////////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |/////////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |/////////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |/////////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |/////////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |/////////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |/////////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |/////////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |/////////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |/////////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |/////////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |/////////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |/////////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Data Returned

```
                                FIRQB
Mne-   Octal                                              Octal   Mne-
monic Offset                                              Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |/////////////////////| current job no. * 2 |  2 FQJOB
          +---------------------+---------------------+
        5 |/////////////////////| no. logins allowed  |  4 FQFIL
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

Errors

No errors are possible with UU.YLG.

UU.ZER (Zero Device)

Privileges Required

DEVICE is required to access a restricted device. Create/rename access (matching PPN, or GWRITE, WWRITE and/or SYSIO privilege) required to zero an account other than caller's. Set account access (GACNT or WACNT privilege) is required to deallocate the UFD when zeroing an account.

Data Passed

```
                                  FIRQB
Mne-  Octal                                                 Octal   Mne-
monic Offset                                                Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
FQFUN   3 |    UU.ZER (= 15)    |/////////////////////|  2
          +---------------------+---------------------+
        5 |/////////////////////|  flag (disk only)   |  4 FQFIL
          +---------------------+---------------------+
        7 |   project number    | programmer number   |  6 FQPPN
          +---------------------+---------------------+
       11 |    volume ID for label in RAD50 format    | 10 FQNAM1
          +                 (2 words)                 +
       13 |        (ANSI format mag tape only)        | 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |  device name (disk, magtape, or DECtape)  | 30 FQDEV
          +---------------------+---------------------+
       33 |≠0, unit number real | device unit number  | 32 FQDEVN
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
```

FIRQB+FQFIL	Zero means delete all files except those write-protected against owner; retain UFD clusters. Minus one means delete all files regardless of protection code; release UFD clusters (users with GACNT or WACNT privilege only). Users without GACNT or WACNT privileges can use this flag to delete all files regardless of their protection codes, but cannot release UFD clusters.
FIRQB+FQPPN	Users without GACNT or WACNT privilege can specify the current account only. Be sure to include the PPN; the system returns an error if both bytes are zero.

Data Returned

Except for a possible error in byte 0 of the FIRQB, the UU.ZER subfunction does not return any meaningful data.

Errors

BADFUO	No PPN was specified.
BADNAM	The specified device is ANSI magnetic tape, and the volume ID is either missing or invalid.
DEVNFS	The specified device does not allow access by file name.
NODEVC	The device specified is not on the system.
NOSUCH	The account specified does not exist on the device specified.
NOTAVL	The device specified exists on the system, but you cannot zero it for one of the following reasons: o A file is currently open on the device. o The device is currently reserved by another job. o The device or its controller is disabled.
PRVIOL	The unit is write-locked, the unit is restricted and the caller does not have DEVICE privilege, or the caller does not have accounting privileges and FIRQB+FQPPN contains a PPN other than the current account.

UU.3PP (Third-Party Privilege Checking)

The UU.3PP directive enables or disables third-party privilege checking. This directive is for the use of server programs that run as a separate job; for example, spoolers.

Once issued, the directive remains in effect until cancelled by another UU.3PP with zero in FIRQB+FQPPN, or by exit from the calling program. While in effect, every privilege check done by the monitor is made twice: once for the job's PPN and privilege mask, and once for the PPN and privilege mask specified in the call. The function proceeds only when the check passes in both cases.

Privileges Required

None

Data Passed

```
                              FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
           +---------------------+---------------------+
        1  |///////////////////////////////////////////|  0
           +---------------------+---------------------+
FQFUN   3  |    UU.3PP (= 37)    |/////////////////////|  2
           +---------------------+---------------------+
        5  |///////////////////////////////////////////|  4
           +---------------------+---------------------+
        7  |   project number    | programmer number   |  6 FQPPN
           +---------------------+---------------------+
       11  |                                           | 10 FQNAM1
           +                                           +
       13  |              privilege mask               | 12
           +                (4 words)                  +
       15  |                                           | 14
           +                                           +
       17  |                                           | 16
           +---------------------+---------------------+
       21  |///////////////////////////////////////////| 20
           +---------------------+---------------------+
       23  |///////////////////////////////////////////| 22
           +---------------------+---------------------+
       25  |///////////////////////////////////////////| 24
           +---------------------+---------------------+
       27  |///////////////////////////////////////////| 26
           +---------------------+---------------------+
       31  |///////////////////////////////////////////| 30
           +---------------------+---------------------+
       33  |///////////////////////////////////////////| 32
           +---------------------+---------------------+
       35  |///////////////////////////////////////////| 34
           +---------------------+---------------------+
       37  |///////////////////////////////////////////| 36
           +---------------------+---------------------+
```

FIRQB+FQPPN	Specify zero to cancel third-party privilege checking.

Data Returned

None

Errors

NOBUFS	No small buffers available.

.WRITA -- Write Data to File or Device (Asynchronously)

Form

.WRITA

Function

The .WRITA directive writes a specified number of bytes of data from a user buffer (defined in the XRB) to a file or device currently open on a channel. Unless an error occurs, the .WRITA always transfers the number of bytes specified. The number of bytes specified must be less than or equal to the size of the buffer. Table 3-11 gives specific details for each device (see the discussion for the .WRITE directive). For comparison, Table 3-11 also shows the "best guess" buffer sizes returned by the monitor at FIRQB+FQBUFL when the file or device is opened.

Privileges Required

EXQTA is required to write in excess of disk quota. TUNE is required for asynchronous I/O.

Data Passed

```
                              FIRQB
Mne-  Octal                                             Octal   Mne-
monic Offset                                            Offset  monic
           +---------------------+---------------------+
         1 |///////////////////////////////////////////|  0
           +---------------------+---------------------+
         3 |///////////////////////////////////////////|  2
           +---------------------+---------------------+
         5 |        completion routine address         |  4 FQFIL
           +---------------------+---------------------+
         7 |          user-supplied parameter          |  6 FQPPN
           +---------------------+---------------------+
        11 |///////////////////////////////////////////| 10
           +---------------------+---------------------+
        13 |///////////////////////////////////////////| 12
           +---------------------+---------------------+
        15 |///////////////////////////////////////////| 14
           +---------------------+---------------------+
        17 |///////////////////////////////////////////| 16
           +---------------------+---------------------+
        21 |///////////////////////////////////////////| 20
           +---------------------+---------------------+
        23 |///////////////////////////////////////////| 22
           +---------------------+---------------------+
        25 |///////////////////////////////////////////| 24
           +---------------------+---------------------+
        27 |///////////////////////////////////////////| 26
           +---------------------+---------------------+
        31 |///////////////////////////////////////////| 30
           +---------------------+---------------------+
        33 |///////////////////////////////////////////| 32
           +---------------------+---------------------+
        35 |///////////////////////////////////////////| 34
           +---------------------+---------------------+
        37 |///////////////////////////////////////////| 36
           +---------------------+---------------------+
```

FIRQB+FQFIL	The address in the user job where program flow should begin when the asynchronous write completes. If this value is zero, no AST completion is executed. In this event, it is understood that the user is using some other mechanism to determine whether the I/O successfully completed.
FIRQB+FQPPN	The user job uses this field to communicate information to the AST completion routine. If FIRQB+FQFIL is zero, this value is ignored.

```
                              XRB
Mne-  Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
         +-------------------+-------------------+
       1 |     length of output buffer, in bytes     |  0 XRLEN
         +-------------------+-------------------+
       3 |       number of bytes to be written       |  2 XRBC
         +-------------------+-------------------+
       5 |         starting address of buffer        |  4 XRLOC
         +-------------------+-------------------+
XRBLKM 7 | MSB of block number | channel number * 2 |  6 XRCI
         +-------------------+-------------------+
      11 |            LSB of block number            | 10 XRBLK
         +-------------------+-------------------+
      13 |///////////////////////////////////////////| 12
         +-------------------+-------------------+
      15 |             optional modifier             | 14 XRMOD
         +-------------------+-------------------+
```

XRB+XRLEN	Length of the output buffer, in bytes. The value of this word must be nonzero.
XRB+XRBC	The number of bytes to be written. The value of this word must be nonzero and less than or equal to the buffer size, as specified at offset XRB+XRLEN. For disk, this must be a multiple of 512.
XRB+XRLOC	Starting address of the output buffer. For disk, flexible diskette, and magnetic tape devices, this address must be on a word (even) boundary. For all other devices, the buffer can begin on an odd address. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)
XRB+XRCI	Channel number times two; defines the channel on which the data is to be written, as previously defined in an open (OPNFQ, CREFQ, CRTFQ, CRBFQ functions of CALFIP).
XRB+XRBLKM	For disk files, this byte contains the most significant bits of the starting block number where the write is to begin. This byte is combined with the word at XRB+XRBLK to form a 24-bit field.
XRB+XRBLK	Starting block number where the write is to begin. Performs the same action as the BLOCK option for disk or the RECORD option for flexible diskette and non-file-structured DECtape (see the *RSTS/E Programming Manual*). This parameter is ignored if the device is not a random-access device. If the device is random-access, and this field is nonzero, it is interpreted as the block number where the write is to start (1 to n, where n is the length of the file in

	512-byte blocks). If zero, the next sequential block is written.
	For disk devices opened as non-file-structured in cluster mode, this value is the starting device cluster number where the write is to begin.
XRB+XRMOD	Output operation modifier; significant only for line printer, terminal, and DMC/DMR devices. (The monitor informs you with the FLGMOD bit of the flag word whether or not the device accepts modifiers.) This parameter performs the same action as the BASIC-PLUS RECORD modifier for these devices (see the *RSTS/E Programming Manual*).

Data Returned to Completion Routine

```
                                   FIRQB
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////|      error code     |  0
          +---------------------+---------------------+
        3 |/////////////////////|   job number * 2    |  2 FQJOB
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |          user supplied parameter          |  6 FQPPN
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
       21 |///////////////////////////////////////////| 20
          +---------------------+---------------------+
       23 |///////////////////////////////////////////| 22
          +---------------------+---------------------+
       25 |///////////////////////////////////////////| 24
          +---------------------+---------------------+
       27 |///////////////////////////////////////////| 26
          +---------------------+---------------------+
       31 |///////////////////////////////////////////| 30
          +---------------------+---------------------+
       33 |///////////////////////////////////////////| 32
          +---------------------+---------------------+
       35 |///////////////////////////////////////////| 34
          +---------------------+---------------------+
       37 |///////////////////////////////////////////| 36
          +---------------------+---------------------+
                               XRB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
           +----------------------+---------------------+
         1 |////////////////////////////////////////////| 0
           +----------------------+---------------------+
         3 |       number of bytes not written          | 2 XRBC
           +----------------------+---------------------+
         5 |////////////////////////////////////////////| 4
           +----------------------+---------------------+
XRBLKM   7 | MSB of block number  |/////////////////////| 6
           +----------------------+---------------------+
        11 |     block number where write began         | 10 XRBLK
           +----------------------+---------------------+
        13 |////////////////////////////////////////////| 12
           +----------------------+---------------------+
        15 |////////////////////////////////////////////| 14
           +----------------------+---------------------+
```

XRB+XRBC	The number of bytes not written in this .WRITE operation. This value is usually zero.
XRB+XRBLKM	For disk files, the MSB of the block number of the block just written. (Combined with XRB+XRBLK to form a 24-bit field.)
XRB+XRBLK	For random-access devices, this word contains the block number of the block just written. Block numbers range from 1 to n, where n is the length of the file, in 512-byte blocks. They define the order in which the file is written. For disk devices opened as non-file-structured in cluster mode, this is the device cluster number of the cluster just written.

Errors

BADCNT	The completion routine address is invalid or odd.
BSERR	The specified channel number is illegal; that is, it is not in the range 0 - 36 (since channel is specified as channel * 2), or it is an odd number.
EOF	A write was attempted to a block beyond the end of the disk when the disk was opened non-file structured.
INUSE	Too much outstanding I/O for the job (over quota).
NOBUFS	No small buffers were available for the request.
NOTOPN	No device is currently open on the specified channel.

PRVIOL	One of the following occurred: o The file or device currently open on the specified channel is read-only. o The open for this file or device did not specify write access. A write was requested that would extend a UFD or a contiguous file. o A write was attempted that would extend a file in special update mode and block 0 was not locked. o A write was attempted that would extend an executable file, increasing its size to more than 65,535 blocks.

All other errors are device-dependent. Common errors are:

DATERR	Some data error occurred. This error is issued when a parity error occurs, and so forth.
HNGDEV	Some hard device I/O error occurred. For example, the specified device is off-line, is physically write-locked, and so forth.
NOROOM	No more room is available on the device. For example, the end-of-tape marker has been encountered, no more available disk space, and so forth.

Example

See the description of .READA for an example using .WRITA.

.WRITE - Write Data to File or Device

Form

.WRITE

Function

The .WRITE directive writes a specified number of bytes of data from a user buffer (defined in the XRB) to a file or device currently open on a channel. Unless an error occurs, the .WRITE always transfers the number of bytes specified. The number of bytes specified must be less than or equal to the size of the buffer. Table 3-11 gives specific details for each device. For comparison, Table 3-11 also shows the best guess buffer sizes returned by the monitor at FIRQB+FQBUFL when the device is opened.

Table 3-11: Data Output with .WRITE

Device	Block Size, Bytes (Decimal)	"Best Guess," Bytes (Decimal)	Restrictions on Number of Bytes Written
Byte-Oriented Devices (FLGFRC=1, FLGRND=0)			
Keyboard (Terminal)	N/A	128	None
Pseudo Keyboard	N/A	128	None
Paper Tape Punch	N/A	128	None
Line Printer	N/A	128	None
Block-Sequential Devices (FLGFRC=0, FLGRND=1)			
Magnetic Tape	14 to 32,767	512	14 to 32,767 bytes
DECtape (file-structured)	510	510	Must be < 510 bytes; if < 510, automatically filled with NULLs to 510 bytes.
DMC/DMR	1 to 8000	512	None

Table 3-11: Data Output with .WRITE (Cont.)

Device	Block Size, Bytes (Decimal)	"Best Guess," Bytes (Decimal)	Restrictions on Number of Bytes Written
Block-Random Devices (FLGFRC=0, FLGRND=0)			
Disk	512	512	Must be multiple of 512 bytes. (Must be 512 if writing UFD.)
Flexible Diskette (RX01 and RX02)	*	512	If not a multiple of block/sector size, zero fill to end of block/sector.
DECtape (non-file-structured)	512	512	Must be < 512 bytes; if < 512, automatically filled with NULLs to 512 bytes.
* 512 (block mode) or 128 (RX01 or RX02 single-density sector mode) or 256 (RX02 double-density sector mode)			

Privileges Required

EXQTA is required to write in excess of disk quota.

Data Passed

```
                                 XRB
Mne-  Octal                                            Octal   Mne-
monic Offset                                           Offset  monic
          +---------------------+---------------------+
        1 |     length of output buffer, in bytes     |  0 XRLEN
          +---------------------+---------------------+
        3 |        number of bytes to be written      |  2 XRBC
          +---------------------+---------------------+
        5 |          starting address of buffer       |  4 XRLOC
          +---------------------+---------------------+
XRBLKM  7 | MSB of block number | channel number * 2  |  6 XRCI
          +---------------------+---------------------+
       11 |            LSB of block number            | 10 XRBLK
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |             optional modifier             | 14 XRMOD
          +---------------------+---------------------+
```

XRB+XRLEN	Length of the output buffer, in bytes. The value of this word must be nonzero.
XRB+XRBC	The number of bytes to be written. The value of this word must be nonzero and less than or equal to the buffer size, as specified at offset XRB+XRLEN. For disk, this must be some multiple of 512.
XRB+XRLOC	Starting address of the output buffer. For disk, flexible diskette, and magnetic tape devices, this address must be on a word (even) boundary. For all other devices, the buffer can begin on an odd address. (See the section "XRB (Transfer Request Block)" in Chapter 2 for more information.)
XRB+XRCI	Channel number times two; defines the channel on which the data is to be written, as previously defined in an open (OPNFQ, CREFQ, CRTFQ, CRBFQ functions of CALFIP).
XRB+XRBLKM	For disk files, this byte contains the MSB of the starting block number where the write is to begin. This byte is combined with the word at XRB+XRBLK to form a 24-bit field.
XRB+XRBLK	Starting block number where the write is to begin. Performs the same action as the BLOCK option for disk or the RECORD option for flexible diskette and non-file-structured DECtape (see the *RSTS/E Programming Manual*). This parameter is ignored if the device is not a random-access device. If the device is random-access, and this field is nonzero, it is interpreted as the block number where the write is to start (1 to n, where n is the length of the file in 512-byte blocks). If zero, the next sequential block is written. For disk devices opened as non-file-structured in cluster mode, this value is the starting device cluster number where the write is to begin.
XRB+XRMOD	Output operation modifier; significant only for line printer, terminal, and DMC/DMR devices. (The monitor informs you with the FLGMOD bit of the flag word whether or not the device accepts modifiers.) Except for modifier 20000, which is useful to MACRO programmers only, this parameter performs the same action as the BASIC-PLUS RECORD modifier for these devices (see the *RSTS/E Programming Manual*). Modifier 20000 provides a no stall option for line printer or terminal output. This modifier causes the monitor to return control to your program if an output

stall is to occur on the device. You can determine the number of bytes still to be written by checking the contents of XRB+XRBC.

Use this modifier for programs that must perform several different functions with optimal performance, such as a line printer spooler that performs message send/receive and prints files at the same time. When an output stall does occur, the program can perform other processing before trying to write the remaining bytes to the line printer or terminal. You may also find this modifier useful in multiterminal service applications.

When you use the No Stall option on a line printer, you can perform a special test to see whether the line printer is busy without causing your program to stall. To perform the test, write a null character and set modifier bits 2 and 13. When both bits are set, the system returns control to your program instead of stalling it. The value at XRB+XRBC tells you the status of the line printer:

XRB+XRBC = 0	The line printer buffers are empty; that is, there are no characters still to print.
XRB+XRBC ≠ 0	The line printer buffers still contain one or more characters to print. Repeat the test until the system returns zero at XRB+XRBC.

For more information on modifier bit 2 for line printers, see Appendix B and the *RSTS/E Programming*

Manual. The *RSTS/E Programming Manual* also contains more information on special programming techniques for line printers.

Data Returned

```
                               XRB
Mne-  Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
          +----------------------+----------------------+
        1 |/////////////////////////////////////////////|  0
          +----------------------+----------------------+
        3 |        number of bytes not written          |  2 XRBC
          +----------------------+----------------------+
        5 |/////////////////////////////////////////////|  4
          +----------------------+----------------------+
XRBLKM  7 | MSB of block number  |//////////////////////|  6
          +----------------------+----------------------+
       11 |      block number where write began         | 10 XRBLK
          +----------------------+----------------------+
       13 |/////////////////////////////////////////////| 12
          +----------------------+----------------------+
       15 |/////////////////////////////////////////////| 14
          +----------------------+----------------------+
```

XRB+XRBC	The number of bytes not written in this .WRITE operation. Except for terminals and line printers, this value is generally zero.
XRB+XRBLKM	For disk files, the most significant bits of the block number of the block just written. (Combined with XRB+XRBLK to form a 24-bit field.)
XRB+XRBLK	For random-access devices, this word contains the block number of the block just written. Block numbers range from 1 to n, where n is the length of the file, in 512-byte blocks. They define the order in which the file is written. For disk devices opened as non-file-structured in cluster mode, this is the device cluster number of the cluster just written.

Errors

BADCNT	The first three words of the XRB that describe the output buffer are illegal (an illegal byte count).
BSERR	The specified channel number is illegal; that is, it is not in the range 0 - 36 (since channel is specified as channel * 2), or it is an odd number.

EOF	A write was attempted to a block beyond the end of the disk when the disk was opened non-file structured.
NOTOPN	No device is currently open on the specified channel.
PRVIOL	One of the following occurred:

- o The file or device currently open on the specified channel is read-only.
- o The open for this file or device did not specify write access. A write was requested that would extend a UFD or a contiguous file.
- o A write was attempted that would extend a file in special update mode and block 0 was not locked.
- o A write was attempted that would extend an executable file, increasing its size to more than 65,535 blocks.

All other errors are device-dependent. Common errors are:

DATERR	Some data error occurred. This error is issued when a parity error occurs, and so forth.
HNGDEV	Some hard device I/O error occurred. For example, the specified device is off-line, is physically write-locked, and so forth.
NOROOM	No more room is available on the device. For example, the end-of-tape marker has been encountered, no more available disk space, and so forth.

Example

The following code writes the contents of a buffer to the file open on channel 5:

```
MOV     #1024.,XRB+XRLEN               ;SET BUFFER LENGTH
MOV     #1024.,XRB+XRBC                ;SET BYTES TO WRITE
MOV     #BUFFER,XRB+XRLOC              ;SET BUFFER ADDRESS
MOVB    #5*2,XRB+XRCI                  ;SET CHANNEL = 5
.WRITE
```

PART III
RSX and RT11 Emulator Directives

Chapter 1 describes the RSX run-time system as emulating the RSX operating system environment on RSTS/E. This is only partially true; the RSX emulator does not provide a real-time, multiprocessing environment to users of a RSTS/E time-sharing system. However, the RSX emulator does process directives that are identical in form, and similar in objective, to a subset of the RSX-11M-PLUS monitor directives that the *RSX-11M/M-PLUS Executive Reference Manual* describes.

If you use the RSX directives Chapter 5 describes, you must assemble your program with the MAC assembler and link the modules with the Task Builder (TKB). The program can then be run as a user job image under the control of the RSX emulator (which is part of the RSTS/E monitor).

Advantage: Transportable Code

The RSX emulator directives are useful if you are coding a program to be run under both the RSX-11M-PLUS and RSTS/E operating systems. However, RSTS/E does not emulate all RSX-11M-PLUS executive directives, and the meanings of those that are emulated are fitted to the RSTS/E environment.

To design a program to be run under both operating systems, you need to know how the directives work under RSX-11M-PLUS and under RSTS/E. This manual describes what the directives do under RSTS/E. Some comparison is made here to RSX-11M-PLUS, but you must refer to the *RSX-11M/M-PLUS Executive Reference Manual* for a complete description of how these directives work in the RSX-11M-PLUS environment.

Another benefit of using compatible directives is that you can execute such programs directly on VAX/VMS under the VAX-11 RSX Application Migration Executive (AME), which is similar in function to the RSX emulator. AME emulates most RSX-11M-PLUS executive directives.

General Services

Besides transportable code, the RSX emulator directives also provide:

- o Non-file-structured I/O -- The RSX emulator directives handle only non-file-structured I/O (for example, terminal I/O). For this type of I/O, using the directives takes less memory than using RMS. See the *RSTS/E RMS-11 MACRO Programmer's Guide* for a description of the record management service capabilities for MACRO programmers.

RMS routines are available to MACRO programmers through resident libraries or through linking to make them part of the user job image. In either case, the RMS routines take space in the job area. The RSX emulator directives that do I/O take no extra space, since the code to handle the processing is in the monitor.

- Trap handling -- The RSX emulator includes processing to handle synchronous and asynchronous exceptions. Mainly, the emulator aborts the program when such exceptions occur and prints an error message at the job's terminal. For some directives, you can request that this processing be bypassed and handle such exceptions in your program. For example, you can specify an address to pass control to when a Floating-Point Processor (FPP) exception occurs.

RSX Directive Emulation Within RSTS/E Monitor

RSX emulation is a standard feature of the RSTS/E monitor. This allows task images to execute without a run-time system, because the monitor handles the RSX emulation directly. As a result, task images may expand to be as large as 64K-32 words; furthermore, a resident library (for example, RMS) may use the address space which would otherwise be taken by the run-time system.

Since RSTS/E jobs are never without a run-time system, the monitor associates the job with a zero-length, run-time system called "...RSX" for the duration of the task's execution. When the task ends, the job exits to its default keyboard monitor.

System Macro Library

The RSX emulator directives that you code into your program are macro calls; they must be expanded into executable code at assembly time. The system macro library (LB:RSXMAC.SML) contains the macro expansions for the RSX emulator directives. This library file can be named in the input-file list when the assembly is done. For example:

```
MAC OBJ,OBJ=SRC1,SRC2,LB:RSXMAC.SML/ML
```

However, the MAC assembler searches the library automatically to resolve undefined symbols, so this is not really necessary.

In your code, though, you must use the MACRO-11 .MCALL directive to define the directives you use as external macros needed to assemble the source program. The .MCALL directive must appear before you call the first such directive. For example:

```
  .MCALL ALUN$,QIO$
         .
         .
         .
  ALUN$ 2,TT,5
         .
         .
         .
```

Alternatively, you can use the .ENABL MCL assembler directive to enable automatic macro library search. See the *PDP-11 MACRO-11 Language Reference Manual* for details.

Directive Processing

At assembly time, MAC expands the RSX directives to a code that defines the action to be taken and the parameters to be used in a Directive Parameter Block (DPB).

Figure 4-1 shows the general form of a DPB.

```
+----------------------------+-----------------------------+
|   DPB length (in words)    |   Directive ID Code (DIC)   |
+----------------------------+-----------------------------+
|                                                          |
|                       (Parameters)                       |
|                            .                             |
|                            .                             |
|                            .                             |
+----------------------------------------------------------+
```

Figure 4-1: General Form of the Directive Parameter Block

The Directive Identification Code (DIC) in the low byte of the first word of the DPB defines the particular function to be performed. The RSX emulator examines the DIC to determine what it is supposed to do. The high byte of the first word of the DPB gives the length of the DPB in words (including the DIC and length byte). The remaining words contain parameters; usually these are values specified by the user in the directive call, expanded through MACRO-11 data definition macros such as .BYTE, .WORD, .RAD50, and so forth.

The directive expansions also include executable code to tell the emulator where the DPB is, an EMT 377 instruction to pass control to the emulator and, optionally, transfer control to an error processing routine specified by the user.

At execution time, RSTS/E pushes either the DPB address or the DPB itself on the stack. The choice depends on the form of the directive. The EMT 377 instruction passes control to the RSTS/E monitor. The monitor recognizes the EMT 377 and passes control to the RSX emulation in the monitor. The RSX emulator examines the DPB and processes the directive. It then pops the DPB address or the entire DPB off the stack and either returns control to the program that executed the directive or, if the directive was an exit of some kind, to whatever location is appropriate to the exit.

When control returns in line, the carry condition code in the program status word (PSW) indicates that the directive executed successfully (carry code = 0) or failed (carry code = 1). The Directive Status Word gives more information on the success or failure of the directive. You can refer to the Directive Status Word in your program with the mnemonic $DSW; TKB defines this variable and resolves its location automatically at link time. Other mnemonics are similarly related to the possible values of $DSW, so the MACRO programmer can test for error conditions using symbols such as IS.SUC for successful completion. See "DSW Return Codes" in Chapter 5 for a listing of the symbols appropriate to each directive.

Directive Forms ($, $C, $S) and Their Expansions

The library file RSXMAC.SML contains three different expansions for each of the RSX-11M directives (see Chapter 5). The form of the directive name you use determines which expansion to be inserted in your code. Directive names consist of one to four letters, followed by a dollar sign ($) and, optionally, a C or an S.

Figure 4-2 shows an example of the three forms for the ALUN$ directive (Assign Logical Unit).

```
Form  Your Code                        Expansion        Description
---------------------------------------------------------------------

$  A: ALUN$ 5,TT,0             A:     .BYTE 7,4        Create DPB in data
      .                               .WORD 5          section of program
      .                               .ASCII /TT/      (at assembly time)
      .                               .WORD 0
      .                               .
                                      .
      DIR$ #A,ERR                     MOV #A,-(SP)     Push DPB address
      .                               EMT 377          Call RSX emulator
      .                               BCC .+6          Branch if no error
      .                               JSR PC,ERR       Go to ERR on error
                                      .
      DIR$ #A,ERR                     .
                                      (same as above) (same as above)

$C    ALUN$C 5,TT,0,PSEC1,ERR         .PSECT $DPB$$    Create DPB at
                               $$$=.                   assembly time and
      .                               .BYTE 7,4        automatically put
      .                               .WORD 5          in separate PSECT
      .                               .ASCII /TT/      called $DPB$$
                                      .WORD 0
                                      .PSECT PSEC1
                                      MOV #$$$,-(SP)   Reenter PSECT
                                      EMT 377          Push DPB address
                                      BCC .+6          Call RSX emulator
                                      JSR PC,ERR       Branch if no error
                                                       Go to ERR on error

$S    ALUN$S #5,#"TT,#0,ERR           MOV #0,-(SP)     Create DPB at
      .                               MOV #"TT,-(SP)   execution time and
      .                               MOV #5,-(SP)     push entire DPB
      .                               MOV (PC)+,-(SP) on stack

                                      .BYTE 7,4
                                      EMT 377          Call RSX emulator
                                      BCC .+6          Branch if no error
                                      JSR PC,ERR       Go to ERR on error
```

Figure 4-2: Example of RSX Directive Forms

$ Form (and DIR$ Directive)

As Figure 4-2 shows, the $ form of an RSX emulator directive generates a DPB inline. The DIR$ directive expands to code that pushes the address of the DPB on the stack and passes control to the RSX emulator. Use the $ form/DIR$ combination when you want to repeatedly execute the same directive. After one directive defines a DPB, as many other DIR$ directives as need it, can use it. This technique saves memory. Also, you can modify the individual parameters in the DPB when you use the same directive many times with varying parameters.

$ Form

The $ form of an RSX emulator directive expands to code that defines the DPB. The expansions consist of MACRO-11 data storage directives (for example, .BYTE, .WORD, .RAD50); they are not executable. Therefore, use the $ form of an RSX emulator directive only in a data section of your program.

Finally, use parameters that fit the MACRO-11 data storage directive in which they will appear in the expanded code. For example, the ALUN$ directive expands to the four instructions that follow:

```
  ALUN$ 5,TT,0
```

Expansion:

```
  .BYTE 7,4
  .WORD 5
  .ASCII /TT/
  .WORD 0
```

Although the example uses the two-character second parameter (TT) in a MACRO-11 .ASCII directive, you do not need to specify the enclosing slashes; the expansion does that for you.

Note that the first word of the expansion (the first word of the DPB) consists of two bytes. The 7 in the low byte is the DIC byte for the ALUN$ directive. The 4 in the high byte indicates that the DPB is 4 words long.

DIR$ Directive

The general form of the DIR$ directive is:

DIR$ [*adr*] [,*err*]

where:

adr is the address of the first word of the DPB. As shown in the following expansion, this address must be a valid source address for a MOV instruction. (The *adr* parameter can be omitted. In this case, the entire DPB must be on the stack when the DIR$ is executed.)

err is an optional error address. Control passes to this address if the directive does not execute successfully. If this parameter is omitted, control returns in line with the C-bit set in the program status word (PSW).

For example, the DIR$ directive expands to the four instructions that follow:

```
  DIR$ #DPB,ERROR
```

Expansion:

```
  MOV    #DPB,-(SP)
  EMT    377
  BCC    .+6
  JSR    PC,ERROR
```

RSTS/E pushes the address of the DPB on the stack, and the EMT transfers control to the RSX emulator. The emulator processes the directive according to the parameters specified in the DPB, pops the DPB address off the stack, and (for ALUN$) returns control in line. The BCC instruction branches around the JSR to ERROR if the C-bit is not set (no error occurred).

Other Features of $ Form/DIR$ Combination

The $ form of the RSX emulator directives generates local offsets that you can use to refer to parameters in the DPB.

Note

> The expansions use the MACRO-11 .NLIST directive so the code that generates these offsets (which is not of interest to your program) does not show up in your assembly listings.

For example, in the ALUN$ directive, you specify a logical unit number, physical device name, and device unit number:

```
  LN5DEF:  ALUN$ 5,TT,0
```

When this directive is expanded, it includes code that allows you to refer to the parameters as:

```
  LN5DEF+A.LULU     (Logical unit number)
  LN5DEF+A.LUNA     (Device name)
  LN5DEF+A.LUNU     (Unit number)
```

Chapter 5 lists these offset names under "Local Symbol Definitions" for each directive. The number of bytes defined for each offset is given in parentheses following a description of the area referenced by the offset. For example:

```
  A.LUNU -- Logical unit number (2)
```

(You can also use offset names with the $C form of directives. However, you cannot use offset names with the $S form because $S generates values that are pushed on the stack.)

The $ form expansions are also designed so you can use them to simply set up the offset symbols to reference a DPB in another module. If you define the symbol $$$GLB in your program (for example, $$$GLB = 0 will do), the $ form of any directive generates the symbolic offsets as global symbols and does not generate the DPB itself. For example, suppose another module has the following code:

```
  MYMY::  ALUN$ 5,TT,0
```

In the current module, define MYMY as a global symbol, define $$$GLB, specify an ALUN$, and you will be able to reference MYMY+A.LULU, and so forth.

$C Form

If you know the DPB parameters at assembly time and need to issue a directive only once, the $C form saves you the trouble of coding two directives: the $ form and DIR$. The $C form also saves execution time over the $S form, which uses MOV instructions to place the entire DPB on the stack and may save space as well.

The $C form of the directives expands to code that:

- o Defines the DPB in a separate PSECT (called $DPB$$).
- o Returns to a user-specified PSECT. If no PSECT name is given, returns to the blank (unnamed) PSECT.
- o Pushes the DPB address on the stack.
- o Passes control to the RSX emulator.
- o On return, passes control to a user-specified error routine if the directive did not execute successfully. If you did not specify an error routine, no such code is generated.

The directive descriptions in Chapter 5 show the form and expansion for the $ form. To use the $C form, you must add the C to the directive name and, optionally, add a PSECT name and error address to the parameter list. For example, the ALUN$C directive expands to the 11 instructions that follow:

```
  ALUN$C 5,TT,0,PART1,ERROR
```

Expansion:

```
.PSECT   $DPB$$
$$$=.
.BYTE    7,4
.WORD    5
.ASCII   /TT/
.WORD    0
.PSECT   PART1
MOV      #$$$,-(SP)
EMT      377
BCC      .+6
JSR      PC,ERROR
```

MAC places the DPB in a separate PSECT called $DPB$$. The $$$ symbol equates to the address of the DPB, which MAC uses in the MOV instruction that pushes the DPB address on the stack before control passes to the emulator by the EMT instruction. The last two instructions set up the test for error and transfer control to the specified error routine. If you do not specify an error address in the call, MAC does not generate those instructions.

Other Features of $C Form

The offset mnemonics that MAC generates for the $ form are also generated for the $C form. Thus, you can save space by using the $$$ address to form the base address for the offsets. The $$$GLB feature for the $ form also works for the $C form of the directives.

$S Form

The $S form is useful if your code is reentrant and the DPB parameters vary. RSTS/E generates the DPB when it executes the directive at run time.

The $S form of the directives expands to code that:

- Generates the DPB at execution time and pushes it on the stack.

- Transfers control to the RSX emulator. The RSX emulator checks the low byte of the first word on the stack to determine if it is a DPB address (even low byte) or a DIC (odd low byte). If the low byte is odd, then the emulator knows that the rest of the DPB is on the stack.

- On return, passes control to a user-specified error routine if the directive did not execute successfully. If no error routine is specified, no such code is generated.

To use the $S form of the directives, add S to the directive names shown in Chapter 5. You can also specify an error address as the last parameter in the line. For example, the ALUN$S directive expands to the eight instructions that follow:

```
  ALUN$S #5,DEV,UN,ERR
```

Expansion:

```
  MOV     UN,-(SP)
  MOV     DEV,-(SP)
  MOV     #5,-(SP)
  MOV     (PC)+,-(SP)
  .BYTE   7,4
  EMT     377
  BCC     +.6
  JSR     PC,ERR
```

RSTS/E generates the DPB at run time and pushes it on the stack. Control passes to the RSX emulator, and if control returns with an error (the C-bit is set on return), control transfers to the error routine. If you do not specify an error address in the call, MAC does not generate the last two instructions.

First 512. Bytes of Low Segment for RSX

As Chapter 2 notes, the first 512. bytes of virtual address space have special meaning to the monitor. TKB, which links programs assembled or compiled under the RSX run-time system and its derivatives, generates a header for your executable program. When the RSX emulator loads your program, it uses information from this header to load certain values into the low 512. bytes of your program.

Table 4-1 shows the allocation of the first 512. bytes of the low segment for RSX-type programs. Note that all locations are subject to change except for $DSW, KEY, FIRQB, XRB, and CORCMN.

Table 4-1: First 512. Bytes of Low Segment for RSX

Byte	Mnemonic	Description
0	R.TSKN	Task Name in RAD50
2		.
4	R.PRTN	Partition Name in RAD50
6		.
10	R.CDSZ	Task Size (without extension, 32-word blocks
12	R.ODTV	ODT SST Vector Address
14	R.ODTL	| ODT SST Vector Length
16	R.SSTV	Task SST Vector Address
20	R.SSTL	| Task SST Vector Length
22	R.FAST	FFP AST Service Address
24	R.CAST	CTRL/C AST Service Address
26	R.TSKF	Task Flags
30	R.PARM	Run Parameter on Entry (from FIRQB+FQNENT)
32	R.CCLF	CCL Entry Flags, Run Size/Extension (XRB+0)
34	R.LDSZ	Load Size of Task in 32-word blocks
36	R.TKSZ	Current Size of Task in 32-word blocks
40	R.LUN	Number of LUNs
42	R.MON1	Monitor RSX Emulation Reserved Word 1
44	R.MON2	Monitor RSX Emulation Reserved Word 2
46	$DSW	Directive Status Word
50	.FSRPT	FCS Impure Area Pointer
52	$OTSV	OTS Impure Area Pointer
54	N.OVPT	Auto-Load Impure Area Pointer
56	$VEXT	Extended Impure Area Pointer

Table 4-1: First 512. Bytes of Low Segment for RSX (Cont.)

Byte	Mnemonic	Description
60	CONTXT	(Reserved for Monitor Context Use)
		.
112	FPPTXT	(Reserved for Monitor FPP Context Use)
		.
200	R.LUNS	LUN Table
		.
400	KEY	Keyword
402	FIRQB	FIRQB
		.
442	XRB	XRB
		.
460	CORCMN	Core common area
		.
660	PDMSP	SP saved here on a PM dump
662	PMDXRB	XRB saved here on a PM dump
		FIRQB also saved on PM dump starting at PMDXRB
		.
722		SP stack space for RSX Emulator
		.
734	USRPPN	User's assignable project, programmer number
736	USRPRT	User's assignable protection code
740	USRLOG	User's logical device table
		.
1000	NSTORG	Stack guard word (must = 0)

The RSX emulator directives:

- o Perform non-file-structured I/O
- o Let you specify your own exception-handler routines
- o Control execution of the program
- o Return system information to the job
- o Provide access to resident libraries

Table 5-1 lists these directives by function. The remaining sections of this chapter present the directives in alphabetical order.

Table 5-1: Summary of the RSX Directives

Name	Description
	Non-File-Structured I/O
ALUN$	Assigns a logical unit number to a device and unit. The logical unit number then serves to define the device in I/O requests.
GLUN$	Returns information about a logical unit.
GMCR$	Returns the command line typed by the user to invoke the program currently running, if it was invoked by a RSTS/E CCL command.
QIO$ QIOW$	Open, read from, write to, and close logical units. I/O in RSX-emulation mode is synchronous. That is, it is not asynchronous as in some systems, such as RSX-11M where a user program can request I/O, do other processing, and get an interrupt when the I/O is complete. Thus, under the RSX emulator, QIO$ (queue input/output) and QIOW$ (queue input/output and wait) do the same thing. Control returns to the calling program when the I/O is done.
WSIG$	Provides compatibility with RSX-11M, where some recoverable error conditions require waiting for some significant event to occur before retrying the operation that caused the error. In RSTS/E, such events are transparent to the user program. WSIG$ simply causes the program to sleep for one second.

Table 5-1: Summary of the RSX Directives (Cont.)

Name	Description
Non-File-Structured I/O (Cont.)	
WTSE$	Provides compatibility with RSX-11M, where RSX uses QIO$ and WTSE$ together to cause the program to wait until I/O is completed. Since I/O in RSX-emulation mode is synchronous, the WTSE$ is implemented here as a NOP instruction.
Exception Handling	
ASTX$	Exits from an FPP exception-handling routine, restoring the DSW, PC, and PSW to the values they had before the exception occurred.
SCCA$	Defines an address to which control passes when a user at the job's terminal types a CTRL/C combination. The SCCA$ directive is unique to RSTS/E; it does not exist in the RSX-11M operating system.
SFPA$	Specifies an address to which control is to pass when an FPP exception occurs. (The FPP is available on all PDP-11 processors that can run RSTS/E, except the PDP-11/35 and 40.)
SVDB$	Specifies a table of synchronous exception addresses that are to override any specified with an SVTK$ directive. This is a useful capability for a debugging routine, to divert breakpoint traps, for example, to the debugging routine regardless of any SVTK$ directive in other modules of the job.
SVTK$	Specifies a table of addresses to which control is to pass when synchronous exceptions occur. The job can handle eight possible exceptions including PDP-11/35 and 40 FIS errors, TRAP instructions, BPT instructions, IOT instructions, and T-bit traps.
Program Execution Control	
ABRT$	Sets the job's exit status to "Severe Error" and returns control to the job's keyboard monitor.
EXIT$	Returns control to the job's keyboard monitor (a normal exit).

Table 5-1: Summary of the RSX Directives (Cont.)

Name	Description
Program Execution Control (Cont.)	
EXST$	Sets the job's exit status to the specified value and returns control to the job's keyboard monitor.
EXTK$	Increases or decreases the amount of memory allocated for execution of the user job image.
SPND$	Suspends execution of the program until the user types a delimiter on the keyboard (KB:).
System Access Information	
GPRT$	Returns partition parameters, which contain general information about the job area and user job image area.
GTIM$	Returns the current day and time.
GTSK$	Returns task parameters: run priority, PPN, number of logical units assigned, address of SST vector table (set up the SVTK$ or SVDB$), and the system in which the job is running.
Resident Libraries Access	
ATRG$	Attaches the job to a resident library, laying the groundwork for access to the library. If necessary, the resident library is loaded from disk.
CRAW$	Creates a window of virtual addresses for reference to a library. Optionally, you can map the window to actual locations in a resident library.
CRRG$	Creates a dynamic region and optionally attaches to it.
DTRG$	Detaches the job from a resident library.
ELAW$	Eliminates an address window; it releases the virtual addresses for other use, such as expanding the job image or creating another window.

Table 5-1: Summary of the RSX Directives (Cont.)

Name	Description
	Resident Libraries Access (Cont.)
MAP$	Relates a created address window to actual memory locations in an attached resident library. The user program can then refer to locations in the library using the virtual addresses defined in the window.
RDBBK$	Defines the offsets shown for the RDB areas for the ATRG$ and DTRG$ directives and, in addition, generates code to allocate space for the RDB and fills it with values you specify in the call.
RDBDF$	Assigns literal values to the offsets and status bit mnemonics shown for the RDB areas for the ATRG$ and DTRG$ directives.
UMAP$	Unmaps a window from a library; you can then map the window to another portion of the library or to a different library.
WDBBK$	Defines the offsets shown for the WDB areas for the CRAW$, ELAW$, MAP$, and UMAP$ directives and, in addition, generates code to allocate space for the WDB and fills it with values you specify in the call.
WDBDF$	Assigns literal values to the offsets and status bit mnemonics shown for the WDB areas for the CRAW$, ELAW$, MAP$, and UMAP$ directives.

ABRT$ - Abort

The ABRT$ directive terminates execution of the calling program. Any open files or devices are reset (that is, closed without the usual clean-up operations). On a RSTS/E system, ABRT$ acts the same as the EXST$ directive with a severe error.

Control passes to the job keyboard monitor at the keyboard monitor entry point P.NEW (see Chapter 2). The job keyboard monitor is the default keyboard monitor unless the job has issued the SET JOB/KEYBOARD_MONITOR command or executed the .RTS directive to establish a specific job keyboard monitor (see Chapter 3).

Privileges Required

None

Macro Call

```
   ABRT$ name
```

RSTS/E ignores the name parameter; the calling program can abort only itself.

Macro Expansion

```
ABRT$     ALPHA
.BYTE     83.,3           ;DIC=83., DPB SIZE = 3 WORDS
.RAD50    /ALPHA/         ;IGNORED IN RSTS/E
```

Local Symbol Definitions

A.BTTN	name (4)

DSW Return Codes

IE.SDP	DIC or DPB size is invalid. This does not occur unless your program changes the value of the first word of the DPB during execution. Control does not otherwise return inline for the ABRT$ directive.

ALUN$ — Assign Logical Unit Number

The ALUN$ directive assigns a logical unit number (LUN) to a physical device unit. You can then use the logical unit number in QIO$ or QIOW$ directives to do I/O (open, read, write, or close the associated device).

You can specify the device name as any 2-character logical name. The RSX emulator checks the name specified in the call against the user logical names; if a match is found, the monitor assigns the logical unit number to the corresponding physical device. If the device name specified does not match any user logical device names, the monitor performs a similar search for a match with a system-wide logical device name. If there is no match, the emulator checks the monitor's physical device table, and, if a match is found, assigns the logical unit number specified in the call to the actual physical device unit specified. Otherwise, the directive returns an error.

Privileges Required

If the device is restricted, DEVICE is required.

Macro Call

ALUN$ *lun,dev,unt*

where:

lun	is the logical unit number; 1-14.
dev	is the device name; two ASCII characters.
unt	is the device unit number. Minus one indicates no unit number, as opposed to a unit number of zero. RSTS/E makes a distinction between SY (meaning the public disk structure) and SY0 (meaning the disk from which the system was booted). Likewise, TI, TT, and KB all refer to the job's terminal, whereas KB0 and TT0 both refer to the system console terminal.

Macro Expansion

```
ALUN$     5,TT,0
.BYTE     7,4                 ;ALUN$ MACRO DIC=7, DPB SIZE=4 WORDS
.WORD     5                   ;LOGICAL UNIT NUMBER 5
.ASCII    /TT/                ;DEVICE NAME IS TT (TERMINAL)
.WORD     0                   ;DEVICE UNIT NUMBER=0
```

Local Symbol Definitions

A.LULU	Logical unit number (2)
A.LUNA	Physical device name (2)
A.LUNU	Physical device unit number (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.IDU	Invalid device and/or unit.
IE.ILU	Invalid logical unit number.
IE.SDP	DIC or DPB size is invalid.

ASTX$ -- Asynchronous Exception Exit

In the RSTS/E environment, you can use the ASTX$ directive to terminate a routine that handles an asynchronous FPP exception, only if the program has indicated its intention to handle such traps with an SFPA$ directive. No other condition in RSTS/E causes an asynchronous exception from which an ASTX$ exit is useful. The CTRL/C exception requires only an RTI instruction to exit properly (see SCCA$S).

When you execute an ASTX$, the stack must be in the state:

```
SP--->  (DSW) at the time exception occurred
        (PC) at the time exception occurred
        (PS) at the time exception occurred
        word to which SP pointed before the exception
```

Note

> The required stack contents is subject to change in future releases of RSTS/E.

Your routine must pop the Floating-Point Exception Code (FEC) and Floating-Point Address (FEA) pushed on the stack when the exception occurred (see SFPA$).

When the ASTX$ directive executes, the RSX emulator restores the DSW value to the DSW location and loads the PC and PS registers with their appropriate values. Thus, these values pop from the stack, and control returns to the user program at the point where it left off when the exception occurred.

You can change the contents of the PC address in the stack before you execute ASTX$ so RSTS/E passes control to some point other than where it left off. Use caution because it is hard to debug errors in an asynchronous exception routine.

Privileges Required

None

Macro Call

ASTX$ [*err*]

where:

err	is the error routine address

Macro Expansion

```
ASTX$S    ERR
MOV       (PC)+,-(SP)     ;PUSH DPB ONTO THE STACK
.BYTE     115.,1          ;DIC = 115., DPB SIZE = 1 WORD
EMT       377             ;TRAP TO RSX EMULATOR
JSR       PC,ERR          ;JUMP TO ERR IF DIRECTIVE UNSUCCESSFUL
```

Note

DIGITAL recommends the $S form for this directive for two reasons: first, only a one-word DPB is generated; second, no BCC instruction is generated for the $S form (or the $C form). Thus, the $S form takes less space than the $form/DIR$ combination. In addition, since the one-word DPB requires only one MOV instruction, the $S form takes less space and no more execution time than the $C form.

Local Symbol Definitions

None

DSW Return Codes

IE.SDP	DIC or DPB size invalid. This occurs only if your program changes the first word of the DPB at execution time.

ATRG$ -- Attach Resident Library

The ATRG$ directive attaches the job to a resident library. The type of access (read-only or read/write) is specified in the call. If the calling job can access the library in that fashion, the monitor sets up its own internal tables that lay the groundwork for the job to map windows to the library. Note, however, that the resident library does not take up space in the job area (virtual memory) with an attach. Virtual memory is assigned when a window is created (CRAW$).

Note

> The job's ability to access the resident library depends upon the protection assigned by the system manager when the resident library was installed. The default protection grants read access to all users and denies write access to all users.

If the attach is successful, a resident library ID is returned to the job. Other directives use this ID to detach the job from the library (DTRG$) or to map and unmap windows to and from the library (MAP$ and UMAP$). Once the monitor attaches the job to a library, you can use the ATRG$ directive to determine the library ID.

You can attach an unlimited number of resident libraries to a job at any given time, provided the system has enough resources to keep track of the list of attached libraries.

Privileges Required

None

Macro Call

ATRG$ *adr*

where:

adr specifies the address of an 8-word area defining the library to be attached and the type of access desired. The ATRG$ directive returns information to this area. Two supplementary directives are available to define (RDBDF$) or define and fill (RDBBK$) such an 8-word area, called the resident library definition block (RDB).

Macro Expansion

```
ATRG$    RDBLK
.BYTE    57.,2        ;DIC = 57., DPB SIZE = 2 WORDS
.WORD    RDBLK        ;ADDRESS OF RDB
```

Data Passed in RDB

```
Mne-  Octal                                           Octal  Mne-
monic Offset                                          Offset  monic
           +--------------------+--------------------+
         1 |//////////////////////////////////////////|  0
           +--------------------+--------------------+
         3 |//////////////////////////////////////////|  2
           +--------------------+--------------------+
         5 |          resident library name           |  4   R.GNAM
           |        (2 words in RAD50 format)         |
         7 |                                          |  6
           +--------------------+--------------------+
        11 |//////////////////////////////////////////| 10
           +--------------------+--------------------+
        13 |//////////////////////////////////////////| 12
           +--------------------+--------------------+
        15 |               access flags               | 14   R.GSTS
           +--------------------+--------------------+
        17 |//////////////////////////////////////////| 16
           +--------------------+--------------------+
```

adr+R.GNAM	is the name of the resident library to which the job is to be attached, as two words of RAD50 data. (The system manager makes resident libraries known to the RSTS/E monitor with the INSTALL/LIBRARY command (see the *RSTS/E System Manager's Guide*). With this command, the system manager defines a file (filename.LIB) as a resident library. The monitor regards "filename" as the resident library's name.)
adr+R.GSTS	is the low-order two bits in this word which defines the desired access to the library.

RDBBK$,
RDBDF$

Mnemonic	**Bit**	**Meaning**
RS.RED	0 = 1	Read-only access is desired
RS.WRT	1 = 1	Read/write access is desired

Once RSTS/E attaches a job to a resident library, the access cannot be changed without detaching (DTRG$) and reattaching. Also, an ATRG$ with the same resident library name specified at *adr*+R.GNAM returns the assigned resident library ID.

Data Returned to RDB

```
Mne-  Octal                                           Octal   Mne-
monic Offset                                           Offset  monic
          +--------------------+--------------------+
        1 |            resident library ID          |  0   R.GID
          +--------------------+--------------------+
        3 |   size of library, in 32-word blocks    |  2   R.GSIZ
          +--------------------+--------------------+
        5 |/////////////////////////////////////////|  4
          +--------------------+--------------------+
        7 |/////////////////////////////////////////|  6
          +--------------------+--------------------+
       11 |/////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |/////////////////////////////////////////| 12
          +--------------------+--------------------+
       15 |/////////////////////////////////////////| 14
          +--------------------+--------------------+
       17 |/////////////////////////////////////////| 16
          +--------------------+--------------------+
```

adr+R.GID	is an identifier which must be used in subsequent calls to identify a resident library, rather than the resident library name. Thus, you use this identifier to detach (DTRG$) and to map and unmap windows (MAP$ and UMAP$) to the library.
adr+R.GSIZ	is the size of the resident library, in 32-word blocks.

Local Symbol Definition

A.TRBA Resident library definition block address (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.UPN	Attaching a job to a resident library requires a small buffer; no small buffers are currently available.
IE.PRI	The attach did not succeed because the caller's privileges do not allow the access requested. This could happen either because the access code specified is not compatible with the possible access defined when the library was installed by the system manager or because the protection code associated with the resident library file excludes the access requested by the user.
IE.PNS	The resident library name specified is not known to the monitor. The system manager must install a resident library before the system can use it.

CRAW$ -- Create Address Window

You can use the CRAW$ directive either to create a window (a range of virtual addresses) or to create a window and map it to a range of actual addresses in an attached resident library. You define the range of addresses by naming a base APR, which defines the starting virtual address for the window, and by specifying the size of the window in 32-word blocks. Thus a window always begins on a 4K-word boundary in virtual memory. It may take more than 4K words, depending on the size of the window.

If the virtual address range overlaps the user job image, the call fails with an error. If the address range overlaps an existing window, the system eliminates the previously created window.

The difference between creating a window and creating and mapping a window is best illustrated by example. By using create without map, you can define one window, which can be mapped to a library or portion of a library and then remapped using MAP$ to another portion of the same library or another library, as many times as desired. For example, suppose your program takes up 24K words, and you want to access a 24K-word resident library of data values. You can use create without map to set up a 4K-word window in APR 6. You can then map the window to the first 4K words of the library, process the data, map to the next 4K words of the library, and so forth.

If, on the other hand, you had a 4K-word program and still wanted to access a 24K-word library, you could use CRAW$ to create a 24K-word window and map it to the entire library in APRs 1-6.

A job can create a maximum of seven windows. A window takes at least one APR, possibly more, depending on the size you specify for the window. Thus, the maximum of seven assumes seven 4K windows in APRs 1 through 7. You can never use APR 0 to create a window, since the user program takes at least this much space. As mentioned previously, a window cannot overlap the user job image; thus, the size of the user job image determines the lowest base APR that the job can use. If the program (user job image) is less than 4K words, you can use APRs 1 and up (to the limit imposed by the run-time system boundary) to create windows. If the user job image is between 4K words and 8K words, you can use APRs 2 and up to create windows, and so forth.

If you create a window that overlaps an already existing window, RSTS/E eliminates the old window. For example, if you create a 6K-word window using a base APR of 5, the window uses APRs 5 and 6. If you then create a 4K-word window using a base APR of 6, RSTS/E eliminates the entire old window. APR 5 is free for other use; RSTS/E uses APR 6 for the new window.

Privileges Required

None

Macro Call

CRAW$ *adr*

where:

adr is the address of an 8-word area defining the window and the resident library to which the window is to be mapped, if mapping is requested. The CRAW$ directive also returns information to this area. Two supplementary directives are available to define (WDBDF$) or define and fill (WDBBK$) such an area, called the window definition block (WDB).

Macro Expansion

```
CRAW$      WDB
.BYTE      117.,2            ;DIC = 117., DPB SIZE = 2 WORDS
.WORD      WDB               ;ADDRESS OF WDB
```

Data Passed in WDB

```
Mne-  Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
          +---------------------+---------------------+
        1 |    base APR (1-7)   |/////////////////////|  0   W.NAPR
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |    size, in 32-word blocks, of window     |  4   W.NSIZ
          +---------------------+---------------------+
        7 |       resident library ID (for map)       |  6   W.NRID
          +---------------------+---------------------+
       11 |    offset, in 32-word blocks (for map)    | 10   W.NOFF
          +---------------------+---------------------+
       13 |      map length, in 32-word blocks        | 12   W.NLEN
          +---------------------+---------------------+
       15 |               access flags                | 14   W.NSTS
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
```

adr+W.NAPR is the base APR of the window, 1-7 that implicitly defines the starting address of the window. This byte cannot be zero, nor can it name an APR already being used to map the user job image (see previous discussion).

adr+W.NSIZ Is the desired size of the window, in 32-word blocks. For example, a value of 128 = 4K words.

adr+W.NRID is the identifier of the resident library to which the window is to be mapped. (This is the value returned at *adr*+R.GID in the RDB for an ATRG$ directive.) This word is ignored for calls requesting a create without mapping (bit 7 at *adr*+W.NSTS is zero).

adr+W.NOFF is the offset, in 32-word blocks, from the start of the library where the mapping is to begin. This word is ignored if no mapping is requested (bit 7 at *adr*+W.NSTS is zero.) A value of zero for this word indicates the window is to be mapped beginning at the first byte of the library. A value of one indicates the window is to be mapped beginning at the 33rd word of the library (starting address + 64), and so forth.

adr+W.NLEN is the length, in 32-word blocks, of the area to be mapped (ignored if bit 7 at *adr*+W.NSTS is zero). This value cannot be greater than the length of the window specified at *adr*+W.NSIZ. In addition, this value, combined with the offset specified at *adr*+W.NOFF, cannot indicate an address beyond the end of the library.

A value of zero defaults to either the size of the window or the space remaining in the library, whichever is smaller.

adr+W.NSTS is the access-flags word. Two bits of this word define whether the window is to be mapped and whether read-only access or read/write access to the window is desired.

WDBDF$,
WDBBK$

Mnemonic	**Bit**	**Meaning**
WS.MAP	7 = 1	The window is to be mapped
	= 0	The window is not to be mapped
WS.WRT	1 = 1	Read/write access is desired
	= 0	Read-only access is desired

Data Returned in WDB

```
Mne-  Octal                                          Octal  Mne-
monic Offset                                         Offset monic
         +----------------------+--------------------+
       1 |//////////////////////|     window ID      |  0    W.NID
         +----------------------+--------------------+
       3 |    starting virtual address of window     |  2    W.NBAS
         +----------------------+--------------------+
       5 |///////////////////////////////////////////|  4
         +----------------------+--------------------+
       7 |///////////////////////////////////////////|  6
         +----------------------+--------------------+
      11 |///////////////////////////////////////////| 10
         +----------------------+--------------------+
      13 |    length, in 32-word blocks, mapped      | 12    W.NLEN
         +----------------------+--------------------+
      15 |               status flags                | 14    W.NSTS
         +----------------------+--------------------+
      17 |///////////////////////////////////////////| 16
         +----------------------+--------------------+
```

adr+W.NID	is the window ID, a value from 1-7. You can use this ID in later MAP$ calls to map the newly created window. You must use the ID in any ELAW$ calls to eliminate the window.
adr+W.NBAS	is the starting virtual address of the new window.
adr+W.NLEN	is the length, in 32-word blocks, actually mapped by the window.
adr+W.NSTS	is the status-flags word.

WDBDF$, WDBBK$ Mnemonic	Bit	Meaning
WS.CRW	15 = 1	The window was created successfully
	= 0	The window was not created
WS.ELW	13 = 1	An existing window was eliminated because it overlapped the newly created window
	= 0	No existing windows were eliminated by this create
WS.UNM	14 = 1	An existing window was unmapped because it overlapped the newly created mapping
	= 0	No existing windows were unmapped by this mapping

Local Symbol Definitions

C.RABA	Window definition block address (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.ALG	Either the base APR and window length specified were invalid or the offset and mapping length values specified were invalid. For example, an offset indicating a starting address for the mapping that was beyond the end of the library returns this error.
IE.NVR	The library ID specified for mapping does not specify any library currently attached to the job.
IE.PRI	The create was unsuccessful because the requested access is not allowed. At this point, since the library has been attached successfully with some access defined, this error means that the access requested in the CRAW$ is not allowed by the access requested in the ATRG$.
IE.WOV	An attempt was made to create more than seven address windows.
IE.UPN	Creating a window requires a small buffer; a small buffer is not currently available.

CRRG$ -- Create Dynamic Region

The CRRG$ directive creates dynamic regions similar to those created by UU.RTS. This call operates in a slightly different manner than its counterpart. If an attempt is made to create a region that already exists, CRRG$ returns with bit 15 in R.GSTS clear. If attachment is specified and the region already exists, the new region is attached to the already existing region.

Privileges Required

INSTAL

Macro Call

CRRG$ *adr*

where:

adr is the address of an 8-word Region Descriptor Block (RDB). The CRRG$ directive also returns information to this area.

Macro Expansion

```
CRRG$     RDBLK
.BYTE     55.,2               ;DIC = 55., DPB SIZE = 2 WORDS
.WORD     RDBLK               ;ADDRESS OF RDB
```

Data Passed in RDB

```
Mne-  Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 | size in 32-word blocks of region to create|  2 R.GSIZ
          +---------------------+---------------------+
        5 |      name of region, or 0 for unnamed     |  4 R.GNAM
          +                 (2 words)                 +
        7 |                                           |  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |               region status               | 14 R.GSTS
          +---------------------+---------------------+
       17 |      region protection in RSX format      | 16 R.GPRO
          +---------------------+---------------------+
```

adr+R.GSTS are the status bits that are currently defined as follows:

Bit 5 -- Attach me to region (forced if unnamed region)

Bit 7 -- Delete region when all users detach (forced if unnamed region)

adr+R.GPRO is the region protection code converted from RSX format to RSTS/E format by the following algorithm:

- o The system field is ignored
- o Delete protection is ignored
- o Extend protection is ignored
- o Read protection is mapped to RSTS/E read protection
- o Write protection is mapped to RSTS/E write protection

Data Returned in RDB

```
Mne-  Octal                                         Octal   Mne-
monic Offset                                         Offset  monic
        +--------------------+--------------------+
      1 |     region ID of created region         |  0 R.GID
        +--------------------+--------------------+
      3 |  size of the region in 32-word blocks   |  2 R.GSIZ
        +--------------------+--------------------+
      5 |/////////////////////////////////////////|  4
        +--------------------+--------------------+
      7 |/////////////////////////////////////////|  6
        +--------------------+--------------------+
     11 |/////////////////////////////////////////| 10
        +--------------------+--------------------+
     13 |/////////////////////////////////////////| 12
        +--------------------+--------------------+
     15 |              status flags               | 14  R.GSTS
        +--------------------+--------------------+
     17 |/////////////////////////////////////////| 16
        +--------------------+--------------------+
```

adr+R.GSTS is the status-flags word. Bit 15 is clear if the region already exists.

DSW Return Codes

IS.SUC	Successful completion.
IE.PRI	User does not have INSTAL privilege.
IE.UPN	Creation failed, examine FIRQB for details.

DTRG$ -- Detach Resident Library

The DTRG$ directive detaches the job from a previously attached resident library. Any windows mapped to the library by the calling job are unmapped and eliminated. If the specified library is a dynamic region marked for deletion when all users detach, the monitor checks if the region is no longer attached by any job. If so, the monitor deletes the region and releases the memory allocated to it.

Privileges Required

None

Macro Call

DTRG$ *adr*

where:

adr is the address of an 8-word area defining the library to be detached. The DTRG$ directive also returns information to this area. Two supplementary directives are available to define (RDBDF$) or define and fill (RDBBK$) such an 8-word area, called the Resident-Library Definition Block (RDB).

Macro Expansion

```
DTRG$     RDBLK
.BYTE     59.,2          ;DIC = 59., DPB SIZE = 2 WORDS
.WORD     RDBLK          ;ADDRESS OF RDB
```

Data Passed in RDB

```
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
          +---------------------+---------------------+
        1 |             resident library ID           |  0   R.GID
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
```

adr+R.GID	is the library ID the ATRG$ directive returns (at the same position in the RDB).

Data Returned in RDB

```
Mne- Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +--------------------+--------------------+
        1 |//////////////////////////////////////////|   0
          +--------------------+--------------------+
        3 |//////////////////////////////////////////|   2
          +--------------------+--------------------+
        5 |//////////////////////////////////////////|   4
          +--------------------+--------------------+
        7 |//////////////////////////////////////////|   6
          +--------------------+--------------------+
       11 |//////////////////////////////////////////|  10
          +--------------------+--------------------+
       13 |//////////////////////////////////////////|  12
          +--------------------+--------------------+
       15 |              status flags                |  14  R.GSTS
          +--------------------+--------------------+
       17 |//////////////////////////////////////////|  16
          +--------------------+--------------------+
```

adr+R.GSTS	is the status flag, bit 14, that defines if any windows were unmapped as a result of this detach.

Local Symbol Definitions

D.TRBA	Resident-Library Definition Block address (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.NVR	The library ID specified does not identify any library currently attached to the job.

ELAW$ -- Eliminate Address Window

The ELAW$ directive eliminates an address window that was created by the job, unmapping the window if necessary. ELAW$ frees the APRs used by the window and makes them available for creating another window or for expanding the user job image size.

Privileges Required

None

Macro Call

```
   ELAW$ adr
```

where:

adr is the address of an 8-word area defining the window to be eliminated. The ELAW$ directive also returns information to this. Two supplementary directives are available to define (WDBDF$) or define and fill (WDBBK$) such an area, called the Window Definition Block (WDB).

Macro Expansion

```
ELAW$     WDBADR
.BYTE     119.,2          ;DIC=119., DPB SIZE=2 WORDS
.WORD     WDBADR          ;ADDRESS OF WDB
```

Data Passed in WDB

```
Mne-  Octal                                          Octal   Mne-
monic Offset                                         Offset  monic
           +----------------------+---------------------+
        1  |//////////////////////|      window ID      |  0  W.NID
           +----------------------+---------------------+
        3  |////////////////////////////////////////////|  2
           +----------------------+---------------------+
        5  |////////////////////////////////////////////|  4
           +----------------------+---------------------+
        7  |////////////////////////////////////////////|  6
           +----------------------+---------------------+
       11  |////////////////////////////////////////////| 10
           +----------------------+---------------------+
       13  |////////////////////////////////////////////| 12
           +----------------------+---------------------+
       15  |////////////////////////////////////////////| 14
           +----------------------+---------------------+
       17  |////////////////////////////////////////////| 16
           +----------------------+---------------------+
```

adr+W.NID   is the ID of the window to be eliminated (returned at the same location in the WDB by the CRAW$ directive).

Data Returned in WDB

```
Mne-  Octal                                                Octal   Mne-
monic Offset                                               Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |                status flags               | 14   W.NSTS
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
```

adr+W.NSTS   is the status-flags word. Two bits in this word give the following information:

WDBDF$, WDBBK$ **Mnemonic**	**Bit**	**Meaning**
WS.ELW	13 = 1	The window was successfully eliminated
	= 0	The window was not eliminated
WS.UNM	14 = 1	The window eliminated was mapped to a resident library and has been unmapped
	= 0	The window eliminated was not mapped; no unmapping was done

Local Symbol Definitions

E.LABA Window Definition Block address (2)

DSW Return Codes

IS.SUC Successful completion.

IE.NVW Bad window ID. It is either outside the range 1 to 7 or it does not match any window currently created for this job.

EXIT$ - Task Exit

The EXIT$ directive stops execution of the calling job. Control passes to the job keyboard monitor at the keyboard monitor entry point P.NEW (see Chapter 2). Any devices still open are reset; that is, closed without performing the usual clean-up operations.

Note

> On exiting from the previously running program, the monitor drops any temporary privileges still in effect.

The job keyboard monitor is the default keyboard monitor unless the job has issued the SET JOB/KEYBOARD_MONITOR command or executed the .RTS directive to establish a specific job keyboard monitor (see Chapter 3).

Privileges Required

None

Macro Call

EXIT$S [*err*]

where:

err	is the error routine address.

Macro Expansion

```
EXIT$S    ERR
MOV       (PC)+,-(SP)     ;PUSH DPB ONTO THE STACK
.BYTE     51.,1           ;DIC = 51., DPB SIZE = 1 WORD
EMT       377             ;TRAP TO THE EMULATOR
JSR       PC,ERR          ;CALL ERROR ROUTINE
```

Note

DIGITAL recommends the $S form for this directive for two reasons: first, only a one-word DPB is generated; second, no BCC instruction is generated for the $S form (or the $C form), since control never returns inline unless an error occurs.

Thus, the $S form takes less space than the $ form/DIR$ combination. In addition, because the one-word DPB requires only one MOV instruction, the $S form takes less space and no more execution time than the $C form.

Local Symbol Definitions

None

DSW Return Codes

IE.SDP	DIC or DPB size is invalid. This occurs only if your program changes the first word of the DPB at execution time.

EXST$ -- Exit with Status

Like EXIT$, the EXST$ directive stops execution of the calling job. On exiting from the previously running program, the monitor drops any temporary privileges still in effect. In addition, EXST$ sets the job's exit status.

Control passes to the job keyboard monitor at the keyboard monitor entry point P.NEW (see Chapter 2). Any devices still open are reset; that is, closed without performing the usual clean-up operations. The monitor detaches any attached resident libraries, and eliminates any created address windows.

Note

> The job keyboard monitor is the default keyboard monitor unless the job has issued the SET JOB/KEYBOARD_MONITOR command or executed the .RTS directive to establish a specific job keyboard monitor (see Chapter 3).

Privileges Required

None

Macro Call

```
   EXST$ sts
```

where:

sts	is the status value. DCL copies this data into the $SEVERITY and $STATUS variables (see the *RSTS/E Guide To Writing Command Procedures*).

Macro Expansion

```
EXST$      STATS
.BYTE      29.,2            ;DIC=29., DPB SIZE = 2 WORDS
.WORD      STATS            ;EXIT STATUS WORD
```

Local Symbol Definitions

E.XSTS	Exit status word (2)
EX$WAR	Warning status code
EX$SUC	Success status code
EX$ERR	Error status code
EX$SEV	Severe error status code

DSW Return Codes

No errors are possible with EXST$.

EXTK$ -- Extend Task

The EXTK$ directive lets you increase the amount of memory allocated to the user job image. For compatibility with RSX-11M, the argument specified in the call indicates a positive or negative increment of 32-word blocks. However, the RSTS/E monitor actually allocates space for the user job image in 1K-word units. Thus, the RSX emulator keeps track of the user job image size as a number of 32-word blocks. When an EXTK$ directive is executed, the emulator checks to see if the extension or reduction needs or frees one or more 1K-word blocks of memory. If so, the emulator directs the monitor to make the new allocation; if not, the current allocation remains.

Note

> Do not use the .CORE directive (see Chapter 3) before an EXTK$ directive in the same program. The monitor executes the .CORE directive directly, without the intervention (or knowledge) of the RSX emulator. If a .CORE executes before an EXTK$, the emulator has incorrect information on the current size of the user job image when it executed the EXTK$, and it bases the new allocation on the size before the .CORE was executed.

You cannot extend the size beyond the least of the following:

- The job's private maximum size -- A private job maximum size can be set by the system manager.
- The system-wide maximum user job image size -- Set by the system manager (see SWAP MAX, *RSTS/E System Installation and Update Guide*).
- 32K-32 words -- The maximum size allowed by the RSX emulator.

You also cannot extend the size of the image into virtual address space used by an address window created for use with resident libraries (see CRAW$ directive).

Note

> If I&D space is in use, the size applies only to the D-space of the task. The size of the I-space is fixed at task load time.

If there is not enough contiguous memory for the extension to be made where the job currently is in memory, the monitor swaps the user job image to disk and then returns it to memory at a location that can accomodate the new size.

If no increment is given in the call, the emulator allocates the amount of space the job had initially, based on information built in at link time (with TKB); that is, before any EXTK$ was executed.

Privileges Required

None

Macro Call

```
   EXTK$ [inc]
```

where:

inc	is the number of 32-word blocks by which the task size is to be extended (positive value for *inc*) or reduced (negative value for *inc*).

Macro Expansion

```
EXTK$     40
.BYTE     89.,3          ;DIC = 89., DPB SIZE = 3 WORDS
.WORD     40             ;EXTEND 40(8) BLOCKS (1K WORDS)
.WORD     0              ;RESERVED WORD
```

Local Symbol Definitions

E.XTIN	Extend increment (2)

DSW Return Codes

IE.SUC	Successful completion.
IE.UPN	The requested increment would have caused the user job image size to exceed that allowed (see previous discussion). The user job image size stays at the current size.

GLUN$ - Get LUN Information

The GLUN$ directive fills a 6-word buffer with information about a physical device unit to which a logical unit (LUN) is assigned.

Privileges Required

None

Macro Call

GLUN$ *lun,buf*

where:

lun is the logical unit number.

buf is the address of the 6-word buffer to receive the information.

Buffer Format

```
Mne-  Octal                                              Octal   Mne-
monic Offset                                              Offset  monic
          +--------------------+--------------------+
        1 |     device name (2 ASCII characters)     |  0   G.LUNA
          +--------------------+--------------------+
        3 |            device unit number            |  2   G.LUNU
          +--------------------+--------------------+
        5 |     first device characteristics word    |  4   G.LUCW
          +--------------------+--------------------+
        7 |     second device characteristics word   |  6
          +--------------------+--------------------+
       11 |//////////////////////////////////////////| 10
          +--------------------+--------------------+
       13 |        standard device buffer size       | 12
          +--------------------+--------------------+
```

buf+0 is the name of the assigned device; two ASCII characters. This must be a valid physical device name. Exceptions are the logical device names OV and LB, which default to SY if they are not defined, and CL, which defaults to TT.

buf+2 is the unit number of the assigned device. If this word is minus one, it indicates there is no unit number. If this word is zero or a positive number, it is a real unit number.

buf+4	is the first device characteristics. The bit settings are:

Bit	Meaning When Set
0	Record-oriented device
1	Carriage-control device
2	Terminal device
3	Directory device
4	Single-directory device
5	Sequential device
6-15	Reserved

Settings for bits 0-5 are (in octal):

01 = Card reader
01 = Paper tape reader/punch
02 = Line printer
03 = Terminal
10 = Disk
40 = Magtape

buf+6 is the privilege status. If this word is zero, it indicates the job is nonprivileged. If this word is eight, it indicates the job is privileged.

buf+12 is the standard device buffer size to use for input or output on this device.

Macro Expansion

```
GLUN$     7,LUNBUF
.BYTE     5,3               ;DIC = 5, DPB SIZE = 3 WORDS
.WORD     7                 ;LOGICAL UNIT NUMBER 7
.WORD     LUNBUF            ;ADDRESS of 6-WORD BUFFER
```

Local Symbol Definitions

G.LULU	Logical unit number (2)
G.LUBA	Buffer address (2)

The monitor assigns the following offsets relative to the start of the LUN information buffer:

G.LUNA	Device name (2)
G.LUNU	Device unit number (1)
G.LUFB	Flags byte (not used in RSTS/E) (1)
G.LUCW	Four device characteristics words (8)

DSW Return Codes

IS.SUC	Successful completion.
IE.ULN	Unassigned LUN.
IE.ILU	Invalid logical unit number (must be in range 1-14).
IE.SDP	DIC or DPB size is invalid.

GMCR$ -- Get MCR (CCL) Command Line

The GMCR$ directive transfers an 80-byte CCL command line to the job. The program that processes an installed CCL command must contain code to analyze what you typed to invoke the file. The GMCR$ directive returns the command line you typed in an 80-byte buffer in the DPB.

This directive has no $S form. Because the DPB receives the CCL command line, the space must be allocated in read/write memory.

Privileges Required

None

Macro Call

```
   GMCR$
```

Macro Expansion

```
GMCR$
.BYTE     127.,41.         ;DIC = 127., DPB SIZE = 41. WORDS
.BLKW     40.              ;80-CHARACTER CCL COMMAND BUFFER
```

Local Symbol Definitions

G.MCRB	CCL command buffer (80)

DSW Return Codes

+n	Successful completion. The value n is the number of data bytes transferred (excluding the termination character).
IE.AST	Task was not run as a CCL command or the GMCR$ directive has already been executed.
IE.SDP	DIC or DPB size is invalid.

GPRT$ - Get Partition (Job) Parameters

The GPRT$ directive fills a 3-word buffer with *partition* parameters. For RSTS/E, a partition corresponds to the job area.

Privileges Required

None

Macro Call

GPRT$ [*name*],*buf*

where:

name	is the partition name. Omit this parameter to get the actual job image size in buf+2.
buf	is the address of the 3-word buffer where information is to be returned.

Buffer Format

```
Mne- Octal                                               Octal   Mne-
monic Offset                                             Offset  monic
         +--------------------+--------------------+
       1 | virtual base address (always 0 in RSTS/E) |  0    G.PRPB
         +--------------------+--------------------+
       3 |   user job image size in 32-word blocks   |  2    G.PRPS
         +--------------------+--------------------+
       5 |      flags word (always 0 in RSTS/E)      |  4    G.PRFW
         +--------------------+--------------------+
```

name	is the partition name. Omit name to get the actual job image size in buf+2.
buf+2	is the partition size (user job image size in RSTS/E), expressed as a multiple of 32 words. If I&D space is enabled, this reflects only the size of the D-space of the task. If you give a partition name, the system SWAP MAX is returned, in 32-word units.
buf+4	is the partition flags word. Always returned as zero, since all jobs are system-controlled in RSTS/E. (RSX-11M returns one for a user-controlled partition.)

Macro Expansion

```
GPRT$     ,DATBUF
.BYTE     65.,4           ;DIC=65., DPB SIZE = 4
.WORD     0,0             ;GENERATE 0 WORDS IF NO NAME
.WORD     DATBUF          ;ADDRESS OF 3-WORD BUFFER
```

Local Symbol Definitions

G.PRPN	Partition name (4)
G.PRBA	Buffer address (2)

The monitor assigns the following offsets relative to the start of the buffer:

G.PRPB	Partition virtual base address (2)
G.PRPS	Partition size (2)
G.PRFW	Partition flags word (2)

DSW Return Codes

Successful completion is indicated by carry clear, and the starting address of the partition is returned in the DSW (always zero in RSTS/E). Unsuccessful completion is indicated by carry set and the following code in the DSW:

ID.SDP	DIC or DPB size is invalid.

GTIM$ -- Get Time Parameters

The GTIM$ directive fills an 8-word buffer with the current day and time of day. All time values are stored as one-word binary integers. The value ranges (in decimal) are shown in the following diagram.

Privileges Required

None

Macro Call

GTIM$ *buf*

where:

buf is the address of an 8-word buffer.

Buffer Format

Mne-monic	Octal Offset		Octal Offset	Mne-monic
	1	year (since 1900); such as 78 =1978	0	G.TIYR
	3	month (1-12); such as 1 = January	2	G.TIMO
	5	day of month (1-31)	4	G.TIDA
	7	hour (0-23); such as 0 = midnight	6	G.TIHR
	11	minute (0-59)	10	G.TIMI
	13	second (0-59)	12	G.TISC
	15	tick of second	14	G.TICT
	17	ticks per second	16	G.TICP

G.TIBA+G.TICT Indicates a tick which is either 1/60th or 1/50th of a second, depending on the clock in use and/or the line frequency. Systems running with the KW11P clock at crystal speeds, rather than at line frequency, have a tick of 1/50th of a second. Otherwise, if the system is operating with a 60 Hz power line, a tick is 1/60th of a second.

Macro Expansion

```
GTIM$     DATBUF
.BYTE     61.,2          ;DIC = 61., DPB SIZE = 2 WORDS
.WORD     DATBUF         ;ADDRESS OF 8-WORD BUFFER
```

Local Symbol Definitions

G.TIBA	Buffer address (2)

The monitor assigns the following offsets relative to the start of the buffer:

G.TIYR	Year (2)
G.TIMO	Month (2)
G.TIDA	Day (2)
G.TIHR	Hour (2)
G.TIMI	Minute (2)
G.TISC	Second (2)
G.TICT	Clock tick of second (2)
G.TICP	Clock ticks per second (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.SDP	DIC or DPB size is invalid.

GTSK$ -- Get Task (Job) Parameters

The GTSK$ directive fills a 16-word buffer with parameters for the task. As with partition in GPRT$, a task in the RSTS/E environment is equivalent to the job.

Privileges Required

None

Macro Call

GTSK$ *buf*

where:

buf	is the address of the 16-word buffer to receive the parameters.

Buffer Format

```
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +-------------------+--------------------+
        1 |              task name                 |  0   G.TSTN
          |       (2 words in RAD50 format)        |
        3 |                                        |  2
          +-------------------+--------------------+
        5 |            partition name              |  4   G.TSPN
          |       (2 words in RAD50 format)        |
        7 |                                        |  6
          +-------------------+--------------------+
       11 |////////////////////////////////////////| 10
          +-------------------+--------------------+
       13 |////////////////////////////////////////| 12
          +-------------------+--------------------+
       15 |          run priority (0-255)          | 14   G.TSPR
          +-------------------+--------------------+
G.TSPC 17 |   project number  | programmer number  | 16   G.TSGC
          +-------------------+--------------------+
       21 |       number of LUNs assigned          | 20   G.TSNL
          +-------------------+--------------------+
       23 |////////////////////////////////////////| 22
          +-------------------+--------------------+
       25 |////////////////////////////////////////| 24
          +-------------------+--------------------+
       27 |      address of SST vector table       | 26   G.TSVA
          +-------------------+--------------------+
       31 |  length of SST vector table in words   | 30   G.TSVL
          +-------------------+--------------------+
       33 | size (in bytes) of memory for job image| 32   G.TSTS
          +-------------------+--------------------+
       35 | system where job is running (4 = RSTS/E)| 34  G.TSSY
          +-------------------+--------------------+
```

buf+0	is the task name. The name supplied for the TASK = name option at the time the task was built (linked) with TKB; otherwise, the file name of the file being run.
buf+4	is the partition name specified for the PAR = name option at the time the task was built (linked) with TKB. If no such name was specified, the characters "GEN " (GEN followed by three blanks), to indicate the general partition, the default system-controlled partition in RSX-11M.
buf+14	is the run priority; may range from 0 to 255. This is the RSTS/E job priority plus 128.
buf+16	is the programmer number of the user currently running the program.
buf+18	is the project number of the user currently running the program.
buf+20	is the number of logical unit numbers (LUNs) currently assigned.
buf+26	is the address of the SST vector table. If *buf*+30 = 0, this word is meaningless.
buf+30	is the length of the SST vector table, in words. If there is no SST vector table, this word is set to 0.
buf+32	is the job size, in bytes. If I and D space is enabled, this size reflects only the D portion of the task.
buf+34	is the system in which the program/job/task is running:

0 = RSX-11D
1 = RSX-11M
2 = RSX-11S
3 = IAS
4 = RSTS/E
5 = VAX/VMS
6 = RSX-11M-PLUS
7 = RT-11

This value is always 4 on RSTS/E systems.

Macro Expansion

```
GTSK$      DATBUF
.BYTE      63.,2          ;DIC=63., DPB SIZE = 2 WORDS
.WORD      DATBUF         ;ADDRESS OF 16-WORD BUFFER
```

Local Symbol Definitions

G.TSBA	Buffer Address

The monitor assigns the following offsets relative to the buffer:

G.TSTN	Task name (4)
G.TSPN	Partition name (4)
G.TSPR	Priority (2)
G.TSGC	Programmer number (1)
G.TSPC	Project number (1)
G.TSNL	Number of logical units (2)
G.TSVA	SST vector address (2)
G.TSVL	Length of SST vector table (2)
G.TSTS	Size of space for user job image (2)
G.TSSY	System being run under (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.SDP	DIC or DPB size is invalid.

MAP$ - Map Address Window

The MAP$ directive maps a previously created address window to an attached resident library. In other words, MAP$ relates the virtual address range defined by a CRAW$ directive to actual locations in memory within a resident library that has been attached to the job by an ATRG$ directive.

If the window specified is already mapped, it is unmapped from its previous actual memory locations and remapped to the new area. A job may map a maximum of seven address windows at any given time.

Privileges Required

None

Macro Call

MAP$ *adr*

where:

adr	is the address of an 8-word area defining the window to be mapped. The MAP$ directive also returns information to this area. Two supplementary directives are available to define (WDBDF$) or define and fill (WDBBK$) such an area, called the Window Definition Block (WDB).

Macro Expansion

```
MAP$      WDBADR
.BYTE     121.,2          ;DIC=121., DPB SIZE=2 WORDS
```

Data Passed in WDB

```
Mne-  Octal                                          Octal   Mne-
monic Offset                                          Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////|      window ID      |  0   W.NID
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |           resident library ID             |  6   W.NRID
          +---------------------+---------------------+
       11 |        offset, in 32-word blocks          | 10   W.NOFF
          +---------------------+---------------------+
       13 |        length, in 32-word blocks          | 12   W.NLEN
          +---------------------+---------------------+
       15 |               access flags                | 14   W.NSTS
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
```

adr+W.NID	is the ID of the window to be mapped (returned at the same location in the WDB by the CRAW$ directive).
adr+W.NRID	is the ID of the resident library to which the window is to be mapped (returned at offset R.GID in the RDB by the ATRG$ or CRRG$ directive that attached the job to the resident library).
adr+W.NOFF	is the offset, in 32-word blocks, from the start of the library where the mapping is to begin. A value of zero for this word indicates that the window is to be mapped beginning at the first byte of the library. A value of one indicates that the window is to be mapped beginning at the 33rd word of the library (starting address +64), and so forth. The offset cannot indicate a starting address past the end of a library.
adr+W.NLEN	is the length, in 32-word blocks, to be mapped to the library. This value cannot be greater than the size of the window defined in the CRAW$ directive with which the window was created. In addition, this value, combined with the offset value at *adr*+W.NOFF, cannot indicate an address beyond the end of the library. A value of zero for this word defaults to the size of the window or the space remaining in the library, whichever is smaller.
adr+W.NSTS	is the access flag. When bit 1 (WS.WRT) is set, read/write access is allowed. A separate setting for

access in MAP$ and in ATRG$ lets you attach to a library read/write and map a portion of the library read-only. You cannot, however, attach to a library read-only and then map to the library read/write.

Data Returned in WDB

```
Mne- Octal                                     Octal  Mne-
monic Offset                                   Offset monic
      +--------------------+--------------------+
    1 |////////////////////////////////////////////| 0
      +--------------------+--------------------+
    3 |////////////////////////////////////////////| 2
      +--------------------+--------------------+
    5 |////////////////////////////////////////////| 4
      +--------------------+--------------------+
    7 |////////////////////////////////////////////| 6
      +--------------------+--------------------+
   11 |////////////////////////////////////////////| 10
      +--------------------+--------------------+
   13 |          length actually mapped            | 12  W.NLEN
      +--------------------+--------------------+
   15 |               status flags                 | 14  W.NSTS
      +--------------------+--------------------+
   17 |////////////////////////////////////////////| 16
      +--------------------+--------------------+
```

adr+W.NLEN	is the length, in 32-word blocks, mapped by the call.
adr+W.NSTS	are the status flags. Bit 14 (WS.UNM) indicates, when set, if the window specified was unmapped before the new map.

Local Symbol Definitions

M.APBA	Window Definition Block address (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.ALG	The offset and length specified are inconsistent; either the mapping attempted to go beyond the end of the library, or the length is greater than the created window.
IE.PRI	The mapping did not succeed because user privileges did not allow the access desired.
IE.NVW	Either the resident library ID or the address window ID is incorrect. (That is, the job is not currently attached to the specified resident library, or no address window has been created with the specified window ID.)

QIO$ and QIOW$ - Queue I/O Request (and Wait)

The QIO$ and QIOW$ directives do non-file-structured I/O. In the RSX emulation mode environment, QIO$ and QIOW$ do the same thing. I/O is synchronous in this environment. That is, you cannot request I/O, do other processing, and get an interrupt when the I/O completes. In RSX emulation mode, control returns to the program only when the I/O is complete.

As non-file-structured implies, the QIO$ and QIOW$ directives are useful for I/O on a terminal, card reader, paper tape reader/punch, or line printer.

Privileges Required

See discussion

Macro Call

$$\left\{\begin{matrix}\text{QIO\$} \\ \text{QIOW\$}\end{matrix}\right\} \quad \textit{fnc,lun,[efn],[pri],[isb],[ast][,par]}$$

where:

fnc is the I/O function code:

IO.ATT	Attach. In RSTS/E, opens the device specified by the logical unit number (*lun*).
IO.DET	Detach. In RSTS/E, closes the device specified by the logical unit number (*lun*).
IO.RAL	For terminals only, performs a read in ODT mode (see .TTDDT in Chapter 3).
IO.RLB IO.RVB	Read. IO.RLB and IO.RVB do the same thing in RSTS/E. The actions performed depend on the device (see the section "Format and Description by Function Code").
IO.WAL	For terminals only, performs a write in binary mode (see RECORD 4096% for terminal output, *RSTS/E Programming Manual*).
IO.WLB IO.WVB	Write. IO.WLB and IO.WVB do the same thing in RSTS/E. The actions performed depend on the device (see the section "Format and Description by Function Code").

lun	is the logical unit number; defines the device for which the I/O operation is to be performed. Must have been previously defined with an ALUN$ directive.
efn	is the event flag. (The RSX emulator ignores this parameter.)
pri	is the priority. (The RSX emulator ignores this parameter.)
isb	is the I/O status block. Address of a 2-word buffer to which information is returned about the outcome of the I/O operation. The contents vary according to the function code and device, as the following describes.
ast	is the asynchronous exception address. (The RSX emulator ignores this parameter.)
par	is the parameter list. Of the form <pl,p2[,,,,p3]> for RSTS/E users. Form and meaning varies according to function code and device, as the following describes.

Format and Description by Function Code

$$\left\{\begin{array}{l}\text{QIO\$}\\\text{QIOW\$}\end{array}\right\}\ \text{IO.ATT,}\mathit{lun}\text{[,,,}\mathit{isb}\text{]}$$

Opens the device specified by the logical unit number (*lun*). Status information is returned to the 2-word buffer at address *isb*. If the device is restricted, this function requires DEVICE privilege.

$$\left\{\begin{array}{l}\text{QIO\$}\\\text{QIOW\$}\end{array}\right\}\ \text{IO.DET,}\mathit{lun}\text{[,,,}\mathit{isb}\text{]}$$

Closes the device specified by the logical unit number (*lun*). Status information is returned to the 2-word buffer at address *isb*. This function does not require any privileges.

$$\left\{\begin{array}{l}\text{QIO\$}\\\text{QIOW\$}\end{array}\right\}\left\{\begin{array}{l}\text{IO.RLB}\\\text{IO.RVB}\end{array}\right\}\ \text{,}\mathit{lun}\text{,,,[}\mathit{isb}\text{],,<}\mathit{stadd}\text{,}\mathit{size}\text{[,,,}\mathit{block}\text{]>}$$

For record-oriented devices in the RSX-11M environment (see GLUN$), IO.RLB and IO.RVB return a record (a line on a terminal, card on a card reader, record on a paper tape reader). The data is read into the buffer whose starting address is defined by *stadd* and whose length in bytes is defined by *size*. The amount of data read never exceeds the buffer size. If the buffer size is smaller than the record, the next IO.RLB or IO.RVB for that *lun* reads another buffer-full. The block parameter is omitted for these devices.

For disk, IO.RLB and IO.RVB fill the buffer whose starting address is defined by *stadd* and whose length in bytes is defined by size. The read begins with the device cluster number specified by the block parameter. If *block* is omitted or 0, sequential reads are performed and the next read begins with the device cluster. The buffer size should be a multiple of the device cluster size for the disk (see Appendix B).

Note

> Only jobs with the necessary privileges can do non-file-structured disk I/O. RDNFS privilege is required for read-only and WRTNFS privilege is required for read/write non-file-structured disk I/O.

For all devices, the *isb* parameter is the address of a 2-word buffer to which status information on the read is returned. If *isb* is omitted, no such status is returned.

$$\left\{\begin{matrix}\text{QIO\$}\\ \text{QIOW\$}\end{matrix}\right\}\left\{\begin{matrix}\text{IO.WLB}\\ \text{IO.WVB}\end{matrix}\right\}\ ,\textit{lun},,,[\textit{isb}],,<\textit{stadd},\textit{size}[,\textit{vfc}]>$$

For record-oriented devices in the RSX-11M environment (see GLUN$), IO.WLB and IO.WVB write the contents of the buffer to the device specified by the logical unit number (*lun*). The starting address for the buffer is defined by *stadd*, and the length in bytes is defined by *size*. For record-oriented and carriage-controlled devices, the *vfc* parameter specifies an action to be performed after the buffer contents are written. (The *vfc* parameter is a vertical format control character for terminals and line printers.)

Table 5-2 lists the values for *vfc* and the actions taken. The *vfc* parameter is omitted for non-carriage-controlled devices.

Table 5-2: Vertical Format Control Characters

Octal Value	Character	Meaning
040	blank	Single space -- Output a line feed, print the contents of the buffer, and output a carriage return. Normally, printing immediately follows the previously printed line.
060	0	Double space -- Output two line feeds, print the contents of the buffer, and output a carriage return. Normally, the buffer contents are printed two lines below the previously printed line.
061	1	Page eject -- Output a form feed, print the contents of the buffer, and output a carriage return.
053	+	Overprint -- Print the contents of the buffer and output a carriage return. Normally overprints the previous line.
044	$	Prompting output -- Output a line feed and print the contents of the buffer. This mode of output is intended for use with a terminal on which a prompt message is output and input is then read on the same line.
000	null	Internal vertical format -- Print the buffer contents without adding vertical format control characters. In this mode, more than one line of guaranteed contiguous output can be printed for each I/O request.

For disk, IO.WLB and IO.WVB write the contents of the buffer to disk, beginning at the device cluster number specified by the *block* parameter. If *block* is omitted, sequential writes are performed. The next write (for sequential writes) begins with the next device cluster. The buffer size must be a multiple of the device cluster size.

Note

No job can write to disk in non-file-structured mode while any other user is accessing the disk.

For all devices, the *isb* parameter defines the starting address of a 2-word buffer to which status information on the write is to be returned. If you omit *isb*, no such status information is returned.

Contents of I/O Status Buffer

The general format of the information returned to the 2-word status buffer defined by *isb* is:

```
+--------------------+--------------------+
| (terminal I/O code) |  general I/O code   |
+--------------------+--------------------+
|      number of bytes read or written      |
+--------------------+--------------------+
```

The first word contains information on the success or failure of the I/O operation requested. As with $DSW, you can use mnemonic codes to test this word. The Task Builder resolves the mnemonics automatically.

For all devices and operations except terminal reads, you test the low-order byte of the first word to determine the I/O status code. Successful terminal reads provide additional information on what character terminated the line typed at the terminal. For terminal reads, you test the low-order byte first to see if the read was successful (IS.SUC). If it was, you can then test the entire first word to find out if the line was terminated with a carriage return (IS.CR) or an ESC character (IS.ESC), which is called ALTMODE on some terminals.

The second word of the block contains the number of bytes read or written, if the read or write was successful. No information is returned in this word for IO.ATT and IO.DET operations.

The following section describes the I/O return codes. IS.CR and IS.ESC are full-word values; the others are tested by the low-order byte only.

Note

> The RSX emulator does the requested I/O operation using the non-file-structured OPNFQ subfunction of CALFIP, the CLSFQ subfunction of CALFIP, .READ, and .WRITE. It does not clear the FIRQB or XRB when it returns control to the user program. The codes given in square brackets [] are the RSTS/E errors that are returned to the emulator. You can do further checking in the case of IE.BAD, for example, by checking byte 0 of the FIRQB.

Return Codes

IS.SUC	Successful completion of I/O operation. For terminal reads, you can determine the terminating character of the line by testing the full first word of the I/O status buffer for:
	IS.CR The terminal line was terminated with a carriage return/line feed combination. (The terminating characters are not included in the character count in the second word. They do appear in the input buffer.)
	IS.ESC The terminal line was terminated with an ESC (called ALTMODE on some terminals). (The terminating character is not included in the character count in the second word. It does appear in the input buffer.)
IE.BAD	This code is returned by the RSX emulator when an error occurs on the I/O operation that did not fit into any of the error categories diagnosed. To determine the RSTS/E error, examine byte 0 of the FIRQB.
IE.DAA	An open (IO.ATT) was attempted for a device that this job has already opened. The device must be closed (IO.DET) before it can be opened again. [NOTCLS]
IE.DNR	Some hard device I/O error occurred (a read for an off-line device, a write for a physically write-locked device, and so forth). [HNGDEV]
IE.EOF	An end-of-file (EOF) mark, record, or control character (CTRL/Z for terminals) was recognized on a read operation. [EOF]
IE.EOT	No more room is available on the device for a write operation (end of paper tape, end of magtape, not enough disk space available, and so forth). [NOROOM]
IE.PRI	Privilege violation. A job with insufficient privileges attempted to read or write to disk while the disk was in use. [PRVIOL]
IE.NLN	A close (IO.DET) was attempted for a device that was not open. [NOTOPN]
IE.RSU	Shareable resource in use. The device is not available; it is assigned to another user. [NOTAVL]
IE.VER	Unrecoverable error, such as parity error, bad card column, and so forth. [DATERR]

Macro Expansion

```
QIO$      IO.RVB,7,,,IOSTAT,,<IOBUFR,80.>
.BYTE     1,12.            ;DIC=1, DPB SIZE = 12. WORDS
.WORD     IO.RVB           ;FUNCTION CODE FOR READ
.WORD     7                ;LOGICAL UNIT NUMBER 7
.BYTE     0,0              ;EFN,PRI IGNORED IN RSTS/E
.WORD     IOSTAT           ;I/O STATUS BUFFER
.WORD     0                ;AST IGNORED IN RSTS/E
.WORD     IOBUFR           :BUFFER ADDRESS
.WORD     80.              ;BYTE COUNT = 80.
.WORD     0                ;NO VFC NEEDED FOR READ
.WORD     0                ;THIS PARAM. IGNORED IN RSTS/E
.WORD     0                ;THIS PARAM. IGNORED IN RSTS/E
.WORD     0                ;NO BLOCK FOR THIS INVOCATION
```

The expansion for QIOW$ is the same except that the DIC = 3.

Local Symbol Definitions

Q.IOFN	I/O function code (2)
Q.IOLU	Logical unit number (2)
Q.IOEF	Event flag number (1)
Q.IOPR	Priority (1)
Q.IOSB	Address of I/O status block (2)
Q.IOAE	AST address (2)
Q.IOPL	Parameter list (6 words) (12)

DSW Return Codes

IS.SUC	Successful completion.
IE.LUN	Unassigned logical unit number (*lun*). Use ALUN$.
IE.ILU	Invalid logical unit number (*lun*). Must be in the range 0 to 14.
IE.SDP	DIC or DPB size is invalid.

RDBBK$ and RDBDF$ - Define and Fill RDBs

Two directives are available for use with resident library definition blocks (RDBs): RDBDF$ and RDBBK$.

The RDBDF$ directive assigns literal values to the offsets and status bit mnemonics shown for the RDB areas for the ATRG$ and DTRG$ directives. You can use these mnemonics to reference offsets and bit values in an RDB you have allocated space for in your program.

The RDBBK$ directive defines these offsets and, in addition, generates code to allocate space for the RDB and fills it with values you specify in the call.

Privileges Required

[Not applicable]

Macro Call

```
   RDBBK$      size,libnam,,status,prtcde (for read/write access)
or
   RDBBK$      size,libnam,,status,prtcde (for read-only access)
```

where:

size is the size of the dynamic region to create, if defining a RDB to create a dynamic region.

libnam is the name of the resident library to be attached or the dynamic region to be created.

status is the region status to be placed in the RDB. The following are defined status bits:

RS.ATT	When creating a dynamic region, setting this bit indicates that the user wants the region attached on creation. Note that RSTS/E forces this bit to be set when creating an unnamed dynamic region.
RS.NDL	When creating a dynamic region, setting this bit indicates that the region is not automatically removed on the last detach. Note that RSTS/E forces this bit to be set when creating an unnamed dynamic region.
RS.RED	Map with read-only access to library. (for ATRG$)
RS.WRT	Map with read/write access to library. (for ATRG$)

prtcde is the protection code (in RSX format) of the dynamic region to be created.

RDBBK$
RDBDF$

Macro Expansion

```
RDBBK$  SIZE,DATLIB,,<RS.WRT>,167000
.WORD   0
.WORD   SIZE
.RAD50  /DATLIB/
.WORD   0
.WORD   0
.WORD   RS.WRT
.WORD   167000
```

(RS.WRT is assigned to a literal value of two, so bit 1 is set in the seventh word of the RDB, requesting read/write access.)

In addition, all the offsets necessary to reference the data returned in the RDB by ATRG$ or DTRG$ are generated. For example, you can use the mnemonic RS.UNM to test bit 14 of the word at offset R.GSTS in the RDB after execution of a DTRG$ directive. RS.UNM is assigned to a literal value of 40000 (octal) by the RDBBK$ directive.

SCCA$S – Specify Control/C AST

The SCCA$S directive (actually, a macro) lets you specify an address to which control is to be transferred if the user types a CTRL/C while the program is executing. Only the $S form is allowed, because no DPB is ever generated. The SCCA$S directive is unique to the RSTS/E environment; no similar directive exists for the RSX-11M operating system.

The SCCA$S expands to code that takes the address specified and stores it in location 24 (part of the first 512 bytes of memory used by the run-time system; see Chapter 4). The RSX run-time system checks this location when a CTRL/C asynchronous exception occurs, and if it is nonzero, clears the word and transfers control to the location that was specified. If this word is zero, the RSX run-time system falls back to its own processing of CTRL/C exceptions. Thus, SCCA$S is a one-time operation; you must reset the handler as part of the exception-processing routine, if desired.

Your routine can handle the CTRL/C in any way it sees fit. An RTI instruction returns control to the point where it left off at the time of the exception. The stack upon entry to such a routine is:

```
SP--->  (PC) at the time exception occurred
        (PS) at the time exception occurred
        word to which SP pointed before the exception
```

Privileges Required

None

Macro Call

```
   SCCA$S [arg]
```

where:

arg	contains the address (usually specified in the form "#addr") to which control passes if the user enters a CTRL/C combination at the job's terminal. Word 24 is cleared, indicating that the RSX run-time system is to handle such traps.

Macro Expansion

```
SCCA$S     #CTRAP
MOV        #CTRAP,@#24
```

If you omit the address, the expansion is:

```
SCCA$S
CLR        @#24
```

Local Symbol Definitions

None

DSW Return Codes

None. (SCCA$S does not execute an EMT; control does not pass to the RSX emulator when SCCA$S is executed.)

SFPA$ - Specify Floating-Point-Processor Exception Address

The SFPA$ directive tells the RSX emulator one of two things:

- o The job wants to handle FPP exceptions itself. The RSX emulator passes control to the address specified in the directive when such an asynchronous exception occurs.

  The FPP available on all PDP-11 processors that can run RSTS/E, except the PDP-11/35 and 40. The FPP executes concurrently with the PDP-11 central processor. Thus, an error in the FPP unit occurs asynchronously with the execution of the PDP-11 CPU.

- o The job does not want to handle FPP exceptions. The address is omitted in the directive. In this case, if an FPP exception occurs, the RSX emulator aborts the job, and displays the error message ??Reserved instruction trap on the job's terminal (TI:).

If you specify an address in the SFPA$ directive, control passes to the address when an FPP exception occurs. The stack has the following information:

```
SP──>   FEA
        FEC
        (DSW) at the time the exception occurred
        (PC) at the time the exception occurred
        (PS) at the time the exception occurred
        word to which SP pointed before the exception
```

Your routine can process the exception in any way it sees fit. To exit from the exception-handling routine, pop the FEA and FEC from the stack and issue an ASTX$ directive.

Privileges Required

None

Macro Call

SFPA$ [*add*]

where:

add	is the address to which control passes when an FPP exception occurs.

Macro Expansion

```
SFPA$     FLTAST
.BYTE     111.,2          ;DIC = 111., DPB SIZE = 2 WORDS
.WORD     FLTAST          ;ADDRESS OF FLOATING POINT AST
```

Local Symbol Definitions

S.FPAE	Exception handler entry address (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.SDP	DIC or DPB size is invalid.

SPND$S - Suspend

The SPND$S directive instructs the system to suspend the execution of the issuing program. A program can only suspend itself, it cannot suspend any other program. The program is restarted when the user types anything terminated by a RETURN or other delimiter (for example, LINE FEED).

Note

> RSTS/E implements this directive using the .READ directive to the job's terminal (channel 0). Thus, if you are using special terminal modes (binary input, echo control, and so forth), the SPND$S directive clears these modes.

Because this directive requires only a 1-word DPB, DIGITAL recommends the $S form of the macro. It requires less space and runs with the same speed as the DIR$ macro.

Privileges Required

None

Macro Call

```
   SPND$S [err]
```

where:

err is the error routine address.

Macro Expansion

```
SPND$S    ERR
MOV       (PC)+,-(SP)      ;PUSH DPB ONTO THE STACK
.BYTE     45.,1            ;SPND$S MACRO DIC, DPB SIZE=1 WORD
EMT       377              ;TRAP TO THE EMULATOR
BCC       .+6              ;BRANCH IF DIRECTIVE SUCCESSFUL
JSR       PC,ERR           ;OTHERWISE, CALL ROUTINE "ERR"
```

Local Symbol Definitions

None

DSW Return Codes

IS.SPD	Successful completion; that is, the task was suspended.
IE.ADP	Part of the DPB is out of the issuing task's address space.
IE.SDP	DIC or DPB size is invalid.

SVDB$ - Specify SST Vector Table for Debugging Aid

The SVDB$ directive, like the SVTK$ directive, defines a table of addresses to which control is to pass when certain synchronous exceptions occur. When a synchronous exception occurs, the RSX emulator first checks the table defined with SVDB$ to see if an address has been specified for that trap. If so, control is passed to that address. If not, the emulator then checks the table defined with SVTK$ (if any).

In other words, the SVDB$ sets up an SST vector table whose entries, if nonzero, override (but do not destroy) any SST vector table entries defined with SVTK$. The SVDB$ directive is useful within a debugging routine that uses BPT instructions (for example, to set breakpoints). With SVDB$, the debugging routine can divert breakpoint traps to itself, regardless of SVTK$ directives in other modules of the job.

For example, the Octal Debugging Tool (ODT) object module uses the SVDB$ directive to divert exceptions to itself, regardless of what may have been requested with SVTK$ in other modules. This makes it possible to debug without changing your source code; you can link with ODT, use ODT to insert breakpoints, test and debug, and when finished, relink without ODT. Note that it is not necessary to change source code to debug, only to relink: the SVDB$ directive makes this possible.

Note

> The ODT object module (the file ODT.OBJ) can be linked to user modules with the RSTS/E Task Builder (TKB) using the /DA switch, as described in the *RSTS/E Task Builder Reference Manual*. ODT.OBJ is different from ODT.BAC, which is a BASIC-PLUS utility in RSTS/E. The commands and syntax for ODT.OBJ are described in the *RSX/IAS ODT Reference Manual*.
>
> To avoid interfering with ODT, do not use the SVDB$ directive in a user program.

Privileges Required

None

Macro Call

```
   SVDB$ [adr,len]
```

where:

adr is the address of the SST vector table. If *adr* and *len* are both omitted, any SST vector table previously defined with an SVDB$ directive will be deassigned; that is, the emulator will not check for SVDB$ addresses. (A previously executed SVTK$ remains in effect.)

len is the length of the SST vector table, in words. You can specify up to eight addresses. If you want to use only the third, you could save space by using a *len* of three, rather than a *len* of eight and filling the last five words of the buffer with zeros. Specifying a length greater than eight has the same effect as a length of eight.

Vector Table Format

A nonzero value for a word is the address to which control passes when that particular exception occurs. A zero value for a word indicates that you have no interest in handling that exception. Should such a exception occur, the emulator checks the vector table assigned by SVTK$, if any. If there is no SVTK$ vector table, the emulator aborts the job and displays an error message at the job's terminal.

Octal Offset		Octal Offset
1	odd address or nonexistent memory error	0
3	memory protection violation	2
5	T-bit trap or execution of a BPT instr.	4
7	execution of an IOT instruction	6
11	execution of a reserved instruction	10
13	execution of a non-RSX, non-RSTS EMT instr.	12
15	execution of a TRAP instruction	14
17	PDP-11/40 FPP exception	16

Macro Expansion

```
SVDB$     SSTTBL,3
.BYTE     103.,3          ;DIC=103. ,DPB SIZE = 3 WORDS
.WORD     SSTTBL          ;ADDRESS OF SST TABLE
.WORD     3               ;SST TABLE LENGTH = 3 WORDS
```

SVDB$

SVTK$

Octal Offset		Octal Offset
1	odd address or nonexistent memory error	0
3	memory protection violation	2
5	T-bit trap or execution of a BPT instr.	4
7	execution of an IOT instruction	6
11	execution of a reserved instruction	10
13	execution of a non-RSX, non-RSTS EMT instr.	12
15	execution of a TRAP instruction	14
17	PDP-11/40 FPP exception	16

The contents of the program status register and the program counter (PS and PC) at the time of the exception have been pushed on the stack when the exception-handling routine is entered. Certain other values may have been pushed on the stack. The values depend on the exception. If the exception was a memory protect violation, the stack contains:

```
(SP)---> Instruction backup register (SR1)
         Virtual PC of the faulting instruction (SR2)
         Memory protect status register (SR0)
         (PC) at time of exception
         (PS) at time of exception
```

Note

> The RSX emulator returns zero for SR0 and SR1; SR2 contains the PC at the time of the exception.

If the exception was a TRAP instruction or EMT other than 377, the stack contains:

```
(SP)---> Instruction operand (low-order byte) times 2, high-order bits
         cleared
         (PC) at time of exception
         (PS) at time of exception
```

All items except the (PS) and (PC) must be popped from the stack before the exception-handling routine exits. You can then return control to the point where execution left off with an RTI or RTT

Macro Expansion

```
SVTK$     SSTTBL,4
.BYTE     105.,3          ;DIC = 105., DPB LENGTH = 3 WORDS
.WORD     SSTTBL          ;ADDRESS OF SST TABLE
.WORD     4               ;SET TABLE LENGTH = 4 WORDS
```

Local Symbol Definitions

S.VTTA	Table address (2)
S.VTTL	Table length (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.SDP	DIC or DPB size is invalid.

UMAP$ -- Unmap an Address Window

The UMAP$ unmaps a specified address window from a resident library. The unmapping does not eliminate the window (created with a CRAW$ directive) or release the APRs used by the window. The virtual address window can now be remapped to new actual memory locations with the MAP$ directive.

Privileges Required

None

Macro Call

UMAP$ *adr*

where:

adr	is the address of an 8-word area defining the window to be unmapped. The UMAP$ directive also returns information to this area. Two supplementary directives are available to define (WDBDF$) or define and fill (WDBBK$) such an area, called the window definition block, or WDB.

Macro Expansion

```
UMAP$     WDBADR
.BYTE     123.,2          ;DIC=123., DPB SIZE=2 WORDS
.WORD     WDBADR
```

Data Passed in WDB

```
Mne-  Octal                                             Octal   Mne-
monic Offset                                             Offset  monic
          +---------------------+---------------------+
        1 |/////////////////////|      window ID      |  0 W.NID
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |///////////////////////////////////////////| 14
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
```

adr+W.NID is the ID of the window to be unmapped (returned at the same location in the WDB by the CRAW$ directive).

Data Returned in WDB

```
Mne-  Octal                                              Octal   Mne-
monic Offset                                             Offset  monic
          +---------------------+---------------------+
        1 |///////////////////////////////////////////|  0
          +---------------------+---------------------+
        3 |///////////////////////////////////////////|  2
          +---------------------+---------------------+
        5 |///////////////////////////////////////////|  4
          +---------------------+---------------------+
        7 |///////////////////////////////////////////|  6
          +---------------------+---------------------+
       11 |///////////////////////////////////////////| 10
          +---------------------+---------------------+
       13 |///////////////////////////////////////////| 12
          +---------------------+---------------------+
       15 |                status flag                | 14 W.NSTS
          +---------------------+---------------------+
       17 |///////////////////////////////////////////| 16
          +---------------------+---------------------+
```

adr+W.NSTS is the status flag. Bit 14 = 1 (WS.UNM) if the window specified was successfully unmapped.

Local Symbol Definitions

U.MABA	Window definition block address (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.ITS	The window ID specified is either invalid (outside the range 1 to 7) or not currently mapped.

WDBBK$ and WDBDF$ -- Define and Fill WDBs

Two directives are available for use with Window Definition Blocks (WDBs): WDBDF$ and WDBBK$.

The WDBDF$ directive assigns literal values to the offsets and status bit mnemonics shown for the WDB areas for the CRAW$, ELAW$, MAP$, and UMAP$ directives. You can use these mnemonics to reference offsets and bit values in a WDB you have allocated space for in your program.

The WDBBK$ directive defines these offsets and, in addition, generates code to allocate space for the WDB and fills it with values you specify in the call.

Privileges Required

[Not applicable]

Macro Call

WDBBK$ *apr,siz,rid,off,len,<bit1![bit2!bit3]>*

where:

apr	is the base APR.
siz	is the size of the window.
rid	is the resident library ID.
off	is the offset into the library, in 32-word blocks.
len	is the length to be mapped.
bit...	are the mnemonic values for bit settings, separated by exclamation points. The following mnemonics are defined for CRAW$:

WS.MAP	Indicates the window is to be mapped
WS.WRT	Means map with read/write/access
WS.RED	Means map with read-only access

Macro Expansion

```
WDBBK$ 7,128.,0,0,0,<WS.MAP!WS.RED>
.BYTE   0,7
.WORD   0
.WORD   128.
.WORD   0
.WORD   0
.WORD   0
.WORD   WS.MAP!WS.RED
```

When used in a CRAW$ directive, this WDB creates a window 4K words long (128 by 32-word blocks) in APR 7. The call also specifies an offset and map length of zero and read-only mapping. Note that to use this WDB, you have to supply the resident library ID, which you get from the RDB returned by an ATRG$ call. To fill in the library ID, move the word at offset R.GID in the RDB to offset W.NRID in the WDB.

The WDBBK$ directive also defines the offsets and bit settings that the ELAW$, MAP$, and UMAP$ descriptions refer to.

WSIG$ - Wait for Significant Event Flag

The WSIG$ directive is included for compatibility with RSX-11M. In RSX-11M, some recoverable error conditions require waiting for a significant event to happen, at which time a retry of the operation that failed may succeed. On a RSTS/E system, any events that may be termed significant within the RSTS/E monitor are transparent to user jobs. Hence, the WSIG$ directive in RSTS/E merely causes the calling job to sleep for one second. The job can then retry whatever operation failed previously.

Privileges Required

None

Macro Call

```
   WSIG$
```

Macro Expansion

```
WSIG$
.BYTE     49.,1          ;DIC=49., DPB SIZE=1 WORD
```

Local Symbol Definitions

None

DSW Return Codes

IS.SUC	Successful completion.
IE.SDP	DIC or DPB size is invalid.

WTSE$ -- Wait for Single Event Flag

The WTSE$ directive is included for compatibility with RSX-11M, in which some programs are written with a QIO$ followed by a WTSE$, to cause the program to wait until the I/O is complete. This has the same effect as a QIOW$ directive. Since all I/O is synchronous in RSX-emulation mode, QIO$ performs the same as QIOW$. Thus, the WTSE$ directive returns immediately; it performs no operation in RSTS/E.

Privileges Required

None

Macro Call

WTSE$ *efn*

where:

efn	is the event flag number. (The RSX emulator ignores this parameter.)

Macro Expansion

```
WTSE$     0
.BYTE     41.,2            ;DIC=41., DPB SIZE = 2 WORDS
.WORD     0                ;FLAG IGNORED IN RSTS/E
```

Local Symbol Definition

W.TSEF	Event flag number (2)

DSW Return Codes

IS.SUC	Successful completion.
IE.SDP	DIC or DPB size is invalid.

Chapter 6: RT11 Run-Time System Environment

The RT11 run-time system emulates the single-job monitor of the RT-11 operating system on RSTS/E systems. You can use many of the programmed requests available to MACRO programmers on RT-11 systems. In addition, the RT11 run-time system provides some directives, suited to the RSTS/E environment, that are not available for the RT-11 operating system.

The size limit for programs running under the RT11 run-time system is 27K words, slightly less than the 28K words allowed by the RT-11 operating system. The run-time system itself takes a 4K-word high segment. It also uses 1K words in the user job image area as a scratch pad. As Chapter 2 notes, run-time systems are usually mapped read-only, so they can be shared by more than one user on a RSTS/E time-sharing system. However, the RT-11 operating system lets you access certain areas within the resident monitor. Thus, to allow a similar capability under RSTS/E, these areas must be located in the user job image area, which is mapped read/write.

If you use the RT11 directives (see Chapter 7), you must assemble your program with the MACRO assembler and link the modules with the LINK linker. The *RSTS/E RT11 Utilities Manual* describes how to run MACRO and LINK to assemble and link your program.

Advantage: Transportable Code

The RT11 directives are useful if you are coding a program to be run under both the RT-11 and RSTS/E operating systems. RSTS/E emulates the majority of the RT-11 single-job programmed requests. Of those which are not, most are ignored on RSTS/E systems, including the foreground/background (FB) and extended memory (XM) calls. That is, most RT11 directives do not cause errors on RSTS/E systems (see Chapter 7 for a list of exceptions).

Those RT11 requests that are emulated are fitted to the RSTS/E environment. For example, under the RT-11 operating system, there are three types of read:

- o .READ -- Initiates a read operation and transfers control back to the user program; the read may or may not be complete.

- o .READW -- Initiates a read operation and transfers control back to the user program only after the read operation is completed.

- .READC -- Initiates a read operation and transfers control to the user program; then, when the read is complete, transfers control to a user-specified completion routine.

Under RT11 emulation, all I/O is synchronous; that is, control is returned to the user program only when the I/O is complete. So, .READ and .READW are implemented the same under RSTS/E: as a "read and wait." The .READC transfers control to the completion routine when the read operation finishes.

The significance of the differences depends on what you want to do. If you are transporting an existing program from RT-11 to RSTS/E, you can generally be confident that the program works as it should.

If you are developing a program for RSTS/E only, you do not have to be concerned with how the directives work on RT-11. You can choose .READ or .READW: both work the same. You can also choose .READC if, for example, you wanted to code one completion routine to be used after different read operations.

If you are developing a program on RSTS/E to be run under RT-11, or both RSTS/E and RT-11, you need to know how the directives work under RT-11 and under RSTS/E. You can choose between .READ, .READW, and .READC according to what they do on both systems.

Chapters 6 and 7 describe what RT11 directives are available, as well as what they do under RSTS/E. See the *RT-11 Programmer's Reference Manual* for a complete description of how these directives work in the RT-11 operating system environment.

You can also transport assembled (binary) programs from RT-11 systems to RSTS/E systems, if you keep in mind the limitations of the emulated RT11 run-time system environment. You must change the run-time system name that RSTS/E maintains in the file directory entry for each runnable file. The easiest way to do this is with the /RUNTIME_SYSTEM qualifier in COPY. For example:

```
  COPY MT0:*.* *.*/RUNTIME_SYSTEM=RT11
```

This command transfers all files on magnetic tape unit 0 to your account on the public disk structure as files to be run under the RT11 run-time system.

For binary files that have already been transferred to your system, type:

```
  SET FILE file.typ/RUNTIME_SYSTEM=RT11
```

General Services

Besides transportable code, the RT11 emulator provides:

- o Simple input/output. The I/O operations under the RT11 run-time system are simple to code and include file-structured input/output operations.

- o Fast program development. The LINK linker executes faster than its RSX emulator counterpart TKB. However, LINK does not provide as many features.

System Macro Library

The RT11 emulator directives that you code into your program are macro calls; MACRO-11 expands them into executable code at assembly time. The macro expansions for the RT11 emulator directives are contained in the system macro library ($SYSMAC.SML). You can name this library file in the input-file list when you assemble your program. For example:

```
  MACRO OBJ,OBJ=$SYSMAC.SML/M,SRC1,SRC2
```

(MACRO-11 searches the library automatically to resolve undefined symbols, so this is not really necessary.)

However, you must use the MACRO-11 .MCALL directive in your code to define the directives you use as external macros needed to assemble the source program. The .MCALL must appear before the first directive is called. For example:

```
  .MCALL .READ,.DATE,....
      .
      .
      .
  .DATE
```

Alternatively, you can enable automatic macro library search using the .ENABL MCL macro directive. See the *PDP-11 MACRO-11 Language Reference Manual* for details.

Note that some RT11 emulator directives have the same form as general monitor directives (.DATE, for example). If you want to use the RT11 .DATE, specify it in an .MCALL directive. Then, regardless of whether you assemble with the prefix file COMMON.MAC, the .DATE is expanded from the system library as a call to the RT11 run-time system.

If you want the general monitor .DATE, do not specify .DATE in an .MCALL directive. Instead, use the special prefix emulator trap (EMT) instruction (EMT 377) immediately before the .DATE call, and assemble with COMMON.MAC.

Note

> You must precede all general monitor directives that you use with an EMT 377. RT11 uses EMTs in the range 0 to 100, the same range used by the RSTS/E general monitor directives. To bypass the general monitor calls, the RT11 run-time system defines a special prefix EMT for the monitor (see Chapter 2), so all EMTs except those preceded by an EMT 377 are processed by the RT11 run-time system.
>
> You should not use the general monitor calls .CHAIN, .EXIT, .CORE, .RTS, .FSS, .RUN, or .CCL from a program running under control of the RT11 run-time system. Results from using these directives under RT11 emulation are unpredictable.

Directive Processing

It is easier to understand how you specify RT11 directives and their arguments if you first understand how the directives are expanded at assembly time, and how they are processed by the RT11 run-time system at execution time.

Generally, the expansions end with an EMT instruction passing control to the RSTS/E monitor. The monitor then passes control to the RT11 run-time system for processing.

Note

> The RT11 run-time system has the PF.EMT bit set in the P.FLAG word in the pseudovector region (see Chapter 2). Thus, the RT11 run-time system can use all possible values in the low byte of the EMT instruction, just as the RT-11 operating system does. The low-byte of the P.FLAG word is set to 377, which, in combination with the setting of the PF.EMT bit, defines the special prefix EMT 377.

The RT11 run-time system examines the low byte of the EMT instruction to determine either the action to be performed or the place to look for information that defines the action to be performed.

For example, if the call translates to code ending in an EMT in the range 340 to 357, the RT11 run-time system determines the action to be performed from the EMT and looks in general register R0 and/or on the stack for other arguments used in the call.

If, on the other hand, the call translates to an EMT 374 instruction, the RT11 run-time system looks in R0 for a one-byte function code and, if necessary, a one-byte argument. For an EMT 375 instruction, the RT11 run-time system looks in R0 for the address of an argument block. It then examines the argument block to find the function code defining the action and whatever other arguments it expects, depending on the function.

Table 6-1 summarizes the EMTs that the RT11 run-time system processes. Unless you want to bypass the directive expansions and load R0 and other arguments yourself, Table 6-1 is important to you because:

- o All the directives use general register R0. You must preserve the contents of R0, if necessary, by saving and restoring the contents before and after issuing a directive. Some directives return information to R0; otherwise, the contents of R0 are unpredictable upon completion of a call. All other general registers are preserved as you left them.

- o Some of the directives require that you allocate space in your program for arguments. The expansions for such directives fill the argument block you specify in the call with the values you specify as other arguments in the call (see the next section "Call Forms").

If an error occurs when a directive is processed, control returns to the user program with the carry bit set in the PSW and an error code in byte 42 of the low 512. bytes of memory.

Table 6-1: EMT Instructions Recognized by the RT11 Run-Time System

EMT Low-Byte	Meaning
EMT 377	Reserved; this is the special prefix EMT. You must code this EMT immediately before any of the general monitor calls in Chapter 3.
EMT 376	Used internally by the RT11 run-time system. Never execute this EMT from a user program; the program aborts with an error. The error message displayed is: ?M -OVLY error at user PC nnnnnn."
EMT 375	Indicates a call that has several arguments: R0 contains the address of the first byte of an argument block. `(R0) = | address of argument block |`
EMT 374	Indicates a call with one argument. R0 contains a function code in the high-order byte and the argument in the low-order byte. `(R0) = | function code | argument |`
EMTs 372-373	Ignored by the RT11 run-time system. Do not use these EMTs; they are reserved for future use by DIGITAL.
EMTs 360-371	EMTs specific to the RT11 run-time system under RSTS/E; not available under the RT-11 operating system.
EMTs 340-357	Indicates a directive that expects arguments either on the stack, in R0, or both.
EMTs 0-337	Indicates a directive used by Version 1.0 of the RT-11 operating system. These EMTs are also recognized by the RT11 run-time system.

Call Forms

Chapter 7 shows RT11 directive calls in two basic formats: those with an area argument and those without. The area argument indicates that the call uses an argument block. You must allocate space for the argument block.

Format for Calls Using Argument Blocks

The previous section describes how directives with an area argument use an argument block. Chapter 7 shows the format for these directives as:

```
  .RTCALL area, arg1, arg2, ...., argn
```

The area argument is the address of the first byte of the area you have set aside for the argument block. The remaining arguments (arg1, arg2, and so forth) are the values needed for the call. The RT11 emulator places them in the argument block at execution time.

In general, the argument block format is:

```
+----------------+----------------+
|  function code | channel number | <--(R0)
+----------------+----------------+
|           argument 1            | (R0) + 2
+----------------+----------------+
|           argument 2            | (R0) + 4
+----------------+----------------+
|                .                |
|                .                |
|                .                |
+----------------+----------------+
|           argument n            | (R0) + n*2
+----------------+----------------+
```

R0 points to location x. The high byte (location x+1) contains the function code defining the action to be performed. The low byte defines a channel number, if one is needed for the call. Remaining arguments, if any, are stored in subsequent words.

Consider the hypothetical directive .HYPOT, using four arguments including the area argument:

```
  .HYPOT area, arg1, arg2, arg3
```

If you were to specify all four arguments in the call, at assembly time MACRO-11 expands the call to a code that:

- o Moves the area argument (an address) to R0.
- o Moves the function code for .HYPOT to the address specified by (R0)+1, and, if a channel number is used, moves that channel number to (R0). (The notation (x) means the contents of x -- thus the channel number is moved to the address specified by the contents of R0.)
- o Moves remaining arguments to subsequent words in the argument block.
- o Executes an EMT 375.

Thus, all of the arguments for this form should be appropriate as operands in MOV instructions. For example:

```
AREA:   .WORD   0,0,0
          .
          .
          .
        .HYPOT  #AREA,#4,NUMNUM,#300
```

Under certain conditions, you can leave out arguments in directives using an argument block. If you leave out area, for example, you must load R0 with the address of the argument block yourself, before executing the directive. For example:

```
MOV     #AREA,R0
.HYPOT  ,#4,NUMNUM,#300
```

If you leave out any of the other arguments, the expansion does not load any new values into those positions in the argument block; those words are left untouched. For example:

```
AREA:   .WORD   0,0,0
          .
          .
          .
        .HYPOT  #AREA,,NUMNUM,#300
```

The execution of .HYPOT uses the value 0, the contents of location NUMNUM, and 300 as arguments. Suppose that the following directive is executed next:

```
.HYPOT  #AREA,#3,,#200
```

When the directive is executed, the argument block contains the value 3, the contents of location NUMNUM, and 200.

Format for Calls Not Using Argument Blocks

For those calls that do not use an argument block, Chapter 7 shows the format as:

```
   .RTCALL arg1, arg2, ..., argn
```

These directives expand to code that stores the arguments, if any, in R0 and/or the stack. (The RT11 emulator removes any values pushed on the stack before control returns to the user program.)

Again, the arguments should be appropriate as operands in MOV instructions. You can omit certain arguments for this form; Chapter 7 indicates these possible omissions with brackets when it describes the directive calls. For example, if the macro call format was as follows, you could leave out the last argument:

```
   .HYPOT arg1, arg2[, arg3]
```

Channel Number and Device Block Arguments

Many RT11 directives use arguments that define a channel number and device block. A device block is a 4-word area containing the device and file specification. The following general comments apply to these arguments.

Channel Number Arguments

Like the general monitor calls, RT11 directives which handle I/O use a channel number to refer to a device. Allowable RT11 channel numbers range from 0 to 15. When you open a file or device using the RT11 directives .LOOKUP or .ENTER, you give a channel number in this range. The RT11 emulator relates the number specified in the call to the first free RSTS/E channel number, with the following considerations:

- The job's terminal (RSTS/E channel 0) is never free for the emulator to relate to an RT11 channel number. In RT11, you do I/O to the job's terminal with specific directives, such as .PRINT and .GTLINE. Thus, since RSTS/E channel numbers range from 0 to 15 but RSTS/E channel 0 is always busy as far as the RT11 run-time system is concerned, you can open up to 15 channels using any 15 of the 16 possible RT11 channel numbers (ranging from 0 to 15).

- The RT11 emulator always attempts to allocate RT11 channel number 15 to RSTS/E channel 15. Channel 15 is the channel on which the monitor opens the user job image file when RSTS/E executes a .RUN directive (see Chapter 3). The run-time system loads the file and begins its execution. If the program is overlaid, the run-time system keeps channel 15 open. If not, the run-time loads the file and then closes channel 15.

Thus, if your program is not overlaid and you open RT11 channel 15, the call succeeds (unless you already have channel 15 open). If your program is overlaid, you cannot open channel 15; any attempt to do so results in an error indicating the channel is in use.

Other than channel 15, there is no correspondence between the RT11 channel number and the RSTS/E channel number. They may or may not be the same.

Device Block Arguments

Several RT11 directives use a device and file specification. The argument in these cases is an address pointing to the first word of a 4-word device block that you allocate in your program. The format of the device block is 12 RAD50 characters:

```
+-------------------------+
| device and unit number  | <--(devblk)
+-------------------------+
|       filename          |
|       (2 words)         |
+-------------------------+
|       file type         |
+-------------------------+
```

You can use any valid RSTS/E device and unit number (0-7) for the first word of the device block. You can also define user logical names up to three characters long. If you need device numbers that have more than one digit (for example, KB27 or TT12), you must use the following steps to form the algorithm:

1. Form the RAD50 representation of unit zero of the device you want. For example:

   ```
   BASEKB=^RKB0
   ```

2. Add the decimal value of the unit number wanted. For example, to get the proper representation of KB36:

   ```
   KBN36=BASEKB+36.
   ```

3. Use this value as the first word of the device block. For keyboards, there is no file name and type, so you can construct the device block as:

```
DEVBLK:         .WORD KBN36
 .WORD          0,0,0
```

Note

If you assign a logical name with the same RAD50 bit pattern as some device/unit-number that you construct, the logical name takes precedence. The previous example, for instance, forms the RAD50 bit pattern equal to the ASCII characters KCZ. Thus, if you assign the logical name KCZ to the line printer, any call referring to DEVBLK in the previous example would use the line printer not KB36.

Low 512. Bytes for RT11 Run-Time System

Table 6-2 summarizes the locations in the low 512. bytes of virtual address space that the RT11 run-time system uses.

Table 6-2: Locations in First 512. Bytes That RT11 Uses

Locations	Contents
30,31	Contains the contents of FIRQB+FQNENT at the time the .RUN was executed. (See the .RUN description in Chapter 3, and P.RUN description in Chapter 2.)
32,33	Contains the contents of XRB+0 when the run-time system was entered at P.RUN (see Chapter 2).
40,41	Contains the starting address of the job.
42,43	Contains the initial value of the stack pointer. If this value is not set by the user program in an .ASECT or through a LINK option, the LINK linker defaults this word to 1000.

Table 6-2: Locations in First 512. Bytes That RT11 Uses (Cont.)

Locations	Contents	
44,45	Contains the Job Status Word (JSW). RSTS/E uses JSW as a flag word for the run-time system. Certain bits are maintained by the RT11 run-time system exclusively, while others can be set or cleared by the user program. The bits you can set are marked with an asterisk in the following list. Do not use currently unassigned bits for your own purposes; future RT11 run-time system releases may use them.	
	Bit	**Meaning**
	*15	Core common bit. If set with .ASECT, preserves core common when the user program is executed.
	*14	Lowercase bit. When set, disables automatic conversion of lowercase to uppercase.
	*13	Reenter bit. When set, indicates that you can restart the program from the terminal with the REENTER command.
	*12	Special mode TT bit. When set, indicates that the job is in a special keyboard input mode. See .TTYIN/.TTINR directive descriptions, Chapter 7.
	9	Overlay bit. When set, indicates that the job uses the LINK overlay structure.
	8	CHAIN bit. If set, virtual addresses 500-776 are loaded from the memory image file when the job is started. (These words are normally used to pass parameters between programs.) The RT11 run-time system sets the bit if you entered the job with the RT11 .CHAIN directive.
	*6	Inhibit TT wait bit. When set, indicates that the job can accept input from the job's terminal in ODT mode with .TTYIN/.TTINR; that is, one character at a time, as you type the characters rather than one character at a time after you type a whole line.
46,47	Points to the first word of the RT11 read/write scratch pad area in the user job image area.	

Table 6-2: Locations in First 512. Bytes That RT11 Uses (Cont.)

Locations	Contents
50,51	Contains the high memory address. The run-time system maintains the highest virtual address the user program can use in this word. It is initially set to the address of the last word of the user program. You can change it with the .SETTOP directive (see Chapter 7) to any address up to the start of the scratch pad area. That is, the maximum value this word can be is the contents of the word at location 46,47 minus 2.
52	Contains the EMT error code. If the directive results in an error, the code number of the error is always returned in byte 52 and the carry bit is set. Always refer to location 52 as a byte, not a word.
54,55	Contains the beginning address of the scratch pad area.

Scratch Pad Area in User Job Image

The RT11 run-time system uses part of the user job image space as a scratch pad. Some of the information in this scratch pad is the same as the information in the RT-11 operating system, as the *RT-11 Programmer's Reference Manual* describes. Other information is unique to RSTS/E. For example, you can set a word in this area with a PPN to be used when opening a file. RT-11 does not allow a PPN in its file specification; RSTS/E does.

In any case, the RT11 emulator adds the scratch pad area to the end (high virtual addresses) of the user job image at execution time. If you do not use RT11 SIZE command (see the *RSTS/E System User's Guide*), the emulator determines the highest address of the program from the .SAV file, an item which the LINK linker calculates, and rounds the address up to the next multiple of 1024 (1K words) to determine the starting address of the scratch pad. If you use the RT11 SIZE command, the emulator places the scratch pad in the top 1K words of the address space that SIZE requests.

Thus, the maximum size of MACRO programs running under the RT11 run-time system is 27K words. Any program larger than this causes the ??Maximum memory exceeded error message when the file is run.

You can determine the start of the scratch pad area from word 54 in the low 512. bytes of memory.

Table 6-3 lists the offsets to locations in the scratch pad area that may be of use to you.

Table 6-3: Offsets to Important Scratch Pad Area Locations

Octal Offset	Mnemonic Offset	Contents
0	PPN	If nonzero, used as a PPN for all of the calls named in this section except .READ and .WRITE. (Note that for .SETFQB, this value is loaded into FIRQB+FQPPN and overrides any PPN resulting from examination of the string by the corresponding .CSISPC directive.) For .READ and .WRITE, this word forms the modifier word (at XRB+XRMOD) in the corresponding RSTS/E .READ or .WRITE directive.
2	PROTEC	If nonzero, this word is the protection code to be used for all of the calls named in this section except .READ and .WRITE. (Note that for .SETFQB, this value is loaded into FIRQB+FQPROT and overrides any protection code resulting from examination of the string by the corresponding .CSISPC directive.) For .READ and .WRITE, this word forms the most significant bits of the block number in the corresponding RSTS/E .READ or .WRITE (at XRB+XRBLKM). This value supplements the block number argument from the RT11 .READ or .WRITE call.
4	MODE	For RT11 calls .LOOKUP, .ENTER, and .REOPEN, this word forms the mode word at FIRQB+FQMODE in the corresponding RSTS/E CALFIP call. For .SETFQB, this word is loaded into FIRQB+FQMODE and overrides any /MODE qualifier that the .CSISPC directive finds.
6	CLUSTR	For RT11 directives that create files, such as .ENTER, this word forms the cluster size (at FIRQB+FQCLUS) for the corresponding RSTS/E CALFIP call. For .SETFQB, this word is loaded into FIRQB+FQCLUS, and overrides any /CLUSTERSIZE qualifier that the .CSISPC directive finds.

Table 6-3: Offsets to Important Scratch Pad Area Locations (Cont.)

Octal Offset	Mnemonic Offset	Contents
10	POSITN	For RT11 directives that create files, such as .ENTER, this word forms the device cluster number for file placement (at FIRQB+FQNENT) in the corresponding RSTS/E CALFIP call. For .SETFQB, this word is loaded into FIRQB+FQNENT and overrides any /POSITION qualifier that the .CSISPC directive finds.
12	CRMSBS	For RT11 calls that preallocate space for a file on its creation (.ENTER), this word forms the most significant bits (MSBs) of the file size (at FIRQB+FQSIZM) in the corresponding CALFIP call. For .SETFQB, this byte is loaded into FIRQB+FQSIZM.
66	NOCTLC	This word is used by the RT11 run-time system to stop user-entered CTRL/Cs from interrupting program execution during certain crucial sequences (.SRESET, .HRESET, .LOOKUP, .ENTER, .REOPEN, .CLOSE, and .SAVESTATUS). When set to 177777, user-entered CTRL/Cs interrupt execution. A zero or positive value inhibits CTRL/C interrupts. While this word is zero or positive, any user-entered CTRL/C increments the contents of this word by one. When the emulator reenables CTRL/Cs, it processes any pending CTRL/C by either: o Passing control to a user routine specified with .SETCC (Chapter 7) o If no .SETCC is in effect, passing control to the job keyboard monitor. If the job keyboard monitor is RT11, the keyboard monitor commands CONTINUE or CCONTINUE resume execution of the program at the point where the interrupt occurred.

Table 6-3: Offsets to Important Scratch Pad Area Locations (Cont.)

Octal Offset	Mnemonic Offset	Contents
66 (cont.)		Your program can clear this word to inhibit CTRL/Cs, process them later in whatever manner you see fit, and reenable CTRL/C interrupts by setting this word to 177777. However, note that the RT11 emulator clears the word and then reenables interrupts when .SRESET, .HRESET, .LOOKUP, .ENTER, .REOPEN, .CLOSE, or .SAVESTATUS is executed.
276		Emulator version number. The same as the RT-11 operating system version number that the current RT11 run-time system emulates.
277		Emulator release number. The same as the RT-11 operating system release number that the current RT11 run-time system emulates.
300		Configuration word. Set for RSTS/E environment, as follows: If FP11 floating-point hardware exists, bit 6 is set to one. Bit 9 is always set to one, and all other bits are set to zero. (If your program tests these bits on an RT-11 system, the code also works on RSTS/E.)
370		Extension configuration word. Under RSTS/E, bits 0-3 are always zero, bit 8 is always one (indicating the EIS option is present), and bits 9, 14, and 15 are always zero. (The conditions they indicate on RT-11 systems are not determinable by the RT11 run-time system under RSTS/E.)
372		SYSGEN options word. This word is always zero on RSTS/E systems.
374		User job image size. This word is always zero on RSTS/E systems.

The first six words in the scratch pad are of special significance; the following general comments apply:

- o The directives .LOOKUP, .ENTER, .RENAME, .REOPEN, .DELETE, and .SETFQB examine the first six words, use any values there as described, and clear the first six words before returning control to the user program.

- o The directives .READ and .WRITE examine, use, and clear only the first two words.

- o The .CSIGEN directive clears the first six words before performing any of the .LOOKUP and .ENTER calls resulting from its examination of the command string.

RSTS/E does not emulate some of the RT-11 operating system's single-job monitor calls. RSTS/E ignores the following: .CMKT, .HERR, .LOCK, .QSET, .MRKT, .RELEASE, .SERR, and .UNLOCK. The .CDFN call always returns an error 0 on RSTS/E systems. The following single-job monitor calls are not expanded to include EMTs, and their results on RSTS/E systems are unpredictable: .INTEN, .MFPS, .MTPS, and .SYNCH.

The RT-11 foreground/background (FB) and extended monitor (XM) calls are not supported; they are essentially ignored on RSTS/E systems. They do not return errors.

Table 7-1 lists the directives that are not processed by the RT11 run-time system on RSTS/E systems.

Table 7-1: RT-11 Calls Not Functional on RSTS/E

Ignored	Return Error 0	Unpredictable
.CMKT	.CDFN	.INTEN
.HERR		.MFPS
.LOCK		.MTPS
.QSET		.SYNCH
.MRKT		
.RELEASE		
.SERR		
.UNLOCK		
All FB calls		
All XM calls		

Table 7-2 lists the directives that are processed by the RT11 run-time system on RSTS/E systems.

Table 7-2: RT11 Run-Time System Directives

Mnemonic	EMT	Function Code	Description
..V1.. ..V2..			Lets you assemble directives as they would be expanded under Version 1 or Version 2 of RT-11.
Input/Output and File Operations			
.WRITE .WRITW .WRITC	375	11	Transfers data from a memory buffer to a file or device on an open channel. As with the .READ/W/C directives, .WRITE and .WRITEW operate identically on RSTS/E; .WRITC transfers control to a completion routine after the transfer is complete.
.READ .READW .READC	375	10	Transfers data from a file or device on an open channel to a memory buffer. I/O is synchronous under RT11 emulation on RSTS/E systems, so for all three reads, control returns to the user program only when the transfer is complete. .READ and .READW are identical on RSTS/E systems. .READC transfers control to a user-specified completion routine when the transfer is complete.
.REOPEN	375	6	Opens a file closed with .SAVESTATUS. Using the .SAVESTATUS/.REOPEN combination is easier than using standard opens and closes; the device, file name and type, and project-programmer number are saved automatically with .SAVESTATUS. You only need to keep track of the address in which you stored the file information to reopen it.
.SAVESTATUS	375	5	Closes a file, keeping the information needed to .REOPEN the file later.
.RENAME	375	4	Changes a file's name on disk or DECtape.

Table 7-2: RT11 Run-Time System Directives (Cont.)

Mnemonic	EMT	Function Code	Description
Input/Output and File Operations (Cont.)			
.ENTER	375	2	Creates a file and opens it on a channel.
.LOOKUP	375	1	Opens an already existing file on a channel. .LOOKUP can also be used for non-file-structured I/O operations.
.DELETE	375	0	Deletes a disk or DECtape file.
.CLOSE	374	6	Terminates I/O operations on a channel, performing clean-up operations (positioning tape, and so forth) and releasing the channel for other use.
.PURGE	374	3	Releases a channel without performing the normal clean-up operations. In particular, a file opened with .ENTER and not yet closed is not made a permanent file.
.HRESET	357	--	Closes all open channels, without performing normal clean-up operations. In particular, files opened with .ENTER and not yet closed are not made permanent.
.SRESET	352	--	Functions the same as .HRESET on RSTS/E.
.PRINT	351	--	Displays an ASCII string on the job's terminal.
.GTLIN	345	--	Accepts a line of input from the job's terminal.
.TTYOUT .TTOUTR	341	--	Transfers one character from R0 to the job's terminal.
.TTYIN .TTINR	340	--	Transfers one character from the job's terminal to R0.

able 7-2: RT11 Run-Time System Directives (Cont.)

Mnemonic	EMT	Function Code	Description
Special Functions Related to I/O			
.SPFUN	375	32	Performs special functions for disk, terminal, magnetic tape, and flexible diskette.
.SCCA	374	35	Causes a CTRL/Z entered by the user to be passed on to the user program.
.WAIT	374	0	Checks to see if channel is open.
.CLRXRB (RSTS/E only)	371	--	Clears the XRB (see Chapter 2).
.CLRFQB (RSTS/E only)	370	--	Clears the FIRQB (see Chapter 2).
.DOFSS (RSTS/E only)	365	--	Analyzes an ASCII string in memory for a RSTS/E file name and returns information as for .FSS (general monitor directive). Use instead of .FSS for programs running under RT11 run-time system.
.ERRPRT (RSTS/E only)	364	--	Prints text corresponding to RSTS/E error code on the job's terminal.
.SETCC (RSTS/E only)	362	--	Defines an entry point for CTRL/C traps; allows the user program to handle its own CTRL/Cs.
.SETFQB (RSTS/E only)	360	--	Sets the FIRQB (see Chapter 2) from information returned by the .CSISPC directive.
.RCTRLO	355	--	Restarts programmed output to job's terminal that was stopped by user-entered CTRL/C or CTRL/O.
.CSISPC	345	--	Analyzes an ASCII string (in memory or from the job's terminal) for a standard RT11 command, and sets up device blocks for later opens.

Table 7-2: RT11 Run-Time System Directives (Cont.)

Mnemonic	EMT	Function Code	Description
Special Functions Related to I/O (Cont.)			
.CSIGEN	344	--	Analyzes an ASCII string (in memory or from the job's terminal) for a standard RT11 command, and opens files accordingly.
.FETCH	343	--	Checks to see if device is available on the system.
.DSTATUS	342	--	Returns information about a device to an area in the user program.
System/Job Information			
.GVAL	375	34	Returns the contents of a word in the scratch pad area.
.GTIM	375	21	Returns the time of day.
.GTJB	375	20	Returns the current job number.
.DATE	374	12	Returns the current date.
.DATTIM (RSTS/E only)	361	--	Converts date or time from RSTS/E internal format to ASCII string.
.SETTOP	354	--	Sets location 40 (indicating top of user job image area) to the address specified.
Execution Control			
.SFPA	375	30	Sets address to be entered for floating point errors.
.TWAIT	375	4	Timed wait. Suspends job execution for specified number of clock ticks.
.TRPSET	375	3	Sets address to be entered for reserved instruction and odd address errors; normally trapped to kernel mode addresses 4 and 10.

Table 7-2: RT11 Run-Time System Directives (Cont.)

Mnemonic	EMT	Function Code	Description
Execution Control (Cont.)			
.CHAIN	374	10	Transfers control to another program (file) to be run under the RT11 run-time system.
.DOCCL (RSTS/E only)	367	--	Examines a string for valid RSTS/E CCL command. If valid, transfers control to new program. Use instead of .CCL (general monitor directive) for programs running under RT11 run-time system.
.GETCOR (RSTS/E only)	366	--	Expands size of user job image. Use when .CHAIN is used to transfer control to a program larger than the current program. In addition, use instead of .CORE (general monitor directive) for programs running under RT11 run-time system.
.DORUN (RSTS/E only)	363	--	Transfers control to another program (file) that runs under a non-RT11 run-time system. Use instead of .RUN (general monitor directive) for programs running under RT11 run-time system.
.EXIT	350	--	Terminates execution of the calling job. Note that you should not use the RSTS/E .RTS or .EXIT directives from a program running under the RT11 run-time system.

The remaining sections of this chapter describe the directives in detail, in alphabetical order. Note that for all calls, you need only specify the first six characters. For example, the abbreviation .SAVES works as well as .SAVESTATUS.

All the examples in this chapter show Version 2 expansions. Note that Version 1 expansions require different arguments in the calls. See RT-11 system documentation if you need to use Version 1 expansions.

.CHAIN -- Pass Control to Another Program Under RT11

The .CHAIN directive transfers control to another program without altering the run-time system. The .CHAIN leaves I/O channels from the old program still open for the new program. Thus, you can get around the 27K word limit for programs to be run under the RT11 run-time system. If the program you are chaining to takes more space than the current program, use the .GETCOR directive to expand the space allocated for the user job image area before executing the .CHAIN.

Virtual addresses 500-507 are expected to contain the device and file name of the program to chain to, in the RT11 device block format (see Chapter 6). The area from locations 510-777 can be used to pass information between the chained programs. Make sure, though, that the program you are chaining to does not inadvertently destroy the data in locations 510-777 by pushing data on the stack. Note that the linker defaults the initial stack setting to 1000.

An executing program can tell whether it was chained to or run by some other means by examining bit 8 of the Job Status Word (JSW) (see Chapter 6). If bit 8 = 1, the program was invoked with .CHAIN, and locations 500-777 are preserved from the program that issued the .CHAIN. If bit 8 = 0, the program was not entered with an RT11 .CHAIN.

The .CHAIN directive works only for RT11-to-RT11 transfers. If you want to chain to a program to be run under some run-time system other than RT11, use the .DORUN directive (see Chapter 6). (The description of .DORUN also describes what happens for RT11 programs receiving control from non-RT11 programs.)

Privileges Required

Read or execute access to the file.

Macro Call

```
   .CHAIN
```

Request Format

```
      +----------+----------+
R0 =  |    10    |     0    |
      +----------+----------+
```

Errors

Since .CHAIN is intended to pass control to another program, no errors are returned to the calling program. If the new program cannot be chained to, the .CHAIN is abandoned, and the keyboard monitor is entered. (Control passes to the P.NEW entry point of the RT11 run-time system, which is the keyboard monitor's entry point for RT11.)

Example

The following code requests an expansion to 20K words and chains to a sample file named MYFILE.SAV in the user's account:

```
  FILE:    .RAD50     /MYFILESAV/     ;Define text passed in chain
           .
           .
           .
           MOV        #20.,R0         ;Indicate 20.K now needed
           .GETCOR
           CLR        500             ;Default for the device
           MOV        #FILE,502       ;Move
           MOV        #FILE+2,504     ;  the
           MOV        #FILE+4,506     ;    file name
           .CHAIN                     ;Invoke the new program
```

.CLOSE - Close a Channel

Use the .CLOSE directive to terminate I/O on a particular channel. The .CLOSE directive:

- o Performs whatever action is appropriate to the device (such as updating a disk directory to indicate a new file, repositioning a magnetic tape, and so forth)
- o Frees the RT11 channel number for use with another device
- o Frees the RSTS/E channel from its association with the RT11 channel number (effectively making another channel available).

A .CLOSE on a file opened with .ENTER makes the file permanent; the file is tentative until that time. Note that, unlike the RT-11 operating system, RSTS/E does not deallocate unused space. Thus, if you preallocated space when you opened the file with .ENTER, the file contains all this space after the .CLOSE.

Privileges Required

None

Macro Call

.CLOSE *chanum*

where:

chanum is a channel number in the range 0 to 17 (octal). See Chapter 6 for rules on channel numbers.

R0 Format

```
        +----------+----------+
(R0) =  |     6    |  chanum  |
        +----------+----------+
```

Errors

No errors are possible with the .CLOSE directive. RSTS/E ignores a .CLOSE for a channel that is not currently open.

Example

The following statement closes channel 5:

```
.CLOSE #5
```

.CLRFQB -- Clear the FIRQB

This directive clears the FIRQB area in the low 512. bytes of virtual memory (see Chapter 2). No data is passed or returned for this directive.

Privileges Required

None

Macro Call

```
   .CLRFQB
```

Note that .CLRFQB, like all RSTS/E-specific directives under RT11, is not expanded at assembly time. (It is not defined in the SYSMAC.SML file as a directive.) You can code a definition of the directive in a data section of your program or use .MACRO to define it. See the following example.

Errors

No errors are possible with the .CLRFQB directive.

Example

The following example clears the FIRQB area in the low 512. bytes of virtual memory:

```
  .CLRFQB = EMT+370
              .
              .
              .
           .CLRFQB
```

.CLRXRB -- Clear the XRB

This directive clears the XRB area in the low 512. bytes of virtual memory (see Chapter 2). No data is passed or returned with .CLRXRB.

Privileges Required

None

Macro Call

```
    .CLRXRB
```

Note that .CLRXRB, like all RSTS/E-specific directives under RT11, is not expanded at assembly time. (It is not defined in the SYSMAC.SML file as a directive.) You can code a definition of the directive in a data section of your program or use .MACRO to define it. See the following example.

Errors

No errors are possible with the .CLRXRB directive.

Example

The following example clears the XRB area in the low 512. bytes of virtual memory:

```
  .CLRXRB = EMT+371
              .
              .
              .
          .CLRXRB
```

.CSIGEN -- Examine String for RT Command, Open Files

The .CSIGEN directive examines a string (either in memory or from the job's keyboard) to process a standard RT11 command string. If the string is to be accepted from the job's keyboard, the .CSIGEN first prints an asterisk (*) prompt on the keyboard and then waits for you to type a line.

If the string is in the correct form, any files or devices open on RT11 channels 0 - 10 (octal) are closed and then either reopened or left inactive, depending on whether or not a file is given for that slot. For example:

```
           objfile,listfile,crefile=srcfile1,srcfile2,...srcfile6

  channel     0        1         2        3        4           10
```

Any valid RSTS/E file specification can appear in the string, including file specifications with PPNs, and logical names. However, the file specifications cannot contain wildcards. Files on the left side of the equal sign are opened with .ENTER, and the files on the right side of the equal sign are opened with .LOOKUP. If the equal sign is missing, RSTS/E assumes an equal sign at the beginning of the string. For example, consider the string:

```
NOW.OBJ,NOW.LST,NOW.CRF=THEN.MAC
```

The file NOW.OBJ would be opened with .ENTER on channel 0; NOW.LST, on channel 1; NOW.CRF, on channel 2. The file THEN.MAC would be opened with .LOOKUP on channel 3. Two more examples are:

```
NOW.OBJ,,NOW.CRF=
```

The file NOW.OBJ would be opened with .ENTER on channel 0 and NOW.CRF on channel 2. Channels 1 and 3-10 would be inactive.

```
LP:
```

The line printer would be opened with .LOOKUP on channel 3.

RSTS/E Options

The file specifications can include any of the standard RSTS/E qualifiers processed by the monitor itself (see the .FSS directive, Chapter 3). These include: /CLUSTERSIZE, /RONLY, /MODE, /PROTECTION, /FILESIZE or /SIZE, and /POSITION. The RT11 Emulator can handle any of these optional qualifiers that .CSIGEN finds in the command string.

RT11 Options

The .CSIGEN directive under RSTS/E also does RT11-type analysis of options. One or more options can appear after any of the file specifications. The general form of an option is:

```
/a:val1:val2:...:valn
```

The slash (/) indicates an option; it must be followed by a one-character option code. (The character does not have to be printable.) The slash-character pair can be followed by one or more values. A colon (:) indicates that a value follows. If given, the colon must be followed by:

- o An octal number in the range 0 to 177777.
- o A decimal integer in the range -32768. to 32767. (the terminating period must appear).
- o From one to three alphanumeric characters, with the first character alphabetic.

The .CSIGEN directive examines any options in the string to ensure that they are in the correct form, reformats them, and pushes them on the stack for the user program to examine. The information pushed on the stack is ordered by channel; if an option is given for the file specification for channel 0 (assuming there is a file specification in the first slot), it is pushed on the stack first, followed by option information for the file specification for channel 1 (if any), and so forth. The last word pushed on the stack is an integer value indicating the total number of options in the string. An option having two values is formatted as two options. The format is:

```
+-------------------------+-------------------------+
|          number of options in string              |
+-------------------------+-------------------------+
|   value-flag/channel    |   option code (ASCII)   |
+-------------------------+-------------------------+
|     value (integer or RAD50) or next option       |
+-------------------------+-------------------------+
|                         .                         |
|                         .                         |
|                         .                         |
+-------------------------+-------------------------+
```

Bit 15 of a value-flag/channel byte is set to one if the option had an associated value, to zero if it did not. Thus, if bit 15 is set, the next word is the value associated with the option. (The option code itself is in the low byte of this word.) If bit 15 is zero, the next word is the next option (if any). Bits 8 to 14 of a value-flag/channel byte indicate the RT11 channel number with which the file specification having the option is associated.

Privileges Required

Read or write access to the files specified. DEVICE privilege is required to access a restricted device.

Macro Call

.CSIGEN *devspc*,*defext*,[*cstrng*][,*linbuf*]

where:

devspc is ignored on RSTS/E systems.

defext is the address of a four-word area containing default file types, in RAD50 format. The format of the area is:

```
+----------------------------------------+
| default for files open on channels 3-10|
+----------------------------------------+
|   default for file open on channel 0   |
+----------------------------------------+
|   default for file open on channel 1   |
+----------------------------------------+
|   default for file open on channel 2   |
+----------------------------------------+
```

These values are then used when no corresponding file type is given in the examined string. For example, in the MACRO assembler, the first word would contain MAC in RAD50 format; the second, OBJ; the third, LST; and the fourth, CRF.

cstrng is the address of the ASCIZ string to be examined or a #0 if input is to come from the job's terminal. If this argument is not specified, input is automatically taken from the job's terminal.

linbuf is the address where the original string (if accepted from the job's terminal) is to be stored. This area should be 81. bytes long. The string will be terminated with a zero byte, instead of a carriage-return/line-feed combination.

If this argument is omitted, the original string is not kept.

Errors

If an error occurs when input is accepted from the job's terminal, the user gets another try. An error message is printed on the terminal, along with another * prompt, and the directive waits for you to type another line.

If the string being examined is in memory and an error occurs, the carry bit is set, and one of the following error codes is returned in byte 52. The options and option-count are removed from the stack.

Code	Meaning
0	Illegal command (bad separators, illegal file name, command too long, and so forth).
1	A device specification was not recognizable or the device is not available on the system.
2	Unused.
3	An attempt to open a file with .ENTER failed because of a full directory.
4	An input file (to the right of the equal sign) was not found in a .LOOKUP.

Example

The following example includes option switches and shows more clearly how options and values are pushed on the stack:

```
DEFEXT: .RAD50    /MACOP1OP2OP3/
        .
        .
        .
        .CSIGEN   #0,#DEFEXT,#0
```

Assume that the line typed at the job's terminal is:

```
OUTFIL/M:32.,,ODDFIL=[2,200]HISFIL,DT0:HERFIL/G:NRD:20
```

When control returns to the user program following the .CSIGEN, the file OUTFIL.OP1 on the public structure has been opened with the .ENTER directive on channel 0, and the file ODDFIL.OP3 on channel 3. The file HISFIL.MAC on the public structure, under account [2,200], has been opened with the .LOOKUP directive. (If the calling job did not have the necessary privilege, an error message would have been displayed on the terminal, and another line accepted.) The file HERFIL.MAC on DECtape unit 0 has also been opened with .LOOKUP.

The stack contains:

	Word	Octal Value	Explanation
(SP) →	1	3	Three options were found in the string. (The G option has two values.)
	2	102107	Last option has a value, and the option is associated with channel 4. The option code is G.
	3	000020	The value of the option is 20 (octal).
	4	102107	Next-to-last option has a value, and the option is associated with channel 4. The option code is G.
	5	055124	Value of option is NRD in RAD50 format.
	6	100115	The first option has a value, and the option is associated with channel 0. The option code is M.
	7	000040	The value of the option is 40 (octal).

.CSISPC - Examine String for RT Command, Create Devblk

Like .CSIGEN, the .CSISPC directive parses a string (either in memory or from the job's terminal) to see if it is a valid RT11-type command string. If the string is to be accepted from the job's terminal, .CSISPC first displays an * prompt on the screen and then waits for you to type a line. Instead of closing and opening files, however, .CSISPC simply returns a 39-word block of information about the files named in the string. The user program can later use this information to open the files. See the previous discussion of .CSIGEN for command string format and option processing.

Note that .CSISPC, like .CSIGEN, stores RSTS/E-specific file information (PPN, and so on) within the scratch pad area. Furthermore, you can issue a .SAVESTATUS or .SETFQB referencing the device-block information returned by .CSISPC (although .SAVESTATUS works only for input files to the right of the equal sign when opened with .LOOKUP).

If you use .CSISPC in this way, be sure that you reference addresses in the *outspec* area returned by .CSISPC. If you move the information in the *outspec* area and then refer to the new area in .SAVESTATUS or .SETFQB, the RSTS/E-specific information, such as PPN, is not used. (The emulator keeps a pointer in the impure area relating the PPN and other RSTS/E-specific information to the area returned by .CSISPC.)

Privileges Required

None

Macro Call

```
    .CSISPC outspc,defext,cstrng[,linbuf]
```

where:

outspc is the address of a 39-word area to contain the file descriptors produced by .CSISPC. This area can overlay the space allocated to *cstrng*, if desired. The format of the information returned is:

Offset	Contents
0 6	devblk for channel 0
10	size
12 20	devblk for channel 1
22	size
24 32	devblk for channel 2
34	size
36 44	devblk for channel 3

Offset	Contents
46 54	devblk for channel 4
56 64	devblk for channel 5
66 74	devblk for channel 6
76 104	devblk for channel 7
106 114	devblk for channel 10

Information for the files for channels 0, 1, and 2 is stored in 5-word blocks. The first four words are RAD50 information in the standard RT11 device block format (see Chapter 6). The last word is the file size indicated by a /FILESIZE or /SIZE option, if any was encountered for these files in the string. Since files on the left side of the equal sign are opened with .ENTER, they can have a size for preallocation. Information for the files for channels 3 to 10 is stored in 4-word blocks, in the standard RT11 device block format. If any files are omitted in the string, the corresponding 4- or 5-word block is filled with zeros.

defext is the address of a four-word block containing the RAD50 default file types. (See the discussion of .CSIGEN for the format of this area.) These defaults are returned to the *outspec* area if no types are given in the examined string.

cstrng is the address of the ASCIZ string to be examined or a #0 if input is to come from the job's terminal. If this argument is not specified, input is automatically taken for the job's terminal. The string must follow the rules for an RT11 command string, as described for .CSIGEN.

linbuf is the address where the original string (if accepted from the job's terminal) is to be stored. This area should be at least 81 bytes long. The string is terminated with a zero byte instead of a carriage-return/line-feed combination.

If this argument is omitted, the original string is not kept.

Errors

If an error occurs when input is accepted from the job's terminal, the user gets another try. An error message is printed on the terminal, along with another * prompt, and the directive waits for you to type another line.

If the string being examined is in memory and an error occurs, the carry bit is set, and one of the following error codes is returned in byte 52. The options and option-count are pushed from the stack.

Code	**Meaning**
0	Illegal command line (bad separators, illegal file name, command too long, and so forth).
1	A device specification was not recognizable or is not available on the system.

Example

The following code accepts a line of input from the terminal, checks it for RT11 command format, and stores the resulting device block information in a 39-word area called OUTSPC:

```
DEFEXT: .WORD     0,0,0,0          ;No default file types
OUTSPC: .BLKW     39.              ;Device block area
        .
        .
        .
        .CSISPC   #OUTSPC,#DEFEXT,#0
```

.DATE – Return Current Date to R0

The .DATE directive returns the current month, day, and year to R0 in the following format:

```
bit|13        10| 9            5| 4            0|
   +------------+---------------+---------------+
   |month(1-12)|  day (1-31)   | year-72(0-31)|
   +------------+---------------+---------------+
```

The year value in bits 4-0 is the current year minus 72.

Note that this directive gets its date information from the RSTS/E clock. Unlike the RT-11 operating system, you do not set the values returned with the DATE keyboard monitor command.

Privileges Required

None

Macro Call

```
   .DATE
```

R0 Format

```
        +----------+----------+
(R0) =  |    12    |    0     |
        +----------+----------+
```

Errors

No errors are returned with the .DATE directive.

Example

The following subroutine can be assembled separately and linked to a user program:

```
.TITLE  DATE.MAC
;
;CALLING SEQUENCE:
;
;       JSR       PC,DATE
;
;INPUT:  NONE
;
;OUTPUT: R0 = DAY (1-31)
;        R1 = MONTH (1-12)
;        R2 = YEAR - 72
;
        .MCALL    .DATE

DATE::  .DATE                         ;Get the system date
        MOV       R0,R2               ;Copy the date
        BIC       #^C37,R2
        ASR       R0                  ;Put month on a byte boundary
        ASR       R0                  ;
        MOV       R0,R1               ;Copy the date
        SWAB      R1                  ;Put month in the low byte
        BIC       #^C37,R1            ;Isolate the month
        ASR       R0                  ;Shift day to a byte boundary
        ASR       R0                  ;
        ASR       R0                  ;
        BIC       #^C37,R0            ;Isolate the day
        CLC                           ;Indicate no error
        RTS       PC

        .END
```

.DATTIM - Return Date or Time

The .DATTIM directive is available only to RT11 emulator users under RSTS/E; it is not available under the RT-11 operating system. .DATTIM converts a date or time from the RSTS/E system internal format (as returned by the .DATE directive, see Chapter 3) to an ASCII string suitable for printing. The format of the string is the same as that returned by the BASIC-PLUS DATE$ and TIME$ functions. That is, a date is in the form "27-Jun-84"; or, if the system uses the numeric date option, in the form "84.06.27". Similarly, a time is in the form "02:30 PM" or "14:30".

The .DATTIM directive assumes that R0 contains the address of the area to which the string is to be returned. It also expects to find the date or time in RSTS/E system internal format in the XRB. If a date is desired, it should be in XRB+0. If a time is desired, the word at XRB+0 should be cleared and the time stored in XRB+2. Note that these are the locations returned by the RSTS/E .DATE directive.

You can request a specific format for date or time rather than accepting the format selected at system generation. If you set bit 15 of XRB+0 (date) or XRB+2 (time), you request a specific format rather than the default selected at system generation. Bit 15 = 0 indicates the default. If bit 15 = 1, then bit 14 = 0 implies numeric format; bit 14 = 1 indicates alphanumeric format.

Upon return from the call, R0 contains the address of the first byte after the time or date string.

Privileges Required

None

Macro Call

```
    .DATTIM
```

Note that .DATTIM, like all RSTS/E-specific directives under RT11, is not expanded at assembly time. (It is not defined in the SYSMAC.SML file as a directive.) You can code a definition of the directive in a data section of your program or use .MACRO to define it. See the following example.

Errors

No errors are possible with the .DATTIM directive.

Example

The following program prints the current date and time at the job's terminal:

```
        .MCALL    .EXIT,.PRINT
        .PRIV = EMT+377           ;Prefix EMT to RSTS/E
        .DATTIM = EMT+361         ;Date/time EMT to RT11

        .CSECT

START:  .PRIV,.DATE               ;Get date from RSTS/E
        MOV       XRB+2,-(SP)     ;Save time
        MOV       DATE$,R0        ;Set addr for ret. string
        .DATTIM                   ;Get date string
        CLRB      (R0)+           ;Clear last byte for ASCIZ
        .PRINT    #DATMSG         ;Print date message
        CLR       XRB             ;Clear date from XRB
        MOV       (SP)+,XRB+2     ;Set time in XRB
        MOV       TIME$,R0        ;Set addr for time string
        .DATTIM                   ;Get time string
        CLRB      (R0)+           ;Make string ASCIZ
        .PRINT    #TIMMSG         ;Print time message
        .EXIT                     ;End program


DATMSG: .ASCII    /The date is /
DATE$:  .BLKW     12
TIMMSG: .ASCII    /The time is /
TIME$:  .BLKW     12

.END    START
```

.DELETE -- Delete File from Disk or DECtape

The .DELETE directive deletes a file from a disk or DECtape, deleting it from the device directory and releasing the space for other use. The file description given in the call includes a device, file name, and type. Set the PPN offset to include a PPN for the file, if desired (see Chapter 6). Otherwise, the PPN of the calling job is used.

Note that on RSTS/E systems, unlike the RT-11 operating system, it is possible to delete files which are open; no error is returned.

Privileges Required

Write access to the file being deleted (if disk). DEVICE privilege is required if the device is restricted.

Macro Call

.DELETE *area,chanum,devblk,seqnum*

where:

area	is the address of a three-word argument block.
chanum	is ignored on RSTS/E systems.
devblk	is the address of a four-word device block containing the device and file specification. See Chapter 6 for the format of this area. See also the PPN offset description for specifying a PPN.
seqnum	is ignored on RSTS/E systems.

Argument Block Format

```
+----------+----------+
|    0     |  chanum  |  <--(R0)
+----------+----------+
|       devblk        |
+----------+----------+
|       seqnum        |
+----------+----------+
```

Errors

Code	Meaning
1	The file named cannot be found.
2	The device specified is not a disk or DECtape.

Example

The following code deletes the file name FILNAM.TYP in the user's account on disk unit 1:

```
AREA:   .BLKW     3
DEVBLK: .RAD50    /DK1FILNAMTYP/
        .
        .
        .
        .DELETE   #AREA,#1,#DEVBLK
```

.DOCCL - Do a RSTS/E .CCL

This directive lets a program running under control of the RT11 run-time system examine a string to see if it is a valid RSTS/E Concise Command Language (CCL) command.

Note

> .DOCCL must be used by programs running under the RT11 run-time system, rather than the general monitor .CCL command.

If the string is a valid CCL command, the RT11 run-time system restores the low 512. bytes of memory to what the monitor expects and issues a general monitor .CCL command. (See Chapter 3 for a description of .CCL.)

The starting address of the string to be examined is passed in R0. The string itself must be in ASCIZ format (terminated with a zero byte).

Privileges Required

Execute access to the program that the CCL command executes.

Macro Call

```
  .DOCCL
```

Note that .DOCCL, like all RSTS/E-specific directives under RT11, is not expanded at assembly time. (It is not defined in the SYSMAC.SML file as a directive.) You can code a definition of the directive in a data section of your program or use .MACRO to define it. See the following example.

Errors

The errors returned for .DOCCL are the same (and in the same format) as for .CCL.

Example

The following code executes the CCL command PIP:

```
 .DOCCL = EMT+367
 STG:     .ASCIZ    /PIP/
          .
          .
          .
          MOV       #STG,R0
          .DOCCL
```

.DOFSS - Do a RSTS/E .FSS

The .DOFSS directive examines a string to see if it is a valid RSTS/E file specification. The RT11 emulator restores the low 512. bytes of memory to what the monitor expects and executes a .FSS directive.

Note

> You must use .DOFSS rather than .FSS for a program running under the RT11 run-time system.

The .DOFSS directive assumes that R0 contains the starting address of the string and that the string itself is in ASCIZ format (terminated with a zero byte). See the discussion of .FSS in Chapter 3 for an explanation of how the string is processed. The data returned on completion of .DOFSS is the same as for .FSS.

Note that .DOFSS, like all RSTS/E-specific directives under RT11, is not expanded at assembly time. (It is not defined in the SYSMAC.SML file as a directive.) You can code a definition of the directive in a data section of your program or use .MACRO to define it. See the following example.

Privileges Required

None

Macro Call

```
   .DOFSS
```

Errors

The errors for .DOFSS are the same (and in the same format) as for .FSS.

Example

The following code checks the string FILE.NAM to see if it is a valid RSTS/E file specification:

```
  .DOFSS = EMT+365
  STG:     .ASCIZ    /FILE.NAM/
           .
           .
           .
           MOV       #STG,R0
           .DOFSS
```

.DORUN - Chain to Non-RT11 RTS Program

Use the .DORUN directive to transfer control to programs that run under a non-RT11 run-time system.

Note

> The .DORUN directive must be used from programs running under the RT11 run-time system, rather than the .RUN directive, since RT11 keeps user logical information in a different area in low-core than the monitor expects.

The .DORUN directive causes the emulator to restore the user logical information to its normal place in the low 512. bytes of memory and translates an RT11 file specification to the proper form in the FIRQB for a .RUN.

The .DORUN assumes that R0 contains the address of a five-word file definition. The first four words contain a normal RT11 "device block" specification (see Chapter 6); the last word contains a parameter to be passed to the run-time system at FIRQB+FQNENT. If the address passed in R0 points to a device block within an *outspec* area returned by a previous .CSISPC directive, all of the RSTS/E-specific information from the file specification (PPN, and so forth) is loaded into the proper area in the FIRQB. Values in the first six words of the scratch pad area take precedence over the values found in the string examined by .CSISPC.

Privileges Required

Execute access to the specified program.

Macro Call

```
   .DORUN
```

Note that .DORUN, like all RSTS/E-specific directives under RT11, is not expanded at assembly time. (It is not defined in the SYSMAC.SML file as a directive.) You can code a definition of the directive in a data section of your program or use .MACRO to define it. See the following example.

Errors

Errors are returned as for a .RUN directive.

Example

The following program uses .CSISPC to scan a user-entered file specification, ensures that the program is entered with a zero parameter passed to the run-time system, and then attempts to execute the file with .DORUN. If an error occurs for either the .CSISPC or the .DORUN, an error message is printed at the job's terminal. The code follows:

```
.SETFQB = EMT+360                     ;Define EMT for SETFQB
.DORUN = EMT+363                      ;Define EMT for .DORUN

        .CSISPC   OUTSPC,DEFEXT,0 ;Read, scan command string
        BCS       ERROR               ;Command string in error
        MOV       OUTSPC,R0           ;Get devblk for first file
        CLR       4*2(R0)             ;Pass 0 at FIRQB+FQNENT
        .DORUN                        ;Try to run the file
ERROR:  .PRINT    #ERRMES             ;An error occurred
        .EXIT

ERRMES: .ASCIZ /?Can't run that file/
```

.DSTATUS - Return Device Status

The .DSTATUS directive returns information about a device to a four-word area defined in the user program.

Privileges Required

None

Macro Call

.DSTATUS *stablk,devnam*

where:

stablk is the address of a four-word area to contain the returned status information.

devnam is the address of one word containing a device and unit number, in RAD50 format. The word can also be a three-character user logical name in RAD50 format.

Information Returned to Statblk

```
+------------------------------------+
|     status word (see format)       |
+------------------------------------+
|   handler size (= 0 for RSTS/E)    |
+------------------------------------+
| load address (=160000 for RSTS/E)  |
+------------------------------------+
|    device size (= 0 for RSTS/E)    |
+------------------------------------+
```

Status Word Format

```
+---------------+---------------+
|     flags     | handler index |
+---------------+---------------+
```

The low byte contains the RT11 handler index for the device:

Octal Value	Device
0	(disk)
1	DT (DECtape)
3	LP (line printer)
4	KB (keyboard)
7	PR (paper tape reader)
10	PP (paper tape punch)
11	MT (magnetic tape)
14	CR (card reader)

The high byte describes the device characteristics. Note that you can combine the device-characteristics flags. For example, a value of 140000 for the entire word would indicate a file-structured, read-only disk. The word values are:

Device Characteristics (Octal Word Value)	Meaning
100000	Device is file-structured
040000	Device is read-only
020000	Device is write-only

Errors

Code	Meaning
0	The device named in the *devnam* argument is not known on the system.

Example

The following code checks for status information for MT0:

```
STAT:   .BLKW     4
DEVNAM: .RAD50    /MT0/
        .
        .
        .
        .DSTATUS  #STAT,#DEVNAM
```

.ENTER - Open File for Output

The .ENTER directive creates a new file and associates the file with a channel number. You then use the channel number to refer to the file for other I/O operations.

A RSTS/E "tentative open" is done; that is, the file is not permanent until a .CLOSE is executed with the same channel number as specified in the .ENTER directive. Then, the tentative file is made permanent, and any already existing file of the same name on the same device is deleted. A .PURGE, .HRESET, or .SRESET directive deletes the tentative file; any already existing file of the same name on the same device is left intact.

You can preallocate space for disk files by setting an argument in the call. Note that you can also set a PPN, protection code, mode, and other RSTS/E parameters applicable to opening a file for output by setting words in the scratch pad area (see Chapter 6).

Privileges Required

Write access to the file.

Macro Call

.ENTER *area,chanum,devblk,len,seqnum*

where:

area	is the address of a four-word argument block.
chanum	is a channel number in the range 0 to 17 (octal). See Chapter 6 for rules on channel numbers.
devblk	is the address of a four-word device-block containing the device and file specifications. See Chapter 6 for the format of this area.
len	is the file size to be preallocated for the file, in 512-byte blocks (for disk files only). Values can range from 1 to 32767. For large disk files on RSTS/E, you must set the Most Significant Bits (MSB) byte in the scratch pad area (see Chapter 6). RSTS/E ignores a zero or negative value.
seqnum	is ignored on RSTS/E systems.

Argument Block Format

```
+----------+----------+
|    2     |  chanum  |  <--(R0)
+----------+----------+
|       devblk        |
+----------+----------+
|         len         |
+----------+----------+
|       seqnum        |
+----------+----------+
```

Errors

Code	Meaning
0	Channel is already in use.
1	The file cannot be preallocated to the specified size, or the device is full.

Example

The following code opens the file NEWFIL.SAV on MT0 for output:

```
AREA:   .BLKW    4
DEVBLK: .RAD50   /MT0NEWFILSAV/
          .
          .
          .
        .ENTER   #AREA,#1,#DEVBLK,#0,#0
```

.ERRPRT - Print RSTS/E Error Message

The .ERRPRT directive prints the RSTS/E error message text corresponding to an error code on the user terminal. The call assumes that the error code is in R0. The text is printed without a carriage-return/line-feed combination.

Privileges Required

None

Macro Call

```
    .ERRPRT
```

Note that .ERRPRT, like all RSTS/E-specific directives under RT11, is not expanded at assembly time. (It is not defined in the SYSMAC.SML file as a directive.) You can code a definition of the directive in a data section of your program or use .MACRO to define it. See the following example.

Errors

No errors are possible with the .ERRPRT directive.

Example

```
          .MCALL    .EXIT,.PRINT,.REGDEF
          .REGDEF

  .ERRPRT = EMT+364

  .CSECT

  START:  MOV       256.,R1        ;Number of errors to print
          CLR       R0             ;Start with error zero
  10$:    .ERRPRT                  ;Print error text
          MOV       R0,-(SP)       ;Save R0
          .PRINT    #CRLF          ;Print a CRLF
          MOV       (SP)+,R0       ;Restore R0
          INC       R0             ;Inc R0 for next message
          SOB       R1,10$         ;Loop until all are printed
          .EXIT
  CRLF:   .BYTE     0
  .END    START
```

.EXIT -- Program Exit

The .EXIT directive terminates execution of the calling program. On exiting from the calling job, the RSTS/E monitor drops any temporary privileges still in effect. Then, control passes to the job keyboard monitor at the keyboard monitor entry point (see P.NEW in Chapter 2).

Note

> The job keyboard monitor is the default keyboard monitor unless the job has issued the SET JOB/KEYBOARD_MONITOR command or has issued a general monitor .RTS directive (see Chapter 3) to specify a job keyboard monitor.

If R0 = 0 when the .EXIT is executed, any channels still open will be released, without the usual clean-up operations. In particular, any temporary files (opened with .ENTER and not yet closed) will be deleted.

Note

> Programs running under the RT11 run-time system must use the RT11 .EXIT directive, not the general monitor .EXIT or .RTS directive.

Privileges Required

None

Macro Call

```
   .EXIT
```

Errors

No errors are possible with the .EXIT directive.

Example

The following code exits such that files opened with .ENTER and not yet closed will be deleted:

```
    SAVOLD: MOV       #0,R0
            .EXIT
```

.FETCH -- Check Whether Device Exists

On RSTS/E systems, the .FETCH directive checks to see if a specified device exists on the system.

Privileges Required

None

Macro Call

.FETCH *addr,devnam*

where:

addr	is ignored on RSTS/E.
devnam	is the address of a one-word, RAD50 device name. The format of this word is the same as that described for the first word of a device block (see Chapter 6).

Errors

Code	Meaning
0	The device specified does not exist on the system.

Example

The following code checks to see if LP0 exists on the system:

```
DEVNAM: .RAD50    /LP0/
        .
        .
        .
        .FETCH    #0,#DEVNAM
```

.GETCOR - Changes Job Image Size

This directive changes the amount of space currently allocated for the user job image. The RT11 emulator checks to ensure that the requested expansion will not destroy the scratch pad area, moves the scratch pad area if necessary, and then expands the job image size using the .CORE directive (see Chapter 3). Thus, the same rules concerning size changes for .CORE also apply to .GETCOR.

The .GETCOR directive assumes that the new job image size, in K words, is in R0. If the required expansion is not possible, the carry bit will be set on return from the call. Otherwise, the size change succeeds, and word 44 now points to the new location for the start of the scratch pad area.

Note

> Programs running under the RT11 run-time system must use .GETCOR, not the general monitor .CORE directive.
>
> You must use .GETCOR to expand the job image size before you use .CHAIN to transfer control to a program that requires more memory than the current program.

Privileges Required

EXQTA privilege is required to exceed the job's memory limit.

Macro Call

```
   .GETCOR
```

Note that .GETCOR, like all RSTS/E-specific directives under RT11, is not expanded at assembly time. (It is not defined in the SYSMAC.SML file as a directive.) You can code a definition of the directive in a data section of your program or use .MACRO to define it. See the following example.

Errors

If the requested change cannot be made, the carry bit is set on return from the call.

Example

The following code uses .GETCOR to expand the user job image area to 24K words before doing a .CHAIN:

```
  .GETCOR = EMT+366
              .
              .
              .
            MOV     #24.,R0
            .GETCOR
            BCS     ERRTN
            .CHAIN
```

.GTLIN - Get Line from Job's Terminal

The .GTLIN directive accepts a line of input from either the job's terminal or core common and stores it in a buffer for later examination by the user program. You can also specify an ASCIZ string to be displayed at the terminal before the line is accepted.

.GTLIN reads the line from core common instead of the job's terminal if the program is invoked with a parameter word of 8192. For example, the parameter word normally has this value when a program is run by a CCL command. A second .GTLIN causes the program to exit.

By default, .GTLIN returns a string in uppercase. To obtain lowercase letters, set bit 14 in the Job Status Word (JSW) before you issue the .GTLIN.

Privileges Required

None

Macro Call

.GTLIN *linbuf*[,*prompt*]

where:

linbuf is the address of an area to receive the input line. Up to 81 bytes can be accepted from a terminal, so the buffer should be at least that long. The line is stored in memory and terminated with a zero byte instead of with a carriage-return/line-feed combination.

prompt is the address of a string to be printed on the job's terminal before input is accepted. The string itself must be terminated with a zero byte or a 200 (octal) byte. If the string is terminated with a zero byte, the display ends with a carriage-return/line feed. If terminated with a 200 (octal) byte, the display ends after the prompt string.

Errors

No errors are possible with the .GTLIN request.

Example

The following example prompts the job's terminal and accepts a line of input:

```
PROMPT: .ASCIZ    /YES OR NO?/
LINBUF: .BLKB     81.
        .
        .
        .
        .GTLIN    #LINBUF,#PROMPT
```

.GVAL - Get Value from Scratch Pad

The .GVAL directive returns the contents of a word in the scratch pad area to R0. You specify an offset from the beginning of the scratch pad.

Privileges Required

None

Macro Call

```
   .GVAL area,offset
```

where:

area	is the address of a two-word argument block.
offset	is the offset, in bytes, from the beginning of the scratch pad area of the word desired. The offset value must be even. An odd offset causes a trap to kernel mode vector at four.

Argument Block Format

```
+----------+----------+
|    34    |     0    |  <--(R0)
+----------+----------+
|       offset        |
+----------+----------+
```

Errors

Code	Meaning
0	The offset given is beyond the limits of the scratch pad area.

Example

The following example returns the contents of the PROTEC offset in the scratch pad area to R0:

```
  AREA:   .BLKW     2
            .
            .
            .
          .GVAL     #AREA,#2
```

.HRESET -- Hardware Reset

The .HRESET (hardware reset) directive closes all open channels for the program without performing any of the normal clean-up operations (except to disassociate the RT11 channel numbers from their RSTS/E counterparts). For example, no trailer labels are written to magnetic tape, no form feed is given for the line printer, and so forth. Temporary files (created with .ENTER but not yet closed) are deleted.

Privileges Required

None

Macro Call

```
   .HRESET
```

Errors

No errors are possible with the .HRESET directive.

Example

```
  .HRESET
```

.LOOKUP - Open File for Input

The .LOOKUP directive opens an existing file and associates it with a channel number; in RSTS/E terms, an OPEN FOR INPUT statement. The channel used is then busy until one of the following directives is executed:

.CLOSE
.SAVESTATUS
.SRESET
.HRESET
.PURGE
.CSIGEN (if the channel number is in the range 0 to 8)

Note that you can set a PPN applicable to a RSTS/E OPEN FOR INPUT statement by setting the appropriate words in the scratch pad area before executing the .LOOKUP (see Chapter 6).

Privileges Required

For disk files, you need read access. If the device is restricted, you need DEVICE privilege. You need RDNFS privilege to access disks in non-file-structured mode.

Macro Call

```
.LOOKUP area,chanum,devblk,seqnum
```

where:

area is the address of a three-word argument block.

chanum is a channel number in the range 0 to 17 (octal). See Chapter 6 for rules on channel numbers.

devblk is the address of a four-word device block containing the device and file specifications. See Chapter 6 for general rules on the format of device blocks.

The file name portion of the device block is ignored for non-file-structured devices, such as a paper tape. If you want to do non-file-structured I/O on a file-structured device, clear the first word of the file name portion of the device block.

seqnum is ignored on RSTS/E systems.

Argument Block Format

```
+----------+----------+
|     1    |  chanum  |  <--(R0)
+----------+----------+
|       devblk        |
+----------+----------+
|       seqnum        |
+----------+----------+
```

Errors

Code	Meaning
0	Channel already open.
1	The file requested (in the device block) was not found.

Example

```
ERRBYT=42
ARGBLK: .BLKW     5                 ;Leave space for arg block
DEVBLK: .RAD50    /DT3/             ;Define device,
        .RAD50    /DATA  001/       ;Filename and type
           .
           .
           .
        .LOOKUP   #ARGBLK,#7,#DEVBLK ;Open on channel 7
        BCC       LDONE             ;File was found
        TSTB      @#ERRBYT          ;Error--test further below
        BNE       NFD               ;File not found
        .PRINT    #CAMSG            ;Print 'Channel Active'
        .EXIT
NFD:    .PRINT    #NFMSG            ;PRINT 'File not found'
        .EXIT
CAMSG:  .ASCIZ    /Channel active/
NFMSG:  .ASCIZ    /File not found/
```

.PRINT - Display String on Job's Terminal

The .PRINT directive displays a specified ASCII string on the job's terminal. The line can be terminated with or without a carriage-return/line-feed combination. Note that on RSTS/E systems, if the terminal is detached when a .PRINT is issued, the job hibernates. That is, execution of the job is suspended until the terminal is reattached to the job.

Privileges Required

None

Macro Call

```
.PRINT stradd
```

where:

stradd is the starting address of the string to be printed. The string itself must be terminated with either a zero byte or a byte value of 200. If the string is terminated with a zero byte, the display ends with a carriage-return/line-feed combination. If the string is terminated with a 200 byte, the display ends with the string.

Errors

No errors are possible with the .PRINT directive.

Example

```
S1:     .ASCIZ    /String will have CR-LF following/
S2:     .ASCII    /String will have no CR-LF following/
        .BYTE     200
        .EVEN
           .
           .
           .
        .PRINT    #S1
        .PRINT    #S2
```

.PURGE - Release Channel

The .PURGE directive can be used to release a channel when it is not desirable to close the file normally. Any temporary file created by a .ENTER is discarded. A .PURGE on a file that was opened with .LOOKUP will cause no action other than to free the channel; any data written to the file will still be there. If the specified channel is not open, the .PURGE is simply ignored; no error is returned.

Privileges Required

None

Macro Call

```
   .PURGE chanum
```

where:

chanum is a channel number in the range 0 to 17 (octal). See Chapter 6 for rules on channel numbers.

R0 Format

```
       +----------+----------+
(R0) = |     3    |  chanum  |
       +----------+----------+
```

Errors

No errors are possible with the .PURGE directive.

Example

The following example verifies that channels 0 to 7 are free:

```
  .TITLE   .PURGE.MAC
           .MCALL     .PURGE,.EXIT
  START:
           CLR        R1              ;Start with channel 0
  1$       .PURGE     R1              ;Purge a channel
           INC        R1              ;Bump to next channel
           CMP        R1,#8.          ;Is it at channel 8 yet
           BLO        1$
           .EXIT
           .END       START
```

.RCTRLO -- Reset CTRL/O

The .RCTRLO directive restarts any programmed output to the user's terminal after such output has been stopped by the user-entered CTRL/O or CTRL/C.

Terminal service on RSTS/E maintains a "discard all program output" indicator for each terminal on the system. When this indicator is set, the driver ignores any output requests for that terminal. When the indicator is clear, output to the terminal proceeds normally. The .RCTRLO directive clears this indicator.

The driver sets the indicator when the user enters a CTRL/C or CTRL/O combination and reverses it when the user enters a CTRL/O again.

Thus, when it is important that a message get through to the terminal, use a .RCTRLO directive before the .PRINT or other request for output to make sure that the message is displayed regardless of any user-entered CTRL/O or CTRL/C.

Privileges Required

None

Macro Call

```
   .RCTRLO
```

Errors

No errors are possible with the .RCTRLO directive.

Example

The following example forces the message to be displayed on the user's terminal:

```
  EOW:    .ASCIZ    /The end of the world is nigh/
              .
              .
              .
          .RCTRLO
          .PRINT    #EOW
```

.READ/.READW/.READC - Read Data

On RSTS/E systems, .READ, .READW, and .READC transfer data from a file or device previously opened on a channel to a user buffer. Once the transfer is complete, control returns to the user program inline, for .READ and .READW, or to a user-specified completion routine, for .READC. You specify the maximum number of words to be transferred; the actual number of words transferred is returned in R0.

Note that you can set a modifier word in the scratch pad area for devices that use this information. See the .READ directive in Chapter 3 for possible values of the modifier word.

Privileges Required

None

Macro Call

$$\left\{\begin{matrix}\texttt{.READ} \\ \texttt{.READW}\end{matrix}\right\} \; \textit{area,chanum,bufadd,wrdcnt,blknum}$$

or

.READC *area,chanum,bufadd,wrdcnt,cmprtn,blknum*

where:

area	is the address of a five-word argument block.
chanum	is a channel number in the range 0 to 17 (octal), previously defined in an open (.LOOKUP).
bufadd	is the address of a buffer to contain the data to be read.
wrdcnt	is the maximum number of words to be read. The actual number of words read may be less than *wrdcnt*; it is never more. The actual number of words transferred is returned in R0.
cmprtn	is the address of a completion routine to which control is transferred when the read is complete. (For .READC only.)
blknum	is the block number relative to the start of the file where the read is to begin (for files opened with file-structured .LOOKUP). To access large files on RSTS/E systems, you must set the word at offset 2 in the scratch pad area; see Chapter 6.

For devices opened with non-file-structured .LOOKUP, blknum is the absolute block number or device cluster number where the read is to begin.

For successive reads, your program must increment *blknum*.

Argument Block Format

```
+----------+----------+
|    10    |  chanum  |
+----------+----------+
|       blknum        |
+----------+----------+
|       bufadd        |
+----------+----------+
|       wrdcnt        |
+----------+----------+
|       cmprtn*       |
+----------+----------+
 *(The last word = 0 for .READW, or 1 for .READ.)
```

Errors

Code	**Meaning**
0	Attempt to read past end-of-file (EOF).
1	Hard error occurred on channel.
2	Channel is not open.

Example

The following code reads data from the device open on channel 7 to a 512. byte user buffer:

```
 AREA:   .BLKW     5
 BUF:    .BLKW     512.
            .
            .
            .
         .READ     #AREA,#7,#BUF,#512.,#0
```

.RENAME - Rename a File

The .RENAME directive changes the name of a file on disk or DECtape and gives that file the current date in its directory entry. If a file with the new name already exists on the specified device, it is deleted.

Privileges Required

If the device is restricted, you need DEVICE privilege. For disk files, you need create/rename access to the file.

Macro Call

.RENAME *area,chanum,namblk*

where:

area is the address of a two-word argument block.

chanum is ignored on RSTS/E systems.

namblk is the address of an eight-word area containing two four-word device and file descriptor blocks (see Chapter 6). The first four-word block is the old file name and the device where it is stored. The second four-word block is the new file name; the device descriptor must be specified, and must be the same as the device descriptor in the first four-word block. If a PPN is specified in the first six words of the scratch pad, it is used for both the old and the new file. If a protection code is given in the first six words of the scratch pad, it is used for the new file. Other values in the first six words of the scratch pad are ignored.

Argument Block Format

```
+----------+----------+
|     4    |  chanum  |  <--(R0)
+----------+----------+
|       devblk        |
+----------+----------+
```

Errors

Code	Meaning
1	No file with the "old" name was found on the device.
2	The device specified is not a disk or DECtape.

Example

The following code renames file OLDFIL.MAC to NEWFIL.MAC:

```
AREA:   .BLKW      2
NAMBLK: .RAD50     /DT3/
        .RAD50     /OLDFIL/
        .RAD50     /MAC/
        .RAD50     /DT3/
        .RAD50     /NEWFIL/
        .RAD50     /MAC/
             .
             .
             .
        .RENAME    #AREA,#0,#NAMBLK
```

.REOPEN - Reopen File Closed with .SAVESTATUS

The .REOPEN directive reopens a file that was closed with a .SAVESTATUS directive. You can use a different channel number than was used in the original .LOOKUP and .SAVESTATUS, as long as the new channel is not already open.

The .SAVESTATUS and .REOPEN directives are easier to use than .CLOSE and .LOOKUP, if your program needs to open more than 15 files during its execution. The .SAVESTATUS directive retains the information necessary to reopen a file, including the PPN. Thus, you do not need to reset the PPN offset in the scratch pad area.

Privileges Required

If the device is restricted, you need DEVICE privilege. You need RDNFS privilege to access a non-file-structured disk. You need read access to the file.

Macro Call

```
    .REOPEN area,chanum,chnblk
```

where:

area	is the address of a two-word argument block.
chanum	is a free channel number in the range 0 to 17 (octal). See Chapter 6 for rules on channel numbers.
chnblk	is the address of the five-word block where the channel status information was stored (see .SAVESTATUS).

Argument Block Format

```
+----------+----------+
|    6     |  chanum  |  <--(R0)
+----------+----------+
|       chnblk        |
+----------+----------+
```

Errors

Code	Meaning
0	The specified channel is in use. The .REOPEN has not been done.

Example

The following code reopens a file closed with .SAVESTATUS using the device block area named SAVBLK:

```
AREA:   .BLKW     2
SAVBLK: .BLKW     5
           .
           .
           .
        .REOPEN   #AREA,#6,#SAVBLK
```

.SAVESTATUS - Save Status of File for Later .REOPEN

The .SAVESTATUS directive closes a disk or DECtape file, saving all the information needed to reopen the file later in execution of the program (see .REOPEN). The information saved is stored in a five-word area in the user program and includes the PPN for the file. Thus, the .SAVESTATUS/.REOPEN combination is easier to use on RSTS/E systems than .CLOSE/.LOOKUP in this case; you do not have to set the PPN offset in the scratch pad area.

The .SAVESTATUS directive can be used only if the file was opened with .LOOKUP. If the file was opened with .ENTER, a .SAVESTATUS on that channel returns an error.

Privileges Required

None

Macro Call

.SAVESTATUS *area,chanum,chnblk*

where:

area •	is the address of a two-word argument block.
chanum	is a channel number in the range 0 to 17 (octal), previously used in a .LOOKUP directive for disk or DECtape.
chnblk	is the address of a five-word area to store the current information for the file. On RSTS/E systems, the first four words of this area are set to the device, file name, and type, in the standard RT11 device-block format (see Chapter 6). The last word of this area is set to the PPN of the file.

Argument Block Format

```
+---------+---------+
|    5    | chanum  |
+---------+---------+
|      chnblk       |
+---------+---------+
```

Errors

Code	Meaning
0	The channel specified is not currently open; that is, a previous .LOOKUP on the channel was never done.
1	The file was opened with .ENTER, or else the file is not on a disk or DECtape device. The .SAVESTATUS is illegal.

Example

The following code saves the status of the file open on channel 12 for a later reopen:

```
AREA:   .BLKW     2
SAVBLK: .BLKW     5
           .
           .
           .
        .SAVEST   #AREA,#12.,#SAVBLK
```

.SCCA - Pass CTRL/Z to User Program

The .SCCA directive lets a user program recognize and process a CTRL/Z combination entered on the job's terminal. When called with any nonzero value for the *addr* argument, subsequent CTRL/Zs entered at the job's terminal are translated to an ASCII 3 and passed like any other character to the program when it issues a .TTYIN or .GTLIN. (Normally, a CTRL/Z causes the RT11 emulator to exit the running program when a .READ or .TTYIN is done.)

Note

> The operation of the .SCCA directive is quite different under RSTS/E than under the RT-11 operating system, where it lets a user program trap CTRL/C combinations.
>
> The .SETCC directive must be used for this function under the RT11 run-time system under RSTS/E. The intent is to let you code an .SCCA directive to trap the delimiter equivalent in function on the two systems.

On RSTS/E systems, CTRL/Z is the "type-ahead terminator." You can, for example, type RUN $MACRO, type the command line for the assembly, enter a CTRL/Z, and then type RUN $LINK. You do not have to wait for the assembly to finish before typing the command to link.

On RT-11 systems, CTRL/C is the "type-ahead terminator." Hence, the .SCCA directive processes the character appropriate to the system on both RSTS/E and RT-11 systems.

Privileges Required

None

Macro Call

.SCCA *area,addr*

where:

area	is the address of a two-word argument block.
addr	is a switch. Any nonzero value for this argument causes CTRL/Zs to be passed to the user program, as described previously. A zero value returns CTRL/Z processing to the emulator.

Argument Block Format

```
+----------+----------+
|    35    |     0    |
+----------+----------+
|        addr         |
+----------+----------+
```

Errors

No errors are possible with the .SCCA directive.

Example

The following code causes CTRL/Zs to be passed to the program:

```
  AREA:    .BLKW      2
               .
               .
               .
           .SCCA      #AREA,#1
```

.SETCC - Process CTRL/C

The .SETCC directive defines a routine to be entered when the user enters one CTRL/C on the job's terminal. (Two consecutive CTRL/Cs still aborts the running program.) The address of the routine to be entered is assumed to be in R0 when the .SETCC is executed.

The .SETCC is only effective until the first CTRL/C; you must reissue the .SETCC from the trap routine if you want it to remain in effect. You must also reenable CTRL/O processing with .RCTRLO. Use an RTI instruction to exit from the trap routine. See the discussion of offset 66 in the scratch pad area in Chapter 6 for further information on CTRL/C processing under the RT11 run-time system.

Privileges Required

None

Macro Call

```
   .SETCC
```

Note that .SETCC, like all RSTS/E-specific directives under RT11, is not expanded at assembly time. (It is not defined in the SYSMAC.SML file as a directive.) You can code a definition of the directive in a data section of your program or use .MACRO to define it. See the following example.

Errors

No errors are possible with the .SETCC directive.

Example

The following code defines a routine to be entered when the user enters a CTRL/C on the job's terminal:

```
.SETCC = EMT+362
MESAGE: .ASCIZ    "You typed a CTRL/C!"
           .
           .
           .
        MOV       #CTLC,R0
        .SETCC
           .
           .
           .
CTLC:   .RCTRLO
        .PRINT    #MESAGE
        MOV       #CTLC,R0
        .SETCC
        RTI
```

.SETFQB - Set Up FIRQB

The .SETFQB directive can be used after a .CSISPC; the two calls combined return information to the FIRQB and XRB as if the file specification string had been examined by .FSS.

The call assumes that R0 contains the address of the first word of a device block returned by .CSISPC. As noted in the discussion of .CSISPC, the device-block information must not have been moved; thus, the address in R0 must be the address of the first word of a device block within the "outspc" area returned by the last .CSISPC.

Note that any information in the first six words of the scratch pad area overrides any RSTS/E-specific information returned by the .CSISPC. In addition, the .SETFQB clears the first six words of the scratch pad after using the information.

Privileges Required

None

Macro Call

```
   .SETFQB
```

Note that .SETFQB, like all RSTS/E-specific directives under RT11, is not expanded at assembly time. (It is not defined in the SYSMAC.SML file as a directive.) You can code a definition of the directive in a data section of your program or use .MACRO to define it. See the following example.

Errors

No errors are possible with the .SETFQB directive.

Example

The following code returns information to the FIRQB and XRB about the file following the equal sign (=) in an RT command:

```
  .SETFQB = EMT+360
                .
                .
                .
            .CSISPC  #OUT,#EXT,#CMD
            MOV      #OUT+<3*5*2>,R0
            .SETFQB
```

.SETTOP - Expand to Start of Scratch Pad

On RSTS/E systems, the .SETTOP directive sets the contents of the word at location 50 in the low 512 bytes of memory to the value specified in .SETTOP. The value itself cannot equal or exceed the starting address of the scratch pad area (see Chapter 6). If such a value is specified, .SETTOP simply sets the contents of location 50 to the address of the first word after the starting address of the scratch pad. The value actually set in word 50 is also returned to R0 on completion of the call.

Privileges Required

None

Macro Call

```
    .SETTOP adrval
```

where:

adrval is a value specifying the address to which location 40 is to be changed.

Errors

No errors are possible with the .SETTOP directive.

Example

The following code sets the contents of location 50 to the address of the word after the start of the scratch pad:

```
   .SETTOP #-2
```

.SFPA - Set Floating-Point Error Address

The .SFPA directive indicates that you want to handle floating-point errors yourself. You specify the address of a routine to be entered if such errors occur. The .SFPA works for errors on both the FIS (floating-point instruction set on the PDP-11/40) and the FPP or FPU (hardware floating-point units on the PDP-11/45 and 11/70 asynchronous unit and its equivalents). In the case of FPP or FPU hardware, two words have been pushed on the stack: the Floating-Point Exception Code (FEC) at the top of the stack and the Floating Error Address (FEA) one word down. Your routine can process these words in any way it sees fit and should pop them from the stack before issuing an RTI instruction to return control to the program where it left off when the errors occurred.

The .SFPA directive works only for one floating-point error trap. You must reexecute the .SFPA from your error-processing routine to ensure that successive errors are handled in the same way. You can disable traps to your error routine by issuing the .SFPA directive with a zero address argument.

Note

> You should not use .SFPA in a routine called by a FORTRAN program, because programs compiled under FORTRAN have their own FPP/FIS exception handling routines and expect to find the FPU status as they have set it.

Privileges Required

None

Macro Call

.SFPA *area,addr*

where:

area	is the address of a two-word argument block.
addr	is the address of the routine to be entered when a floating-point error occurs.

Argument Block Format

```
+--------+--------+
|   30   |    0   |
+--------+--------+
|      addr       |
+--------+--------+
```

Errors

No errors are possible with the .SFPA directive.

Example

The following code defines ERRTN as the address of the error routine to be entered when a floating-point error occurs:

```
AREA:   .BLKW     2
          .
          .
          .
        .SFPA     #AREA,#ERRTN
```

.SPFUN - Special Functions for I/O

The .SPFUN directive handles special functions for disk, terminal, magnetic tape, and flexible diskette.

Privileges Required

None

Macro Call

.SPFUN *area,chanum,func,buf*[*,wcnt*][*,blk*][*,crtn*]

where:

area is the address of a six-word argument block.

chanum is a channel number in the range 0 to 17 (octal), defining the device on which the special function is to be performed. The channel must have been previously opened with .LOOKUP.

func is a special function code. To request magnetic tape special functions emulating RT11's .SPFUN, set this argument to one of the following values:

377	Write end-of-file
376	Forward 1 record
375	Backspace 1 record
373	Rewind to load point
372	Offline rewind

Any other value for this argument is used as the function code (at XRB+XRLEN) in a RSTS/E .SPEC directive (see Chapter 3).

buf is used as the value of the word at XRB+XRLOC in a RSTS/E .SPEC directive. It must be set to zero if the *func* argument is 377, 376, 375, 373, or 372.

wcnt is used as the value of the word at XRB+XRBC in a RSTS/E .SPEC directive. It is ignored (and can be omitted) if the *func* argument is 377, 376, 375, 373, or 372.

blk is used as the value of the word at XRB+XRBLK in a RSTS/E .SPEC directive. It is ignored (and can be omitted) if the *func* argument is 377, 376, 375, 373, or 372.

crtn is the address of a completion routine to which control is to be transferred when the operation is complete. If this argument is omitted, control returns to the instruction following the .SPFUN when the operation is completed.

Note that if you are using .SPFUN to do a RSTS/E .SPEC, the RT11 emulator fills in the RSTS/E channel number and device handler values in the XRB for you.

Argument Block Format

```
+----------+----------+
|    32    |  chanum  |
+----------+----------+
|         blk         |
+----------+----------+
|         buf         |
+----------+----------+
|         wcnt        |
+----------+----------+
|   func   |   377    |
+----------+----------+
|         crtn        |
+----------+----------+
```

Errors

Code	Meaning
0	Tried to read past end-of-file.
1	Hard error occurred on channel.
2	Channel is not open.

Example

The following code rewinds a magnetic tape (open on channel 4):

```
  AREA:   .BLKW     6
            .
            .
            .
          .SPFUN    #AREA,#4,#373,#0
```

.SRESET - Software Reset

On RSTS/E systems, the .SRESET directive does the same thing as a hardware reset (.HRESET). It closes all open channels for the program without performing any of the normal clean-up operations (except to disassociate the RT11 channel numbers from their RSTS/E counterparts). For example, no trailer labels are written to magnetic tape, no form feed is given for the line printer, and so forth. Tentative files (created with .ENTER but not yet closed) are deleted.

Privileges Required

None

Macro Call

```
   .SRESET
```

Errors

No errors are possible with the .SRESET directive.

Example

```
  .SRESET
```

.TRPSET - Intercept Traps to 4 and 10

The .TRPSET directive allows the user program to trap to a user routine on errors that normally cause the program to abort execution. Control passes to the address specified in the call when the job tries to execute a reserved instruction (normally trapped to kernel mode address 10) or issues an instruction with an odd address (normally trapped to kernel mode address 4).

The carry bit indicates which error occurred. When control is passed to the error-handling routine, the carry bit is zero if the error was an odd-address error. If the carry bit is one, the error was a reserved-instruction error. An RTI instruction will return control to the user program from the trap routine.

Note that the .TRPSET directive affects only the next error causing such a trap. You must reissue the .TRPSET from your error-handling routine if you want to handle such errors consistently.

Privileges Required

None

Macro Call

```
   .TRPSET area,addr
```

where:

area is the address of a two-word argument block.

addr is the address to which control is to be passed when a reserved-instruction or odd-address error occurs.

Argument Block Format

```
+----------+----------+
|     3    |     0    |
+----------+----------+
|        addr         |
+----------+----------+
```

Errors

No errors are possible with the .TRPSET directive.

Example

The following code defines TRPRTN as the address of the routine to be entered when a reserved-instruction or odd-address error occurs:

```
AREA:    .BLKW     2
            .
            .
            .
         .TRPSET   #AREA,#TRPRTN
            .
            .
            .
TRPRTN: BCC        ODDADR
        (code for reserved instruction)
            .
ODDADR: (code for odd address)
```

.TTYIN/.TTINR - One-Character Read From Terminal

The .TTYIN and .TTINR directives both accept input from the job's terminal and transfer one character to the low byte of R0.

Bits 6 and 12 of the job status word (JSW) affect how the transfer is done.

If bit 12 of the JSW = 0, the .TTYIN/.TTINR waits until a whole line has been typed at the job's terminal and then transfers one character to the low byte of R0. Succeeding .TTYIN/.TTINR requests will then transfer remaining characters, including carriage return and line feed. In addition, the usual monitor processing for terminal input occurs: all characters typed are echoed at the terminal, and CTRL/U and DELETE delete a line or a character, respectively.

If bit 12 of the JSW = 1, ODT-mode input is done. That is, characters are transferred when they are available, rather than waiting for a whole line to be typed before passing a character to the user program. One .TTYIN/.TTINR transfers one character. Furthermore, characters typed are not echoed at the terminal, and CTRL/U and DELETE are simply passed on to the user program. (CTRL/C and CTRL/O are processed as usual, however. CTRL/S and CTRL/Q are processed, if that option was requested for your terminal with TTYSET.) No ALTMODE conversion is done.

If bit 6 = 1, control returns to the user program with the carry bit set if no character can be returned to the program. Note that this should only be used with .TTINR, and only when the program needs to perform other processing while waiting for terminal input. If bit 6 = 0, control does not return to the user program until a character can be passed to the program.

If used, bits 6 and 12 in the JSW must be set by the user program; they are cleared automatically when a program terminates.

Note

> Single character I/O places heavy demand on RSTS/E system resources and can degrade system performance. These calls are provided only for compatibility with RT-11.

.TTYIN
.TTINR

Privileges Required

None

Macro Calls

```
   .TTYIN [addr]
```

or

```
   .TTINR
```

where:

addr	is the address of a location where the character is to be stored, in addition to being stored in R0. If the address is omitted, the character is stored only in R0.

Errors

Code	Meaning
0	No character or characters are available in the monitor buffer for terminal I/O (only applies to .TTINR).

Example

The following code transfers one character from the job's terminal to the low byte of R0:

```
  .TTYIN
```

.TTYOUT/.TTOUTR - Transfer One Character to Job's Terminal

The .TTYOUT and .TTOUTR directives both transfer one character from the low byte of R0 to the job's terminal. Using .TTYOUT or .TTOUTR is slower than using .PRINT because they require more system overhead. Use .PRINT whenever possible.

Note that on RSTS/E systems, if the terminal is detached when a .TTYOUT or .TTOUTR is executed, the job hibernates. That is, execution of the job is suspended until the terminal is reattached to the job.

Privileges Required

None

Macro Call

```
   .TTYOUT [addr]
```

or

```
   .TTOUTR
```

where:

addr is the address of a memory location containing the character to be printed. The character is loaded into R0 first. When control returns from the call, R0 still contains the character. If you omit this address, RSTS/E prints the character in the low byte of R0 at the job's terminal.

Errors

No errors are possible with the .TTYOUT and the .TTOUTR directives under RSTS/E.

Example

The following example transfers one character from the low byte of R0 to the job's terminal:

```
 PROMPT: .ASCII    /*/
           .
           .
           .
         MOV       #PROMPT,R0
         .TTYOUT
```

.TWAIT - Timed Wait

The .TWAIT directive suspends execution of the job for a specified number of clock ticks: a sleep, in RSTS/E terms.

Privileges Required

None

Macro Call

```
   .TWAIT area,addr
```

where:

area	is the address of a two-word argument block.
addr	is the address of a two-word block specifying the length of time to sleep in units of 1/60th second. The first word contains the high-order bits; the second word contains the low-order bits.

Argument Block Format

```
+----------+----------+
|    24    |     0    |
+----------+----------+
|        addr         |
+----------+----------+
```

Errors

No errors are possible with the .TWAIT directive.

Example

The following code suspends program execution for 5 seconds:

```
  AREA:   .BLKW     2
  ADDR:   .BLKW     2
             .
             .
             .
          CLW       #ADDR
          MOV       #300.,#ADDR+2
          .TWAIT    #AREA,#ADDR
```

.WAIT - Check for Channel Open

On RSTS/E systems, the .WAIT directive checks to make sure that the specified channel is open. If not, the call returns with an error.

Privileges Required

None

Macro Call

.WAIT *chanum*

where:

chanum is the channel number to be tested. The number can be in the range 0 to 17 (octal).

R0 Format

```
      +----------+----------+
R0 =  |    0     |  chanum  |
      +----------+----------+
```

Error

Code	**Meaning**
0	The specified channel is not open.

Example

The following code checks to see if channel 7 is open and branches if it is not:

```
.WAIT   #7.
BCS     #NOTOPN
```

.WRITE/.WRITW/.WRITC -- Write Data

On RSTS/E systems, .WRITE, .WRITW, and .WRITC transfer data from a user buffer to a file or device previously opened on a channel. After the transfer is complete, control is passed to the user program inline, for .WRITE and .WRITW, or to a user-specified completion routine, for .WRITC. The number of words actually written is returned in R0 when the call completes.

Privileges Required

EXTQA privilege suppresses disk quota checks when extending a disk file.

Macro Call

$$\left\{\begin{array}{l}\texttt{.WRITE}\\ \\ \texttt{.WRITW}\end{array}\right\}\ \textit{area,chanum,bufadd,wrdcnt,blknum}$$

or

.WRITC *area,chanum,bufadd,wrdcnt,cmprtn,blknum*

where:

area is the address of a five-word argument block.

chanum is a channel number in the range 0 to 17 (octal), previously defined in an open (.ENTER or .LOOKUP).

bufadd is the address of the buffer containing the data to be written.

wrdcnt is the number of words to be written.

cmprtn is the address of a completion routine to which control is transferred when the write is complete. (For .WRITC only.)

For .WRITE and .WRITW, control is returned to the instruction following the directive when the write is complete.

blknum is the block number relative to the start of the file where the write is to begin. (For files opened with file-structured .ENTER or .LOOKUP.)

For devices opened with non-file-structured .ENTER or .LOOKUP, the *blknum* argument is used as the absolute block number or device cluster number where the write is to begin.

Note

To access large files on RSTS/E systems, you must set the word at offset 2 in the scratch pad area (see Chapter 6).

To do successive writes, your program must increment *blknum*.

Argument Block Format

```
+----------+----------+
|    11    |  chanum  |
+----------+----------+
|       blknum        |
+----------+----------+
|       bufadd        |
+----------+----------+
|       wrdcnt        |
+----------+----------+
|       cmprtn*       |
+----------+----------+
```
*The last word is set to 1 for .WRITE or to 0 for .WRITW.

Errors

Code	Meaning
0	No more room is available on the device. For example, the end-of-tape (EOT) marker has been encountered, no more available disk space, and so forth.
1	Some hard I/O device error occurred. For example, the specified device is offline, is physically write-locked, and so forth.
2	The channel specified is not open.

Example

The following code writes a 256-word block to the device open on channel 12:

```
AREA:   .BLKW     5
BUF:    .BLKB     512.
          .
          .
          .
        .WRITE    #AREA,#12.,#BUF,#256.,#0
```

..V1../..V2.. - Use Version 1/Version 2 Expansion

If you have a source program that was coded for Version 1 or Version 2 of the RT-11 operating system and you want to assemble on RSTS/E using the expansions that were used for these versions, you must use either the ..V1.. directive (for Version 1 expansions) or the ..V2.. directive (for Version 2 expansions).

You cannot use both the ..V1.. and ..V2.. directives. You can use directives from more than one version by specifying the ..V1.. directive and simply using the directives from later versions. If the same directive exists in two or three versions, though, you get the expansion from the earliest version.

These directives generate code that defines and sets a variable called ...V1 to either 1 or 2, which the MACRO assembler uses to determine the expansions to use. So, if the ..V1.. or ..V2.. directive is used, it must appear before any other RT11 directives in your program.

Note

> All the examples in this chapter show Version 2 expansions. Version 1 expansions require different arguments in the calls. See RT-11 system documentation if you need to use version 1 expansions.

Privileges Required

None

Macro Call

```
..V1..
```

or

```
..V2..
```

Appendixes

This appendix lists the error messages that you may encounter when using RSTS/E. The question mark (?) and percent sign (%) characters preceding the full error text indicate the error's severity:

- o A single question mark precedes nonfatal errors; the command failed but you can continue
- o A double question mark precedes severe errors; the command failed and you cannot continue
- o A percent sign precedes a warning message; the command may not have performed as you intended but execution continues

Informational messages have neither character; they provide extra detail about the effects of a command.

Table A-1 summarizes the RSTS/E error messages.

Table A-1: RSTS/E Errors

ERR.STB Mnemonic	Decimal Value	Octal Value	Full Error Text
BADDIR	1	1	??Bad directory for device
BADNAM	2	2	?Illegal file name
INUSE	3	3	?Account or device in use
NOROOM	4	4	?No room for user on device
NOSUCH	5	5	?Can't find file or account
NODEVC	6	6	?Not a valid device
NOTCLS	7	7	?I/O channel already open
NOTAVL	8	10	?Device not available
NOTOPN	9	11	?I/O channel not open
PRVIOL	10	12	?Protection violation
EOF	11	13	?End of file on device
ABORT	12	14	??Fatal system I/O failure
DATERR	13	15	?Data error on device
HNGDEV	14	16	?Device hung or write locked
HNGTTY	15	17	?Keyboard wait exhausted

Table A-1: RSTS/E Errors (Cont.)

ERR.STB Mnemonic	Decimal Value	Octal Value	Full Error Text
FIEXST	16	20	?Name or account now exists
DTOOOF	17	21	?Too many open files on unit
BADFUO	18	22	?Illegal SYS () usage
INTLCK	19	23	?Disk block is interlocked
WRGPAK	20	24	?Pack IDs don't match
NOTMNT	21	25	?Disk pack is not mounted
PAKLCK	22	26	?Device is restricted
BADCLU	23	27	?Illegal cluster size
PRIVAT	24	30	?Account does not exist
INTPAK	25	31	%Disk pack needs REBUILDing
BADPAK	26	32	??Disk pack mount error
DETKEY	27	33	?I/O to detached keyboard
CTRLCE	28	34	Programmable ^C trap
SATTBD	29	35	??Corrupted file structure
DEVNFS	30	36	?Device not file-structured
BADCNT	31	37	?Illegal byte count for I/O
NOBUFS	32	40	?No buffer space available
B.4	33	41	??Odd address trap
B.10	34	42	??Reserved instruction trap
B.250	35	43	??Memory management trap
B.STAK	36	44	??SP stack overflow
B.SWAP	37	45	??Disk error during swap
B.PRTY	38	46	??Memory parity failure
MAGSEL	39	47	?Maptape select error
MAGRLE	40	50	?Maptape record length error
NRRTS	41	51	??Non-res run-time system
VCSERR	42	52	?Virtual buffer too large
VCAERR	43	53	?Virtual array not on disk
SIZERR	44	54	?Matrix or array too big
VCOERR	45	55	?Virtual array not yet open
BSERR	46	56	?Illegal I/O channel
LINERR	47	57	?Line too long
FLTERR	48	60	%Floating point error
EXPERR	49	61	%Argument too large in EXP
FMTERR	50	62	%Data format error
FIXERR	51	63	%Integer error
BDNERR	52	64	?Illegal number
LOGERR	53	65	%Illegal argument in LOG
SQRERR	54	66	%Imaginary square roots
SUBERR	55	67	?Subscript out of range

Table A-1: RSTS/E Errors (Cont.)

ERR.STB Mnemonic	Decimal Value	Octal Value	Full Error Text
MINVER	56	70	?Can't invert matrix
ODD	57	71	?Out of data
ONBAD	58	72	?ON statement out of range
NEDERR	59	73	?Not enough data in record
IOLERR	60	74	?Integer overflow, FOR loop
DIVBY0	61	75	%Division by 0
NORTS	62	76	?No run-time system
FIELDE	63	77	?FIELD overflows buffer
NORACS	64	100	?Not a random access device
NOTMTA	65	101	?Illegal MAGTAPE() usage
ERRERR	66	102	?Missing special feature
BADSWT	67	103	?Illegal switch usage
EOV	68	104	?End of volume
QUOTA	69	105	?Quota exceeded
	70	106	??Unused ERROR message 70
STMERR	71	107	?Statement not found
EXITTM	72	110	?RETURN without GOSUB
EXITNR	73	111	?FNEND without function call
UNDFNI	74	112	?Undefined function called
COSERR	75	113	?Illegal symbol
TLOPNV	76	114	?Illegal verb
TLNZSP	77	115	?Illegal expression
TLNOIT	78	116	?Illegal mode mixing
TLIFFE	79	117	?Illegal IF statement
TLCONI	80	120	?Illegal conditional clause
TLNOTF	81	121	?illegal function name
TLQDUM	82	122	?Illegal dummy variable
TLMFND	83	123	?Illegal FN redefinition
TLRNNM	84	124	?Illegal line number(s)
MODERR	85	125	?Modifier error
	86	126	?Can't compile statement
OUTOAS	87	127	?Expression too complicated
FUNERR	88	130	?Arguments don't match
TLTMAF	89	131	?Too many arguments
TLINCD	90	132	%Inconsistent function usage
CPNSDF	91	133	?Illegal DEF nesting
CPUPFR	92	134	?FOR without NEXT
CPUFNX	93	135	?NEXT without FOR
CPUPDF	94	136	?DEF without FNEND
CPUPED	95	137	?FNEND without DEF

Table A-1: RSTS/E Errors (Cont.)

ERR.STB Mnemonic	Decimal Value	Octal Value	Full Error Text
TLJNKY	96	140	?Literal string needed
TLNOFN	97	141	?Too few arguments
SASYNE	98	142	?Syntax error
SAFNOS	99	143	?String is needed
SASNOI	100	144	?Number is needed
TLURTP	101	145	?Data type error
TLXDIM	102	146	?1 or 2 dimensions only
FUCORE	103	147	??Program lost-sorry
RESERR	104	150	?RESUME and no error
DIMED2	105	151	?Redimensioned array
TLIDIM	106	152	%Inconsistent subscript use
NOGOTO	107	153	?ON statement needs GOTO
EOSERR	108	154	?End of statement not seen
TLCNTD	109	155	?What?
TLPRNM	110	156	?Bad line number pair
EDBMCE	111	157	?Not enough available memory
EDEXON	112	160	?Execute only file
NRNERR	113	161	?Please use the RUN command
EDCONE	114	162	?Can't CONTinue
EDARSV	115	163	?File exists-RENAME/REPLACE
PRERRS	116	164	?PRINT-USING format error
UDMERR	117	165	?Matrix or array without DIM
PRNER1	118	166	?Bad number in PRINT-USING
NONOIM	119	167	?Illegal in immediate mode
PRNER2	120	170	?PRINT-USING buffer overflow
BADERR	121	171	?Illegal statement
DISERR	122	172	?Illegal FIELD variable
STPERR	123	173	Stop
DIMERR	124	174	?Matrix dimension error
NOMATH	125	175	?Wrong math package
XCDCOR	126	176	??Maximum memory exceeded
SCAERR	127	177	%SCALE factor interlock
	128	200	?Tape records not ANSI
	129	201	?Tape BOT detected
	130	202	?Key not changeable
	131	203	?No current record
	132	204	?Record has been deleted
	133	205	?Illegal usage for device
	134	206	?Duplicate key detected
	135	207	?Illegal usage

Table A-1: RSTS/E Errors (Cont.)

ERR.STB Mnemonic	Decimal Value	Octal Value	Full Error Text
	136	210	?Illegal or illogical access
	137	211	?Illegal key attributes
	138	212	?File is locked
	139	213	?Invalid file options
	140	214	?Index not initialized
	141	215	?Illegal operation
	142	216	?Illegal record on file
	143	217	?Bad record identifier
	144	220	?Invalid key of reference
	145	221	?Key size too large
	146	222	?Tape not ANSI labeled
	147	223	?RECORD number exceeds max
	148	224	?Bad RECORDSIZE on OPEN
	149	225	?Not at end of file
	150	226	?No primary key specified
	151	227	?Key field beyond record end
	152	230	?Illogical record accessing
	153	231	?Record already exists
	154	232	?Record/bucket locked
	155	233	?Record not found
	156	234	?Size of record invalid
	157	235	?Record on file too big
	158	236	?Primary key out of sequence
	159	237	?Key larger than record
	160	240	?File attributes not matched
	161	241	?Move overflows buffer
	162	242	?Cannot open file
	163	243	?No file name
	164	244	?Terminal fmt file required
	165	245	?Cannot position to EOF
	166	246	?Negative fill or string len
	167	247	?Illegal record format
	168	250	?Illegal ALLOW clause
	169	251	??Unused ERROR message 169
	170	252	?Indexed not fully optimized
	171	253	?RRV not fully updated
	172	254	?Record LOCK failed
	173	255	?Invalid RFA field
	174	256	?Unexpired file date
	175	257	?Node name error

Table A-1: RSTS/E Errors (Cont.)

ERR.STB Mnemonic	Decimal Value	Octal Value	Full Error Text
	176	260	?Negative TAB not allowed
	177	261	?Too much data in record
	178	262	?OPEN error - file corrupted
	179	263	??Unused ERROR message 179
	180	264	?No support for op in task
	181	265	?Decimal overflow
	182	266	?Network operation rejected
	183	267	?REMAP overflows buffer
	184	270	?Unaligned REMAP variable
	185	271	%RECORDSIZE overflows MAP
	186	272	?Improper error handling
	187	273	?Illegal record lock clause
	188	274	??Unused ERROR message 188
	189	275	??Unused ERROR message 189
	190	276	??Unused ERROR message 190
	191	277	??Unused ERROR message 191
	192	300	??Unused ERROR message 192
	193	301	??Unused ERROR message 193
	194	302	??Unused ERROR message 194
	195	303	??Unused ERROR message 195
	196	304	??Unused ERROR message 196
	197	305	??Unused ERROR message 197
	198	306	??Unused ERROR message 198
	199	307	??Unused ERROR message 199
	200	310	??Unused ERROR message 200
	201	311	??Unused ERROR message 201
	202	312	??Unused ERROR message 202
	203	313	??Unused ERROR message 203
	204	314	??Unused ERROR message 204
	205	315	??Unused ERROR message 205
	206	316	??Unused ERROR message 206
	207	317	??Unused ERROR message 207
	208	320	??Unused ERROR message 208
	209	321	??Unused ERROR message 209
	210	322	??Unused ERROR message 210
	211	323	??Unused ERROR message 211
	212	324	??Unused ERROR message 212
	213	325	??Unused ERROR message 213
	214	326	??Unused ERROR message 214
	215	327	??Unused ERROR message 215

Table A-1: RSTS/E Errors (Cont.)

ERR.STB Mnemonic	Decimal Value	Octal Value	Full Error Text
	216	330	??Unused ERROR message 216
	217	331	??Unused ERROR message 217
	218	332	??Unused ERROR message 218
	219	333	??Unused ERROR message 219
	220	334	??Unused ERROR message 220
	221	335	??Unused ERROR message 221
	222	336	??Unused ERROR message 222
	223	337	??Unused ERROR message 223
	224	340	??Unused ERROR message 224
	225	341	??Unused ERROR message 225
	226	342	??Unused ERROR message 226
	227	343	?String too long
	228	344	?RECORDTYPES not matched
	229	345	??Unused ERROR message 229
	230	346	?No fields in image
	231	347	?Illegal string image
	232	350	?Null image
	233	351	?Illegal numeric image
	234	352	?Numeric image for string
	235	353	?String image for numeric
	236	354	?TIME limit exceeded
	237	355	?1st arg to SEQ$ > 2nd
	238	356	?Arrays must be same DIM
	239	357	?Arrays must be square
	240	360	?Cannot change array DIMs
	241	361	?Floating overflow
	242	362	?Floating underflow
	243	363	?CHAIN to non-existent line
	244	364	?Exponentiation error
	245	365	?Illegal exit from DEF*
	246	366	?Error trap needs RESUME
	247	367	?Illegal RESUME to SUBR
	248	370	?Illegal subroutine return
	249	371	?Argument out of bounds
	250	372	?Not implemented
	251	373	?Recursive subroutine call
	252	374	?FILE ACP failure
	253	375	?Directive error
	254	376	??Unused ERROR message 254
	255	377	??Unused ERROR message 255

Appendix B: Device Information

This appendix summarizes MODE and RECORD values and other useful information for:

- o Disks
- o Flexible diskettes
- o Magnetic tape
- o Line printers
- o Terminals
- o Pseudo keyboards

For your convenience, the tables show both decimal and octal values, and list the FIRQB and XRB offsets. All decimal values have a decimal point; values without a decimal point are in octal.

This appendix is a quick reference. See the *RSTS/E Programming Manual* (except where otherwise noted) if you need more detail on any topic.

Disks

This section summarizes disk MODE values and lists the device cluster size and total size for each disk that RSTS/E supports.

Table B-1 summarizes disk MODE values for file-structured access.

Table B-1: MODE Values for File-Structured Disk Access (FIRQB+FQMODE)

Decimal	Octal	Function
0.	0	Normal read/write
1.	1	Update
2.	2	Append
4.	4	Guarded update (4 + 1)
8.	10	Special extend (RSTS/E updates file's size and retrieval pointers during extend operations)
16.	20	Create contiguous file
32.	40	Create tentative file
64.	100	Create contiguous file conditionally
128.	200	No supersede
256.	400	Random data caching (requires TUNE privilege)
512.	1000	Create file and place at beginning of directory (with 2000)
1024.	2000	Create file and place at end of directory
2048.	4000	Sequential data caching (with 400)
4096.	10000	Read normally regardless
8192.	20000	Open file read-only
16384.	40000	Write UFD (requires WRTNFS privilege)

Table B-2 summarizes disk MODE values for non-file-structured access.

Table B-2: MODE Values for Non-File-Structured Disk Access (FIRQB+FQMODE)

Decimal	Octal	Function
0.	0	Access device clusters.
128.	200	Access disk blocks.
512.	1000	Read beyond last writable portion of disk; suppress error logging. (Reserved for use by DIGITAL.)

Table B-3 lists the device cluster size and device size (in 512-byte blocks) for each disk that RSTS/E supports. All values are decimal.

Table B-3: Disk Device Sizes

Disk	Device Cluster Size	Device Size
RX50	1	800
RK05	1	4800
RK05F	1	4800 per unit; 2 units for each drive
RL01	1	10220
RL02	1	20460
RD51	1	21600
RK06	1	27104
RK07	1	53768
RC25	1	50902 per unit; 2 units per spindle
RP02	2	40000
RP03	2	80000

Table B-3: Disk Device Sizes (Cont.)

Disk	Device Cluster Size	Device Size
RM02	4	131648
RM03	4	131648
RP04	4	171796
RP05	4	171796
RA80	4	237208
RM80	4	242575
RP06	8	340664
RA60	8	400175
RM05	8	500352
RA81	16	888012

Flexible Diskettes

This section summarizes MODE and RECORD values for flexible diskettes. Table B-4 lists MODE values for flexible diskettes.

Table B-4: Flexible Diskette MODE Values (FIRQB+FQMODE)

Decimal	Octal	Function
0.	0	Block mode
16384.	40000	Sector mode

Table B-5 lists RECORD values for flexible diskettes.

Table B-5: Flexible Diskette RECORD Values (XRB+XRBLK)

Decimal	Octal	Function
8192.	20000	Access logical record 0
16384.	40000	Write deleted data mark
32767.+1.	100000	Perform this I/O operation in block mode

Magnetic Tape

This section summarizes MODE and CLUSTERSIZE values for magnetic tape. Table B-6 lists MODE values for file-structured magnetic tape.

Table B-6: MODE Values for File-Structured Magnetic Tape (FIRQB+FQMODE)

Decimal	Octal	Function
0.	0	Read file label at current tape position
2.	2	Do not rewind tape when searching for file
16.	20	Write over existing file
32.	40	Rewind tape before searching for file
64.	100	Rewind on CLOSE
128.	200	Open for append
512.	1000	Write new file label without searching
16384.	40000	Search for DOS-formatted file label
24576.	60000	Search for ANSI-formatted file label

Table B-7 lists CLUSTERSIZE values for file-structured magnetic tape. The value in FIRQB+FQCLUS is the sum of the values chosen from Table B-7. The default value is zero.

Note

The CLUSTERSIZE option is only effective when processing ANSI-formatted magnetic tape.

Table B-7: CLUSTERSIZE Values for File-Structured Magnetic Tape Files

Label Field Name	Decimal	Octal	Label Result
Record format	0.	0	U = undefined.*
	16384.	40000	F = fixed-length.
	32767.+1.	100000	D = variable length.
	-16384.	140000	S = spanned.**
Record length (in bytes)	Between 0. and 4095.	Between 0 and 7777	For U, always 0. For F, fixed record length. For D, maximum record length. For S, not used.**
System Dependent (File characteristics)	0.	0	M = carriage control embedded.
	4096.	10000	A = FORTRAN carriage control.
	8192.	20000	(space) = implied carriage control. When printed, line feed precedes and carriage return follows each record.

* RSTS/E undefined record formats cannot be processed directly by most of the other operating systems.
** RSTS/E does not support ANSI format S records.

MODE values in non-file-structured magnetic tape processing have the form:

```
  MODE(FIRQB+FQMODE) = D + P + S
```

where:

```
  D (density) = 12.(14) for 800 bpi
                256.(400) for 1600 bpi, phase-encoded

  P (parity) = 0.(0) for odd
               1.(1) for even (800 bpi only)

  S (stay)   = 0.(0) to clear MODE value after CLOSE
               8192.(20000) to retain MODE value after CLOSE
```

Note that DIGITAL recommends the use of odd parity. When you use even parity, you cannot write binary data. In addition, many operating systems and tape drives do not support even parity.

For information on magnetic tape special functions, the magnetic tape status word, and the file characteristics word, see the description of .SPEC for magnetic tape (see Chapter 3).

Line Printers

This section summarizes line printer MODE and RECORD values. Table B-8 lists the MODE values for line printers.

Table B-8: Line Printer MODE Values (FIRQB+FQMODE)

Decimal	Octal	Function
1. to 127.	1 to 177	Sets form length to number of lines per page for software formatting (512., 1000) and automatic page skip (2048., 4000).
128.	200	Changes the character 0 (zero) to O (letter "oh").
256.	400	Truncates lines that are longer than unit was configured for instead of printing the rest of the line on the next physical line on the page.
512.	1000	Enables software formatting. Forms control characters are $\geq$ 200(8).
1024.	2000	Translates lowercase characters to uppercase characters. Applies only to uppercase and lowercase line printers.
2048.	4000	Skips six lines (that is, skips over perforation line) at the bottom of each form.
4096.	10000	Moves paper to top of hardware form.
8192.	20000	Suppresses form feed on CLOSE.

Table B-9 lists the RECORD values for line printers.

Table B-9: Line Printer RECORD Values (XRB+XRMOD)

Decimal	Octal	Function
2.	2	Print over perforation (disables MODE 4000 for this output step).
4.	4	Do not return control to program until output is complete or an error occurs.
8.	10	Clear pending output buffers before buffering characters for the request.
8192.	20000	Return control to program if output stall is to occur. (See the .WRITE directive for more information.)

Terminals

This section summarizes MODE and RECORD values for terminals. It also includes information about echo control mode and VT100 ANSI-compatible escape sequences.

Table B-10 lists MODE values for terminal input.

Table B-10: Terminal MODE Values (FIRQB+FQMODE)

Decimal	Octal	Function
1.	1	Enable binary input from a terminal
2.	2	Reserved for TECO
4.	4	Suppress automatic CR/LF at right margin
8.	10	Enable echo control (turns off other modes and automatically enables MODE 4)
16.	20	Guard program against CTRL/C interruption and dial-up line hibernation
32.	40	Enable incoming XON/XOFF processing
64.	100	Reserved
128.	200	Enable use of RUBOUT as a delimiter on video terminals
256.	400	Set escape sequence mode
16384.	40000	Transparent control character mode

Table B-11 lists RECORD values for terminal input.

Table B-11: RECORD Values for Terminal Input (XRB+XRMOD)

Decimal	Octal	Function	
256.	400	Force interactive read (always obtain data from the user's terminal, never from a command file)	
8192.	20000	Perform conditional input (execute input request without waiting for data to be available).	
32767.+1.+K	100000+K	Perform multiterminal service input from assigned keyboard number K.	
32767.+1.+ 16384.+S	140000+S	Perform multiterminal service input from any assigned keyboard:	
		S = 0	Wait until input is available from any terminal. The error ?Data error on device may occur due to a race condition with CTRL/C.
		1. < S < 255. 1 < S < 377	Wait up to S seconds for input from any terminal and then return ?Data error on device if no input is available.

		S = 8192.
		S = 20000
			Request input immediately; return ?Data error on device if no input is pending.

Table B-12 lists RECORD values for terminal output.

Table B-12: RECORD Values for Terminal Output (XRB+XRMOD)

Decimal	Octal	Function
256.	400	Declare echo control field (use with MODE 10).
4096.	10000	Output binary data to terminal.
8192.	20000	Return control to program if output stall is to occur. (See the .WRITE directive for more information.)
16384.	40000	Transparent control character mode.
32767.+1.+K	100000+K	Perform multiterminal service output to assigned keyboard K.

In echo control mode, the system strips the parity bit from all characters. All characters returned to your program have ASCII values in the range 1 to 255. The system does not pass synchronization or editing characters to your program. Delimiters are passed to your program but are never echoed.

Table B-13 lists the echo control mode character set.

Table B-13: Echo Control Mode Character Set

Type of Character	ASCII Code (Octal)	Code Returned to User	Comments
Ignored	0		Used as filler for timing
Delimiters	Private	?	Private delimiter.
	3	3	^C (CTRL/C).
	4	4	^D (CTRL/D).
	12	12	Line feed.
	14	14	Form feed.
	15	15,12	Carriage return (with line feed appended).
	32	32	^Z (CTRL/Z); generates error 11(10).
	33	33	If the Escape_Sequence characteristic is not in effect and escape character is received, 33 is returned to user and is treated as a delimiter. If the Escape_Sequence characteristic is in effect, escape character triggers an escape sequence. The escape sequence is returned to user and the whole sequence is considered the delimiter.
	175	33 or 175	If the Alt_Mode characteristic is in effect, 175 is translated to escape (33). If the Alt_Mode characteristic is not in effect, 175 is data.

Table B-13: Echo Control Mode Character Set (Cont.)

Type of Character	ASCII Code (Octal)	Code Returned to User	Comments
	176	33 or 176	If the Alt_Mode characteristic is in effect, 176 is translated to escape (33). If the Alt_Mode characteristic is not in effect, 176 is data.
Editing	177	-	RUBOUT (DEL character); on video terminals, generates a backspace followed by the paint character and another backspace; on hard-copy terminals, echoes deleted characters between backslashes(\\).
	25	-	^U (CTRL/U); repeatedly simulates RUBOUT until no characters remain in field.
Data	40-137 241-337	40-137 241-337	Normal 64-character graphic set.
	140-176 340-376	100-136 300-336	If the Lowercase=input characteristic is not set, lowercase letters are translated to uppercase.
	140-176 340-376	140-176 340-376	If the Lowercase=input characteristic is set, lowercase letters are returned to user.
Synchro-nization	21	-	XON (CTRL/Q). Resume suspended output (if the TTSync characteristic is set).
	23	-	XOFF (CTRL/S). Suspend output (if the TTSync characteristic is set).

Table B-13: Echo Control Mode Character Set (Cont.)

Type of Character	ASCII Code (Octal)	Code Returned to User	Comments
Other	1,2,5,6 7,10,11, 13,16-20, 22,24, 26-31, 34-37	-	Echoed as BEL (7) but otherwise ignored.
	21,23	-	If the terminal is set No TTSync, synchronization characters are also echoed as BEL (7) and ignored.

Use .WRITE to declare a field. The .WRITE must include the value 400 at XRB+XRMOD and the value N at XRB+XRBC. N, which must be between 1 and the size of the buffer, describes how many bytes in the buffer represent the field declaration:

N = 1	The byte contains field size and overflow handling information. The field size must be in the range 1 to 177. Adding 200 to the field size specifies keypunch overflow handling instead of normal overflow handling.
N = 2	The first byte contains field size and overflow handling as described for N = 1. The second byte contains the ASCII value of the paint character. If this byte is 0 or N = 1, a space is the paint character.
N > 2	The first (N-2) bytes contain a prompt to display on the terminal before the field. Byte (N-1) is the field size declaration as described for N = 1. The last byte is the paint character as described for N = 2.

Pseudo Keyboards

This section summarizes MODE and RECORD values for pseudo keyboards. It also lists errors your program can receive on a pseudo keyboard output request.

Table B-14 lists MODE values for pseudo keyboards.

Table B-14: Pseudo Keyboard MODE Values (FIRQB+FQMODE)

Decimal	Octal	Function
0.	0	System kills controlled job when the pseudo keyboard is closed
1.	1	System detaches controlled job when the pseudo keyboard is closed

Table B-15 lists RECORD values for pseudo keyboards.

Table B-15: RECORD Option Bit Values for Pseudo Keyboard Output (XRB+XRMOD)

Bit	Decimal	Octal	Result
0	1.	1	If set, the system does not check job status before sending data to the pseudo keyboard.
1	2.	2	If set, the system tests whether pseudo keyboard is waiting for a system command (CTRL/C state) or is waiting for program input (KB wait state).
2	4.	4	If set, the system does not send data to the pseudo keyboard but instead returns control to the controlling job.
3	8.	10	If set, and there are no small buffers for keyboard input, the system waits until small buffers are available. However, your program receives the NOROOM error if the output buffer chain is full.
4	16.	20	If set, the system kills the job currently running at the pseudo keyboard.

Table B-16 lists errors that your program can receive on a pseudo keyboard output request.

Table B-16: Possible Errors on Pseudo Keyboard Output Request

Error	Meaning
INUSE	Job at pseudo keyboard is not ready for input
NOROOM	No buffer space is available
NOSUCH	No controlled job exists at pseudo keyboard
CTRLCE	Job at pseudo keyboard is not in CTRL/C state

INDEX

RSTS/E
System Directives Manual
AA-EZ10A-TC

Reader's Comments

Note: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement. ____

____

____

____

____

____

____

Did you find errors in this manual? If so, specify the error and the page number. ________________

____

____

____

____

____

____

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) ____________________________

Name ______________________ Date ______________________

Organization ____

Street ____

City ______________________ State ____________ Zip Code or Country ____________

--Do Not Tear - Fold Here and Tape --

digital

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN: Office Systems Publications MK01-2/E02
RSTS/E Documentation
DIGITAL EQUIPMENT CORPORATION
CONTINENTAL BOULEVARD
MERRIMACK, N.H. 03054

--Do Not Tear - Fold Here and Tape --